1 EXPERIENCES IN MINING DATA FROM COMPUTER SIMULATIONS

Chandrika Kamath, Erick Cantú-Paz, Sen-ching S. Cheung, Imola K. Fodor, Nu Ai Tang

Lawrence Livermore National Laboratory
Livermore, California

1.1 INTRODUCTION

Computer simulations enable us to understand complex phenomena through the analysis of mathematical models on high performance computers. By filling the gap between physical experiments and analytical approaches, computer simulations provide both qualitative and quantitative insights into physical phenomena. They are particularly useful when the phenomena is too complex to be analyzed analytically, such as the flow around an airplane, or too expensive, impractical, or dangerous to study experimentally. In the latter case, we include examples such as the effects of a volcano eruption on the earth's surface temperature, the evolution of stars, car crash tests, the structural integrity of buildings and bridges under various wind conditions and loads, modeling of material behavior at macro- and micro-scales, the spread of diseases via entity-based simulations, etc.

In this chapter we survey the work being done in the mining of simulation data, describe our experiences in the analysis of data from a fluid mixing problem, and outline the challenges and opportunities in the mining of this relatively new form of data. We first provide a brief introduction to simulation data in Section 1.2, followed by a survey of various applications in Section 1.3. Then, in Section 1.4, we discuss our work in similarity-based object retrieval, and illustrate how it can be used in several problems of interest. Finally, in Section 1.5 we identify open problems in mining simulation data as well as potential solution approaches.

1.2 A BRIEF INTRODUCTION TO SIMULATION DATA

To understand the role data mining can play in the analysis of data from simulations, we first need to understand the source of this data. Numerical simulations typically involve the solution of partial differential equations (PDEs). Using fundamental principles such as the conservation of mass, momentum, and energy, partial differential equations create a mathematical model of a physical phenomena. These models typically have several independent variables, such as space (i.e., x, y, z coordinates) and time, and several dependent variables (e.g., pressure, density, and velocity). The PDE describes how the dependent variables vary as a function of the independent variables. Since it is a continuous formulation, the PDE is solved numerically by discretizing it in space and time using mesh-based methods such as finite elements and finite differences, or through mesh-less methods. Then, as the PDE evolves over time, at each time step, determined by the discretization in time, we obtain the values of the dependent variables at each grid point, that is, the location in space identified by the spatial discretization.

The data from numerical simulations can be quite large, especially as complex phenomena are simulated in two and three spatial dimensions. For example, the simulation of a high resolution 3-D shock tube was performed on a $2048 \times 2048 \times 1920$ grid over 27,000 time steps, on 960 nodes of the IBM-SP Sustained Stewardship TeraOp system at Lawrence Livermore National Laboratory [23]. The simulation ran for just over a week, producing over 3 Terabytes of graphics data (one byte per grid point), distributed over 300,000 files. In addition, much larger data sets were generated for the dynamical variables (e.g., density, pressure, velocity, material fraction) which were stored in 16-bit integer format. While this may be an extreme example, even moderate-sized computer simulations can produce datasets that require automated analysis techniques. As an example, consider the re-analysis data from the National Centers for Environmental Predictions [28]. The data is the result of a simulation in atmospheric sciences which outputs the mean temperature for each month. It is available over a 144×73 longitude-by-latitude grid, with 17 elevation levels, for 264 months. Even in these relatively smaller data sets, data mining can play an important role when it is used either in isolation, or as a complement to the more traditional visualization approaches.

When mesh-based methods are used in computer simulations, the values of the variables are available at discrete mesh or grid points. Mesh data from simulations can be broadly categorized into structured and unstructured data. Structured data can be either regular in the form of a Cartesian mesh, panel (a) in Figure 1.1, or curvilinear as shown in panel (b) in Figure 1.1. In the case of a regularly-spaced Cartesian mesh, techniques from image processing can be applied to pre-process the mesh data. In the unstructured mesh data category, for example panel (c) in Figure 1.1, the data can be composed of either homogeneous elements (in this case triangles) or heterogeneous elements as in Figure 1.2. In addition to structured and unstructured meshes there are also meshes that are the hybrid of the two - these include hierarchical meshes which are globally unstructured but locally structured (Figure 1.3), as well as meshes that are structured in one part of the problem domain and unstructured

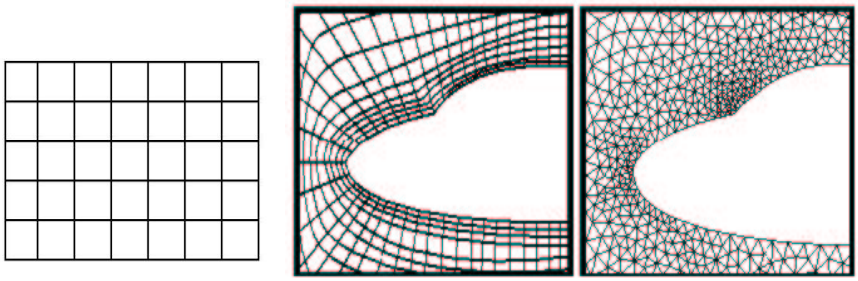


Fig. 1.1 Examples of two-dimensional meshes: (a) an image or a regularly-space Cartesian mesh, (b) two-dimensional structured mesh, and (c) unstructured mesh around the front of an aircraft. Panels (b) and (c) from http://www.nas.nasa.gov/Pubs/Docs/FAST/chp_16.surferu.html

