Visualizing and Investigating Multidimensional Functions

S. R. dos Santos and K. W. Brodlie

School of Computing, University of Leeds, Leeds, UK

Abstract

This paper addresses the problem of visualizing multidimensional scalar functions. These functions are often encountered in fields such as Engineering, Mathematics, and Physics to understand and model complex phenomena. We propose a novel method based on the dimension reduction philosophy called HyperCell. Its basic concept is to represent the function by means of dynamic orthogonal low (1D, 2D, or 3D) dimensional subspaces, called Cells. Firstly the user defines a N-dimensional region of interest, in which the data can be visualized. Then the user interactively creates cells by selecting up to three dimensions from the function domain. A cell can be visualized using a standard visualization algorithm such as isosurfacing or volume rendering. The analysis of the function is done by investigating several cells, which can be sampled simultaneously at different regions of interest in N-Space. The HyperCell method allows several useful operations to help the exploratory process such as navigation and rotation in N-Space, and brushing.

Categories and Subject Description: I.3.2 Graphics Systems; I.3.6 Methodology and Techniques; I.3.3 Picture/Image Generation.

1. Introduction

A large number of problems and processes in Engineering, Mathematics, and Physics are frequently modelled by *scalar functions of many variables*, also known as *multidimensional functions*. These functions can be found in problems such as the sensitivity analysis for linear and non-linear programming involving multiple parameters, the mathematical exploration of the geometry of objects in high-dimensional space, and the complex modelling of physical phenomena, to name just a few.

Scientific Visualization can greatly enhance the analytical process by offering a suitable mapping from the problem to the visual domain in such a way to provide insight about the object under investigation.

In this paper we have addressed the problem of visualizing *multidimensional functions*, by introducing a novel visualization method based on the dimension reduction philosophy. We are interested in functions defined in Euclidean space of dimension equal to or greater than four, which we call *multidimensional space*. They can be mathematically represented by either an analytical formula or interpolation of data values obtained from a complex simulation. They are abstractly represented by $f(x_1, x_2, ... , x_N) = y$, where each x_i represents the *i-th* dimension, and *y* is a scalar value.

1.1 Previous Work

In recent years there have been several attempts to deal with the visualization of multidimensional data.

In 1987 Inselberg and Dimsdale[19] introduced the *parallel co-ordinates* method. This technique is designed to represent N-dimensional points, hence supporting both the analysis of multivariate data and the visualization of the geometry of multidimensional objects[30]. In this technique vertical parallel axes are used to represent the dimensions and an N-dimensional point is plotted as a polyline that intersects all N axes. Each polyline intersection with the axes corresponds to the point's value for that dimension (axis).

Parallel co-ordinates have the advantage that all dimensions are treated equally. In addition, there are interesting reciprocal dualities between Cartesian and parallel coordinates. Points in the former are mapped to lines in the latter, rotations become translations, and inflection points become cusps.

One disadvantage of parallel co-ordinates is that there is no pre-defined order in which to arrange the dimensions. Hence relationships among dimensions could be obscured by an inappropriate arrangement of dimensions.

In 1990 Feiner and Beshers[13] conceived the *Worlds within Worlds* technique. It is based on dimension stacking, with at most three variables represented at each level, creating an interactive hierarchy of displays. The user interacts with the system using a data glove to define position in the space in which a new three-axis "world" is created to accommodate three more variables. The process is then repeated until all dimensions have been mapped, ending up with a surface corresponding to the last variable defined. Of course, different mapping of the variables yields different views of the data. A strong point of this technique is the direct manipulation of the graph by means of a haptic device such as the data glove.

Similar work, following the basic principle of nesting dimensions, was presented by Mihalisin *et al.*[25] in 1991. In this

case the objective was accomplished by plotting the dependent variable on hierarchical axis corresponding to the independent data. One disadvantage of this method is that the higher the function dimensionality the more cluttered is the display. In addition, the process of assign colours to dimensions is arbitrary without perceptual considerations.

Bajaj *et al.*[3] introduced a more recent approach called Hypervolume in 1998. It creates a projection of scalar fields of any dimension into a single image that corresponds to a global view of the data and would be 'fully explanatory'. They have applied a generalisation of direct parallel projection methods from volume rendering. Their method depends heavily on a user interface that provides real time control over n-dimensional rotation and projection parameters. They also make use of tree hierarchical representation in order to cope with the huge amount of data involved in high dimensional problems. Therefore their storage scheme allows the user to control the balance between accuracy and speed so as to achieve a real time performance.

In 1993 van Wijk and van Liere[29,28], introduced the HyperSlice technique. It is a method for visualization of scalar functions of many variables. The central concept is the representation of a multidimensional function as a matrix of orthogonal two-dimensional slices. Each slice is a function subspace obtained by fixing the value of a number of variables so that there are either two or one free variables.