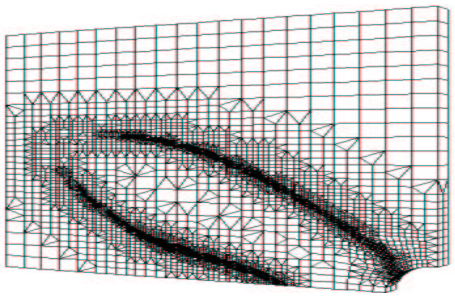


Fig. 1.2 An example of a three-dimensional unstructured mesh with heterogeneous elements (http://cox.iwr.uni-heidelberg.de/ ug/Images/benchmark_grid.gif)

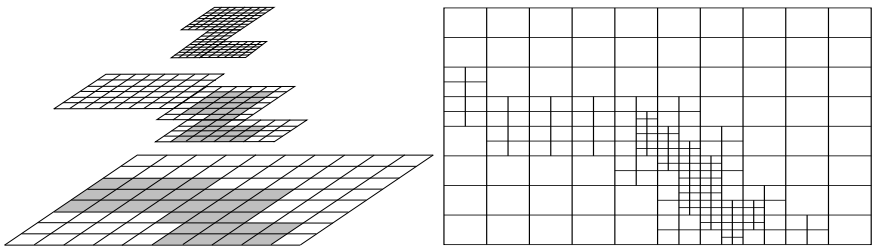


Fig. 1.3 An example of a composite mesh made up of regular meshes illustrating (a) the hierarchy of regular meshes and (b) the composed "unstructured mesh". (http://www.llnl.gov/casc/samrai)

in another. In many problems, the meshes are not static - as the physical features in the simulation evolve, the mesh too must adaptively change to accurately model these features. There are also mesh-less methods used in simulations that result in irregularly spaced output data. In addition, as the simulation evolves in time, a series of two- or three-dimensional data sets are produced and output at each time step.

1.3 ROLE OF DATA MINING IN COMPUTER SIMULATIONS

In this section we survey the role that data mining is beginning to play in the area of computer simulations. Simulation data is a relatively new source of massive data and, as a result, there has been little work done in mining such data. However, there are several examples where techniques commonly used in data mining, such as neural networks or principal component analysis, are being used in the context of computer simulations. We will include such examples in our survey of existing work to illustrate application areas where data mining can play a greater role.

1.3.1 Detection and tracking of coherent structures

There are several examples of the use of data mining techniques to identify and track coherent structures. For instance, [11] use a fuzzy ARTMAP neural network to identify eddy motions in a turbulent wake flow, using data from a wind tunnel. Frames of 3×3 adjacent velocity vectors are considered, and patterns are represented by normalized velocity vectors. From this data, the training patterns are selected by manually identifying patterns that represent clockwise and counterclockwise structures. The neural network is applied multiple times varying a "vigilance" parameter. Those patterns labeled as eddies by all the neural networks are reported as the final results. The results agree with previous work that used template matching to find the eddies.

In other applications, there is an interest to identify and track coherent structures over multiple time steps. Traditionally, tracking is done by a human or by computationally expensive heuristics that track objects through all intermediate time steps. Banerjee, Hirsch, and Ellman [2] propose an alternative tracking method based on machine learning techniques and describe an application to track vortices. Vortices change shape, split, merge, appear, and cease to exist unexpectedly. Their method begins by constructing a training data set composed of vectors that describe related and unrelated vortices after a time gap. The training set is used to create a decision tree, which is then applied to identify related vortices in different time steps. As expected, the accuracy degrades as the time gap increases. The decision tree performs worse than the tracking approach for small time gaps, but over intermediate and long time gaps, the decision tree performs better than a traditional tracking program.

Another technique that has been extensively used for the analysis of coherent structures is wavelets [8]. As wavelets operate on different scales, the information they extract from the data are ideal for identifying structures at different scales in problems such as turbulent flow. For example, Farge and Schneider [9] describe a

nonlinear procedure to filter wavelet coefficients to separate the coherent vortices from the background flow. Their work shows that very few of the wavelet modes contain most of the enstrophy (the variable of interest), and selecting the top modes (2 or 3%) is sufficient to select the coherent structures. In a similar manner, in [36], Siegel and Weiss use wavelet packet technology, again in the context of fluid flow, to separate signals into coherent and non-coherent parts. They are particularly interested in a quantitative analysis of vortex behavior to complement the qualitative analysis obtained through visualization technology. Their experience shows that by keeping only a small subset of the wavelet coefficients they are able to extract individual coherent structures in turbulent flow. A slightly different approach is taken in [16], where Hangan et al. combine wavelets with the more traditional template matching approach from pattern recognition to match patterns at different scales.

1.3.2 Understanding simulations

Data mining methods can also be used to interpret results from simulations. For example, Mladenić et al. [24] describe the use of regression trees and a rule inducer to interpret results from simulations of a supermarket and a pub. The simulators take user-defined parameters as input (number of cashiers, arrival rates, etc.) and produce summary statistics (cashier utilization, length of customer's queues, etc.). The goal is to examine which input parameters affect the output of the simulators. The simulators are executed multiple times with different parameters and their outputs are recorded. Each execution produces a training example that is used as input to the tree or rule inducers. Both the regression tree algorithm and the rule inducer created accurate models that could be interpreted by humans.

In a similar way, Mladenić et al.[25] applied decision trees to a simulation of a steel production plant with the goal of identifying the reasons for excessive waste. Decision trees have also been used to analyze the massive datasets from simulations in [3].

1.3.3 Dimension reduction

Principal component analysis (PCA) is an important analysis tool used in many fields. In simulation data, it can be used to analyze the spatial or temporal variability of physical fields. The technique has been used extensively under the name of Empirical Orthogonal Functions (EOFs) in atmospheric sciences [32] and as Proper Orthogonal Decomposition (POD) in turbulence [19]. In particular, it has been used to build low dimensional models by Lumley and Blossey in [18], who exploit the fact that in certain problems, some parts of the domain are dominated by large-scale coherent structures. These can be resolved by a low dimensional model of the region, while parameterizing the smaller scale effects. This low dimensional model is generated by keeping a few eigen-functions obtained from the POD.

Non-linear extensions to Principal Components Analysis, based on auto-associative neural networks, have been studied by Sengupta and Boyle [29, 35]. They found that the non-linear method effected somewhat greater data reduction and that the leading

nonlinear mode captured the seasonal cycle of precipitation more clearly than the leading linear mode. They concluded that non-linear techniques should be considered especially when linear PCA fails to uncover physically meaningful patterns in climatological data analysis.

A different application of dimension reduction technique has been used in the Sapphire project to separate signals in climate data [5]. Since simulation data contains the effects of many different sources, such as El Niño and volcano signals, they must be removed in order to make meaningful comparisons between different climate models. This separation is made complicated by the fact that recent volcano eruptions coincided with El Niño events. Our experiences showed that by combining the traditional Principal Component Analysis with the newer Independent Component Analysis [17], we obtained better separation than by just using PCA alone [13].

1.3.4 Information retrieval in simulation data

Finding objects of interest in simulation data is a task that is essential to analyzing the data. The traditional solution to this problem has been the use of visualization techniques for "feature" extraction and tracking. In contrast to data mining, where the term "feature" is a descriptor of an object in the data, the visualization community uses the term to refer to the object itself. For example, in [20] Machiraju et al. describe the detection of swirl, which is a scalar quantity used to identify features such as vortices, which are are characterized by swirling flow. The swirl is a well defined mathematical quantity derived from the velocity and velocity gradients at each grid point. However, since simulation data is often stored in a compressed form using wavelets, it is important to be able to calculate the swirl feature directly from the compressed data. The authors in [20] illustrate how the velocity and velocity gradients can be approximated using data only in the compressed wavelet domain. They borrow ideas from multi-grid algorithms and finite-difference approximations to incorporate correction terms that account for the different spacing between the grid points at different levels in the wavelet representation of the compressed data.

A more recent development in the analysis of simulation data is the use of techniques from the Content-Based Image Retrieval (CBIR) and related communities. In CBIR, the task is to find images in a database that are similar to a query image. Since simulation data that has been processed for visualization, and mapped to a rectangular display, can be considered to be an image, CBIR techniques can be applied to this processed data.

CBIR systems extract various features or descriptors from the images and model similarity through the use of mathematical distance functions between the feature vectors. Extensive research has been performed to derive compact, representative features and distance functions to model visual cues such as color, texture, and shapes. Excellent reviews of different CBIR systems can be found in [4, 6, 37, 27, 30, 14, 7, 31]. All of these systems focus on photographic imagery, remotely sensed images, medical images or geological images. However, simulation data is different from such image data. First, unless the underlying grid is a simple Cartesian mesh, simulation data is not available at regular points in a rectangular domain. So, many

of the image processing algorithms cannot be directly applied. One solution is to sample the data at regular intervals, similar to the approach used when the data is converted to a regular rectangular grid for display in visualization. Second, there are often several physical quantities associated with each grid point, unlike an image, which is just a single quantity. Third, the "objects" in a simulation rarely correspond to physical objects such as a car or a person. In addition, the "objects" in an image often change shape as the simulation evolves, and may even appear or disappear with time. Finally, we observe that much of the CBIR work focuses on extracting features for, and retrieving, the entire image, while in simulation data, we are interested in parts of an image. This implies that the image must be segmented to identify these parts prior to the extraction of features.

Despite these differences, a couple of efforts have begun to use CBIR techniques in the context of simulation data. The Feature Extraction and Evaluation for Massive ASCI Data Sets (FEEMADS) project [10] is extracting features using the data from a hydro-dynamics simulation. The grid used is from an adaptive mesh refinement scheme (similar to Figure 1.3). The authors consider two scenarios: one, where the adaptive mesh data is resampled on a uniform grid so image processing techniques can be directly applied, and the other, where the data is directly obtained from the oct-tree data structure used to store the unstructured mesh. In the latter scenario, they exploit the fact that in the areas of interest, the mesh is refined, and the level of refinement can be used to limit the search. An approach more closely resembling the traditional CBIR systems is taken in the Sapphire project [34]. This work is described in more detail in Section 1.4.

1.3.5 Code validation

Another area where data mining techniques are being applied in simulation data is the area of code verification and validation. In particular, for code validation, high quality experiments are now becoming available, enabling effective comparisons of simulations to experiments. While such comparisons are in the preliminary stages [33, 38, 12], it is clear that techniques from data mining have a greater role to play in this area.

1.4 SIMILARITY-BASED OBJECT RETRIEVAL

Similarity-based object retrieval (SBOR) is a system being developed as part of the Sapphire project at Lawrence Livermore National Laboratory [34]. The original motivation for the system was to support searches in simulation data that are similar to those in CBIR systems. However, instead of focusing on the entire image as in CBIR, we were interested in retrieving sub-images, or objects, in the data. We have also applied this system to other image data such as remote-sensing imagery, and found other uses of the modules that form such a system. As an example, consider the code validation problem in Figure 1.4, where we need to determine how close a simulation is to an experiment. One approach to this would be to consider each

of the mushroom-shaped objects in the simulation and measure its similarity to the corresponding object in the experiment. This would require techniques similar to those used in CBIR. Having measured the similarity of these objects taken in isolation, we could then combine these measures with additional information such as the number and locations of the mushrooms to quantitatively compare the experimental image with the ones from the simulation.

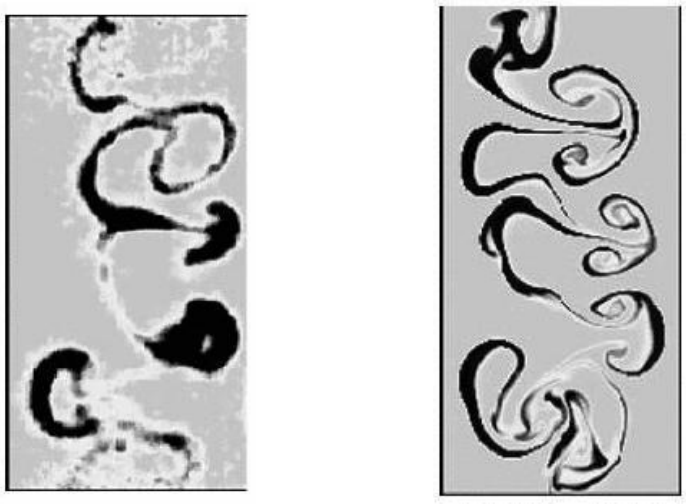


Fig. 1.4 The left image shows the flow pattern of a Richtmyer-Meshkov experiment performed at Los Alamos National Laboratory. The same experiment is simulated by high resolution numerical methods and the result is shown in the right image [33].

The SBOR system consists of five major functional modules: object identification, feature extraction, dimension reduction, similarity search, and relevance feedback. Figure 1.5 shows the flow of data in the SBOR system. We assume that the user has identified an object of interest in a query image, and is interested in finding other similar objects in the image database. Here, we use the term "database" in the context of a data store, rather than a true database. We also use the term "object" in a generic sense. The image database contains simulation data in the form of images; the query image is also part of this database. First, the images in the database are input to the object identification module. This module is responsible for extracting objects of interest within the images for subsequent analysis. These objects are not used directly, as a pixel-to-pixel comparison of objects is very sensitive to small changes in geometry and intensity. Instead, for each object in the image, we extract higher level features that are invariant under various conditions such as rotation, translation, and scaling. The feature extraction module supports several different features, enabling the user to extract the subset that best meets the goals of the similarity search. After the feature vectors are generated for each region in an image, we identify the most important subset using dimension reduction techniques. This

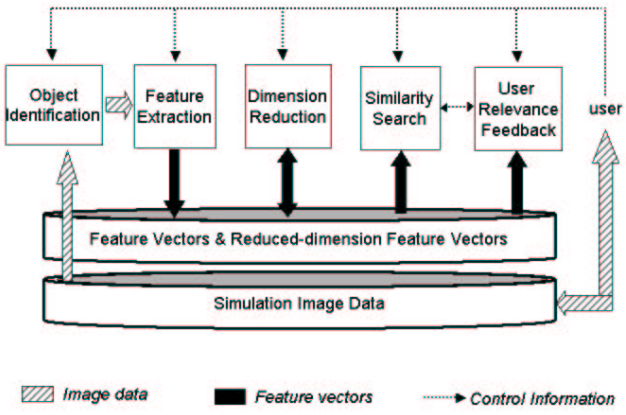


Fig. 1.5 Schematic diagram of the similarity-based object retrieval system.

can lead to a significant speedup in the similarity search, with little effect on the retrieval performance. Once we have the features (in the reduced dimension) for the query object and for all the objects in the image database, we use the similarity search module to find those objects which are "close-to" the query object. Different distance measures and search methods can be used to capture the notion of similarity. Further refinement is also possible through relevance feedback, where user feedback is used to improve the retrieval results.

It is important to realize that not all modules are used in every situation. During the initial stages of data exploration, users may want to experiment with a large number of different features. Sophisticated object identification techniques and dimension reduction may be unnecessary at this stage. Once the set of features is determined, the user can invoke more complex object identification schemes to capture objects of different scales, and apply dimension reduction to speed up the similarity search. We next describe each module of the SBOR system in further detail.

1.4.1 Object Identification

In the SBOR system, given a query object of interest in a simulation image, we are interested in finding similar objects in other images. Thus far, we have used the term "object" in a generic sense. In its simplest form, an object could be a tile, that is, a rectangular region in the image. Or, it could be more complex, and correspond to an object with a complex shape that clearly stands out from the background, such as the mushroom-shaped objects in Figure 1.4. We next compare and contrast the various options we have considered for the object identification module.

Tile-based versus object-based In the tile-based approach, an image is covered by tiles of a given size. The tiles can be overlapping or non-overlapping, and can be square or rectangular (assuming 2-dimensional simulation data). Overlapping tiles would require more storage as there are more of them covering each image. However, they are more likely to include objects that might be split among tiles. Since a tile is very easy to define - it just needs the coordinates of opposite corners - the task of object identification is relatively simple. But, there are several drawbacks to the tile approach. First, some of the tiles may not have any interesting information, for example, a tile with all pixels with approximately the same value. In this case, it may not make sense to process such tiles any further. Second, a tile may be much smaller than the object of interest, and therefore, a tile-based approach would completely miss the object. However, a multi-resolution approach, as described later in this section, can be very useful in identifying such objects. Third, a tile may contain more than one object, with only one of the objects satisfying the similarity criterion. Since the tile is treated as a whole, this object may be missed in the retrieval. An obvious solution to these problems is the object-based approach, where we have processed each image in the database to extract the objects in the image. Such a system might be more efficient than the tile-based approach as it only stores and computes feature vectors for individual objects. However, it requires segmentation of images, which is known to be a very difficult problem [15, ch. 14]. For simulation data such as the images shown in Figure 1.4, such a segmentation may not even exist as there is no precise definition of an object. In addition, even if we successfully segmented an object from an image, the retrieval results may be poor if the query object was part of an object rather than the whole.

In the SBOR system, we support both the tile-based and the object-based approaches. Object segmentation is done using a simple thresholding scheme; more sophisticated techniques will be added in the future.

Fixed-size versus variable-size tiles In the case of fixed-size tiles, the same tile size is used for all tiles in an image. This works well when the feature extracted describes a characteristic of the entire tile, such as texture. However, when we are interested in objects of a size larger, or smaller, than the tile size, the retrieval results may be poor. An object-based approach could provide a solution to this problem. Another alternative is to use variable sized tiles. The SBOR system allows the user to select different tile sizes through multi-resolution. A dyadic wavelet transform is first applied to each image in the dataset to create multiple images at different resolutions. A constant-size tile is then applied to each resolution, effectively creating tiles of variable sizes. Due to the reduction in image dimension, this approach is computationally more efficient than using large tiles on the original image. In addition, multi-resolution can also be applied to the query object, or a smaller tile used, enabling us to retrieve objects smaller than the query object.

Query-dependent versus query-independent tile size The tile size can be fixed prior to the use of the SBOR system, or determined when the query is made. If the tile size is dependent on the query, then the features must be extracted on-line, after the query has been made. As a quick response is often needed, this limits the number of images that can be queried. On the other hand, a query-independent tile size allows the features to be pre-computed, resulting in a faster turn-around.

Shape of the tile In addition to a rectangular or square shaped tile, a circular shape can also be used to ensure the resulting tiles are rotational invariant. However, care must be taken to ensure that the entire image is adequately covered by such tiles.

We note that these options can be used in combination. For example, a circular tile may be query dependent or query independent. Or, we can use the object-based approach in conjunction with the tile approach by initially segmenting the image and then considering the (complete or partial) objects that fall within each tile. Our experiences indicate that this might help in the identification of partial objects. Also, the multi-resolution approach can be applied to both the tiles and the objects.

1.4.2 Feature Extraction

This section describes the features provided in the SBOR system. We focus primarily on features that are scale, rotation, and translation invariant.

Simple Features This set of features consists of the mean, the standard deviation, the maximum, and the minimum of all pixel values in a tile image.

Histogram This is a 16-bin histogram of pixel values in a tile image. The bins are uniform across the dynamic range.

Simple Geometry The simple geometry feature contains the parameters of the ellipse that best represents the spatial distribution of the pixel values. It consists of five numbers: the location of the centroid, the lengths of the major and minor axes, and the angle between the major and the x axes.

ART The Angular Radial Transform (ART) belongs to a broad class of shape analysis tools based on moments [26]. Our implementation of ART is based on the region-shape descriptor defined in MPEG-7 [22, ch. 15]. ART projects a two-dimensional signal within the unit circle onto a set of complex orthonormal basis functions. The ART basis function of angular order n and radial order n in polar coordinates is given by

$$V_{nm}(\rho,\theta) = \begin{cases} \exp(jm\theta)/2 & n = 0 \\ \exp(jm\theta)\cos(\pi n\rho) & n \neq 0 \end{cases}$$

The ART coefficient F_{mn} of a 2-D signal $f(\rho, \theta)$ is defined by

$$F_{mn} = \frac{1}{\pi} \int_0^{2\pi} \int_0^1 V_{nm}^*(\rho, \theta) f(\rho, \theta) \rho \, d\rho \, d\theta \tag{1.1}$$

The ART feature is implemented by discretizing Equation (1.1). The origin of the functions is set at the centroid of the tile image. In order to cover the entire tile image, each basis function has a square-shaped support with each dimension equal to twice the longer side of the tile image. Rotational invariance is achieved by using only the magnitude of F_{mn}, and scale invariance is achieved by normalizing all F_{mn} by the area of the image, i.e. F_{00}. Following the MPEG-7 standard, twelve angular basis and three radial basis are used. This results in a 35-dimensional feature vector as the normalized F_{00} is always one and thus dropped from the representation. Unlike MPEG-7, we do not quantize the coefficients and retain the full floating-point precision for similarity search.

Binary Simple Geometry This feature is just the simple geometry feature applied to a tile in which the objects have been extracted by segmentation using simple thresholding. All object pixels are set equal to 255, and the background to 0.

Binary ART The Binary ART feature is just the ART feature applied to the tile in which the objects have been extracted by segmentation using simple thresholding.

Texture Features These features, applied at the tile level, capture the texture of the tile. The wavelet-based texture features are obtained by calculating the mean and standard deviation of the pixel values in each band at different levels of the wavelet decomposition [21]. The power spectrum features are statistical measures such as the average magnitude and variance of magnitude derived from the Fourier spectrum of the image [1].

1.4.3 Dimension Reduction

The large number of features extracted from the simulation data can pose problems during the retrieval. First, not all features may be relevant to the query of interest. Second, extracting, storing, and using the irrelevant features will take additional computational resources and slow down the retrieval time. Third, many of the efficient data structures that support fast retrieval are designed for 10-15 features. And, finally, if the number of features is very large, we encounter the "curse of dimensionality" problem associated with high dimensional spaces. To address this problem, we include dimension reduction techniques such as principal component analysis and independent component analysis in the SBOR system. Additional techniques, borrowed from the information retrieval community, will be added in the future.

1.4.4 Similarity Search

The similarity search module identifies the feature vectors in the database that are close to the query feature vector. We define similarity between two feature vectors by a distance function, where the similarity between two objects decreases as the distance between their feature vectors increases. We support some of the more commonly-used distance measures as defined in Table 1.1. Each feature can also normalized by subtracting the mean and dividing by the standard deviation to ensure that the dynamic range of each feature is the same.

Table 1.1 Table of different types of distance measures. $X = (x_1, \ldots, x_n)$ and $Y = (y_1, \ldots, y_n)$ are the two feature vectors. $\bar{X} = (\bar{x}_1, \ldots, \bar{x}_n)$ denotes the average of all feature vectors in the database. The range of summation or maximum is always from 1 to n.

Distance Measures	*Definitions*
Manhattan or l_1	$\sum \lvert x_i - y_i \rvert$
Euclidean or l_2	$\sum (x_i - y_i)^2$
Chebychev or l_∞	$\max \lvert x_i - y_i \rvert$
Cosine	$\arccos \left[\frac{\sum (x_i - \bar{x}_i)(y_i - \bar{x}_i)}{\sqrt{l_2(x,\bar{x}) \cdot l_2(y,\bar{x})}} \right]$
Kullback-Leibler Divergence	$\sum x_i \log(x_i / y_i)$
Chi-Square	$\sum \frac{x_i^2 - y_i^2}{y_i}$

We currently support two types of search functions: ϵ-search and k-Nearest-Neighbor (k-NN) search. In the k-NN search, the k feature vectors in the database closest to the query feature vector are returned. A small k in a k-NN search allows a user to quickly examine the small set of returned results. In an ϵ-search, the module returns all feature vectors in the database whose distance from the query feature is within a positive threshold ϵ. ϵ-search is intended for experienced users who can correlate distance values with the level of similarity.

1.4.5 Relevance Feedback

The SBOR system also includes options for relevance feedback. Relevance feedback is the process by which user input is included in the retrieval by having the user provide feedback on the relevance or irrelevance of the current retrieval results. The simplest form of relevance feedback either changes the weights on the features being used or modifies the query itself to be closer to the user input of relevant results and farther from the irrelevant results. An alternative method to incorporate the user's feedback is to use machine learning techniques. To apply these techniques, the user

creates a training set with positive and negative examples by labeling the results of the similarity search. The training set is used to train a classifier, which is applied to the entire database of feature vectors. The classifier assigns positive and negative labels to each feature vector in the database, and the positive examples are returned to the user. If necessary, the user can re-label the results and iterate the feedback process.

1.4.6 Sample Results

In this section, we illustrate the use of the SBOR system using the data from a high resolution simulation of a 3-dimensional shock tube. Initially, two fluids of different densities are separated by a membrane in the tube; then the membrane is pushed against a wire mesh. The simulation uses a regular 3-dimensional grid (described in Section 1.2) to model the resulting mixing of the two fluids [23]. The data generated for visualization consists of the the entropy variable at each grid point. Working with 256×256 sized slices of this data, we demonstrate how we can use the SBOR system to find mushroom-shaped objects in the data. As an illustration, we use a tile-based method for object identification, with the tile size determined by the query size. The image database is generated using the three images shown in Figure 1.6

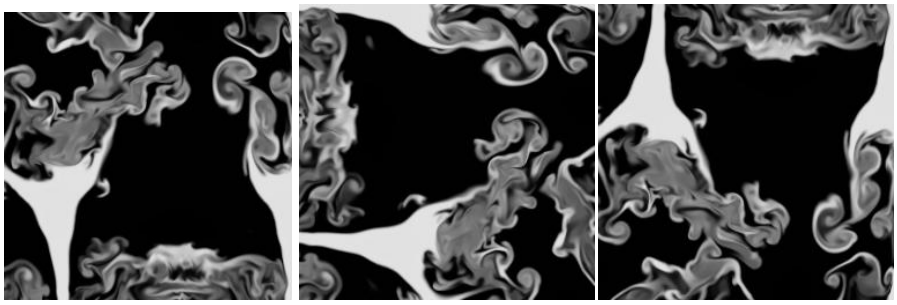
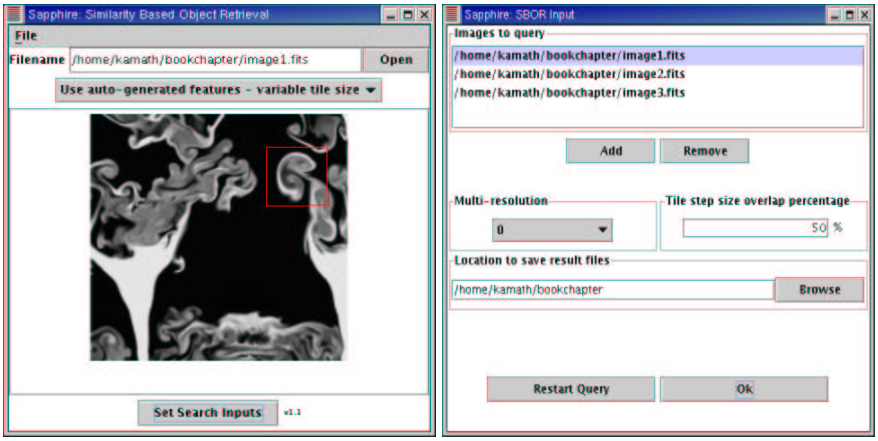


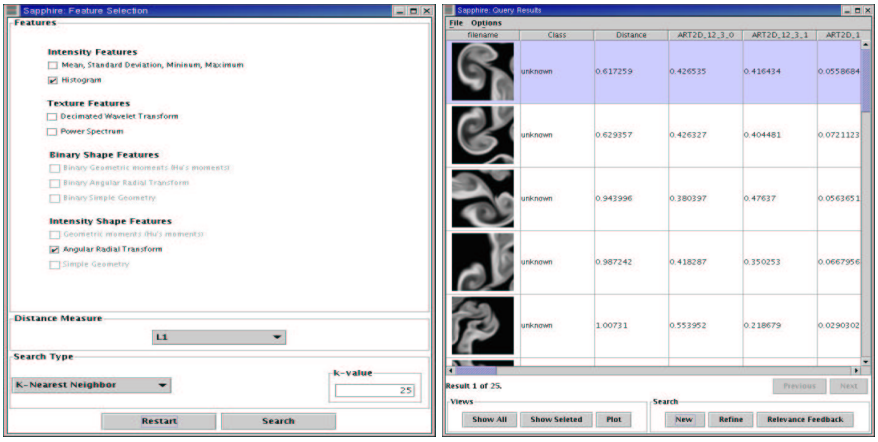
Fig. 1.6 Three sample images used to illustrate the SBOR system. These are 256 by 256 size slices through a three-dimensional data. They are taken at slightly different times from different locations in the grid. They may appear similar, though they are different in intensities. Each has a mushroom-shaped "object" at different orientations.

The user interacts with the SBOR system through a graphical menu driven system. The user first brings up the image of interest and identifies the area that forms the query object as in Figure 1.7, panel (a). When the user clicks the search button, it brings up the window that enables the choice of files to search on, the overlapping factor for the tiles, and the directory to be used for saving the results, as shown in Figure 1.7, panel (b). Next, in Figure 1.8 panel (a), the user selects the features to use in the similarity search, the distance metric, and the type of search, which results in the output shown in Figure 1.8 panel (b). Using the histogram and ART features, with the l_1 norm, we see that all the three mushroom-shaped objects are returned as



(a) (b)

Fig. 1.7 (a) Query selection on an image of interest. (b) Options for the tile-based approach.



(a) (b)

Fig. 1.8 (a) Choosing the features of interest and options for the search. (b) Results returned in decreasing order of similarity.

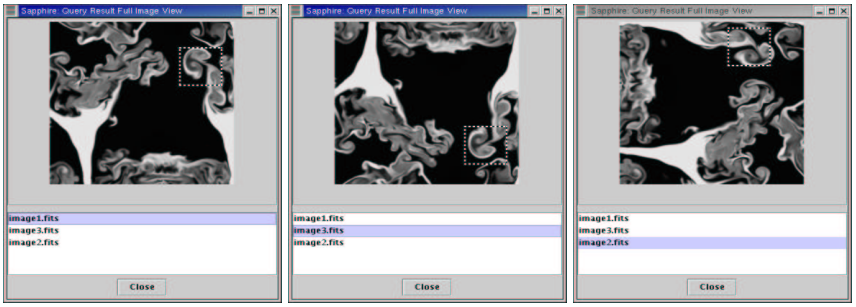


Fig. 1.9 The top three results returned are outlined on the respective images.

being very similar to the query object. Figure 1.9 shows the top three results on the corresponding images.

1.5 CHALLENGES AND OPPORTUNITIES IN MINING SIMULATION DATA

The application of data mining to scientific simulation data is a relatively new field. As a result, there are several challenges and opportunities that await a data mining practitioner analyzing such data. Much of our work thus far has been with data on a regularly spaced grid. While this data is similar to an image, it is not possible to directly use image processing software as simulation data is mainly in floating point format, while image data is often in integer or byte format. Thus, we found it necessary to rewrite the image processing techniques for floating point data. In addition, in computer simulations, several different variables are generated at each mesh point. A straightforward approach to this might be to apply techniques from the data fusion community, where data from different sensors are combined to exploit the complementary information from each sensor.

A different problem arises when the data is available on an unstructured mesh, where we cannot directly apply the techniques from traditional image processing. As mentioned earlier, we could sample the data at regular points and treat it as data on a regular grid. Alternatively, we could locally use variable-sized tiles to process a mesh such as the one in Figure 1.3. Also, we could use ideas from the PDE-based image processing community to directly extract the features of interest. The issue gets more complex when the data is available in the compressed form, and the size of the data is such that it is preferable to operate on it without uncompressing it. In Section 1.3, we have reviewed some approaches being used to process the data in the compressed form.

In our experiences, we found that simulation data shares some problems common to other scientific data sets, such as the need to work with spatio-temporal data, massive sizes of the data, the lack of sufficient training examples, the need to mine data as it is being generated, and the need for real time turnaround. In addition, as in the case of image data, we found that the initial preprocessing of the data is very time consuming and difficult.

1.6 SUMMARY

In this chapter, we have summarized our experiences on mining a relatively new type of data, that is, data from computer simulations. We have surveyed the work done in this field, and discussed our experiences with the similarity-based object retrieval system which is being used to mine such data. Finally, we briefly discussed the issues that are specific to simulation data, and potential solutions that are being considered in the solution of these problems.

Acknowledgments

This work was performed under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48.

REFERENCES

1. M. F. Augusteijn, L. E. Clemens, and K. A. Shaw. Performance evaluation of texture measures for ground cover identification in satellite images by means of a neural network classifier. *IEEE Transactions on Geoscience and Remote Sensing*, 33(3):616–626, 1995.

2. A. Banerjee, H. Hirsh, and T. Ellman. Inductive learning of feature tracking rules for scientific visualization. In *Proceedings of the IJCAI-95 Workshop on Machine Learning in Engineering*, 1995.

3. K. Bowyer, L. Hall, N. Chawla, T. Moore, and W. Kegelmeyer. A parallel decision tree builder for mining very large visualization datasets. In *IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2000.

4. V. Castelli and D. Bergman, editors. *Image Databases: Search and Retrieval of Digital Imagery*. John Wiley & Sons, Inc., 2002.

5. Separation of climate signals, Sapphire Web page, 2003. http://www.llnl.gov/-casc/sapphire/sep_climate/index.html.

6. C. Djeraba et al. Special issue on content-based multimedia indexing and retrieval. *IEEE Multimedia Magazine*, 9(2):18–60, 2002.

7. C. Faloutsos. *Searching Multimedia Databases by Content*. Kluwer Academic Publishers, 1996.

8. M. Farge, N. Kevlahan, V. Perrier, and E. Goirand. Wavelet and turbulence. *Proceedings of the IEEE*, 84(4):639–669, 1996.

9. M. Farge and K. Schneider. Analyzing and compressing turbulent fields with wavelets. Technical report, Institute Pierre Simon de Laplace, June 2002. http://monteverdi.ens.fr/.

10. Feature Extraction and Evaluation for Massive ASCI Data Sets, FEEMADS Web page, 2003. http://www.c3.lanl.gov/feemads/.

11. J. Ferre-Gine, R. Rallo, A. Arenas, and F. Giralt. Identification of coherent structures in turbulent shear flows with a fuzzy artmap neural network. *International Journal of Neural Systems*, 7(5):559–568, 1996.

12. B. Fishbine. Code validation experiments. Technical Report Fall 2002, Los Alamos National Laboratory, 2002. http://www.lanl.gov/quarterly/q_fall02/.

13. I.K. Fodor and C.Kamath. Using independent component analysis to separate signals in climate data. In *Proceedings, Independent Component Analyses, Wavelets, and Neural Networks, number 5102 in SPIE Proceedings*, pages 25–36. SPIE, 2003.

14. D. Forsyth, J. Malik, and R. Wilensky. Searching for digital pictures. *Scientific American*, pages 88–93, June 1997.

15. D.A. Forsyth and J. Ponce. *Computer Vision A Modern Approach*. Prentice Hall, New Jersey, U.S., 2003.

16. H. Hangan, G. A. Kopp, A. Vernet, and R. Martinuzzi. A wavelet pattern recognition technique for identifying flow structures in cylinder generated wakes. *Journal of Wind Engineering and Industrial Aerodynamics*, 89:1001–1015, 2001.

17. A. Hyvärinen, J. Karhunen, and E. Oja. *Independent Component Analysis*. Wiley, 2001. Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control.

18. J. Lumley and P. Blossey. Control of turbulence. *Annu Rev. Fluid Mech.*, 30:311–327, 1998.

19. J. L. Lumley. The structure of inhomogeneous turbulent flows. In A.M. Yaglom and V.I. Tatarski, editors, *Atmospheric turbulence and radio propagation*, pages 166–178. 1967.

20. R. Machiraju, J. E. Fowler, D. Thompson, B. Soni, and W. Schroeder. EVITA - Efficient Visualization and Interrogation of Tera-Scale Data. In R. L. Grossman, C. Kamath, P. Kegelmeyer, V. Kumar, and R. R. Namburu, editors, *Data Mining*

for Scientific and Engineering Applications, pages 257–279. Kluwer Academic Publishers, 2001.

21. B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, 1996.

22. B.S. Manjunath, P. Salembier, and T. Sikora, editors. *Introduction to MPEG-7: Multimedia Content Description Interface*. John Wiley & Sons, Ltd., 2002.

23. A. Mirin et al. Very high resolution simulation of compressible turbulence on the IBM-SP system. Technical Report UCRL-JC-134237, Lawrence Livermore National Laboratory, 1999.

24. D. Mladenić, I. Bratko, R.J. Paul, and M. Grobelnik. Using machine learning techniques to interpret results from discrete event simulation. In *Proceedings of the Sixth European Conference on Machine Learning ECML94*. Springer-Verlag, 1994.

25. D. Mladenic, M. Grobelnik, I. Bratko, and R.J. Paul. Classification tree chaining in data analysis. In *IJCAI Workshop on Data Engineering for Inductive Learning*, Montreal, 1995.

26. R. Mukundan and K. R. Ramakrishnan. *Moment Functions In Image Analysis: Theory and Applications*. World Scientific, 1988.

27. M.Yeung et al. Special section on storage, processing, and retrieval of digital media. *Journal of Electronic Imaging*, 10(4), October 2001.

28. National Centers for Environmental Predictions Web page, 2003. http://www.-ncep.noaa.gov/.

29. Principal-Component Statistical Studies, PCMDI Web page, 2003. http://www-pcmdi.llnl.gov/pcmdi/statistics.html.

30. B. Perry et al. *Content-based access to multimedia information – from technology trends to state of the art*, chapter 4.3. Kluwer Academic Publishers, Massachusetts, U.S.A., 1999.

31. R.W. Picard, A.P. Pentland, et al. Special issue on digital libraries. *IEEE Transactions on Pattern Analysis and Machine intelligence*, 18(8), August 1996.

32. R. W. Preisendorfer and C. D. Mobley. *Principal Component Analysis in Meteorology and Oceanography*. Elsevier Science, 1988.

33. W. Rider et al. Using Richtmyer-Meshkov driven mixing experiments to impact the development of numerical methods for compressible hydrodynamics. In *Proceedings of the Ninth International Conference on Hyperbolic Problems Theory, Numerics, Applications*, pages 84 – 88, 2002. http://www.acm.caltech.edu/hyp-2002/program.html.

34. Similarity-Based Object Retrieval, SBOR Web page, 2003. http://www.llnl.gov/-casc/sapphire/sbor.html.

35. S. K. Sengupta and J. S. Boyle. Nonlinear principal component analysis of climate data. Technical Report PCMDI Report No. 29,, Program for Climate Model Diagnosis and Intercomparison (PCMDI), 1995.

36. A Siegel and J. B. Weiss. A wavelet-packet census algorithm for calculating vortex statistics. *Phys. Fluids*, 9(7):1988–1999, 1997.

37. R. C. Veltkamp, H. Burkhardt, and H.-P. Kriegel, editors. *State-of-the-Art in Content-Based Image and Video Retrieval*. Kluwer Academic publishers, 2001.

38. C. A. Zoldi. *A numerical and experimental study of a shock-accelerated heavy gas cylinder*. PhD thesis, State University of New York at Stony Brook, 2002.