Its strongest point is the direct relation between screen space and data space. This relation made possible the design of a user interface that affords direct manipulation of operations such as navigation, identification of extrema, and definition of paths in hyperspace.

Although the authors decided to visually represent the function by two–dimensional slices, they recognised that three-dimensional techniques seem to be the most natural choice for the basic representation of the multidimensional function. In their opinion the 3D representation has the advantage of encoding as many dimensions as possible simultaneously.

Despite this, the authors decided not to use 3D as the basic representation for the HyperSlice for three reasons: (1) at that time the techniques for volume rendering were too slow for direct manipulation, (2) difficulty in interpreting a 3D representation in comparison with simpler representation forms, and (3) interaction in 3D is not trivial.

Although the HyperSlice method has proved useful, we have identified three shortcomings in the method:

- Sometimes a two-dimensional representation is not enough to promptly identify special features in a function. Hence it is necessary to investigate several 2D slices or even perform navigation in the hyperspace to mentally "visualize" such a feature, whereas with a 3D representation the same feature (e.g. a saddle point) could have been visualized in a single subspace.

- Because HyperSlice represents the multidimensional function as a matrix of orthogonal two-dimensional slices simultaneously, all the data corresponding to those subsets must be present before drawing. Thus the

method is constrained by both the dimensionality of the function and the number of samples necessary to plot each slice.

- It provides only one focal point. Hence the only way of comparing different regions in the hyperspace is moving the focal point through a user defined path, which inevitably will lead, as a side effect, to the loss of the visualization of the initial reference point.

Therefore, in this paper we put forward a method that, in a sense, could be thought of as the three-dimensional extension of the HyperSlice idea. We have based our approach on the projection-based philosophy, in which the multidimensional data is projected down to lower subspaces.

The main motivations for this work are:

- To test the hypothesis that providing the users with a dynamic three-dimensional representation of the subspaces is better than a static 2D one;

- We believe that, several years later, the three reasons given by the HyperSlice's authors not to use three-dimensional representation may be challenged, and;

- To overcome the three identified HyperSlice's shortcomings.

In our approach the user interface also plays an important role. It manages the interactive creation of dynamic subspaces, called **Cells**, affords the hyperspace navigation, and controls interactive operations over the cells or group of cells. Our technique, called **HyperCell**, works well for the case where each independent variable is sampled in a regular grid or lattice-like fashion. However, it is flexible enough to be generalised to a variety of less restrictive domains.

The paper is organised as follows. Section 2 describes the key ideas behind the HyperCell method. Section 3 presents some extensions to the HyperCell such as brushing, rotation in N-space, and cell splitting. Section 4 discusses some conceptual issues related to our proposed approach. Section 5 deals with implementation details and contains the results from the first experiment with the method. Section 6 contains the analysis of the results. Finally Section 7 sums up our findings and lists some issues we need to tackle shortly.

2. Description of the HyperCell Method

We describe in this section how the HyperCell visually represents N-dimensional functions by dynamic three-dimensional boxes, called cells.

A cell is a *low dimensional subspace* of the function image. They can be composed of one, two, or three dimensions selected from the original N-dimensional function domain. Each cell contains data from the evaluation of the function within the boundaries of the low dimensional subspace it defines. Then the data is visualized inside the cell through a standard visualization technique. By standard visualization we mean any visual algorithm that can be applied to visualize a three-dimensional data set, such as isosurfacing (to highlight a specific function value) or volume rendering (to represent the whole function image domain contained in the subspace).

To comprehend the function as a whole, one needs to investigate several individual *dynamic* cells to build up a mental image of it. The cell entity is dynamic because it can change, in real time, both (a) the dimensional subspace from 1D up to 3D and vice versa, and (b) the visual algorithm employed to visualize its data.

We argue that the visualization of the multidimensional function through several three-dimensional dynamic subspaces reduces the overall complexity of the problem, since the human mind is not trained to create a mental image of complex functions defined in N-space (N ≥ 4).

2.1 Interaction Graph Interface

Because HyperCell visually represents the N-dimensional function through three-dimensional subspaces, there are $^nC_3 = (N^3-3N^2+2N)/6$ possible combinations of distinct dimensions to form a cell. This leads us to a secondary problem: How could one manage all the possibilities without being lost in such hyperspace?

Presenting all the possible cells simultaneously for a high number of dimensions is somehow impracticable, given the limited screen space and the large number of 3D cells that may be created in this process. To solve this problem we let the user manage the instantiation of cells. Thus one creates in real time as many cells as necessary to build up a cognitive map of the N-dimensional function.

This alternative has two direct benefits. The first one is that it eliminates the need of having all the data available before the visualization process takes place, delaying it to the moment when the user selects a cell. This, in turn, makes it possible, for example, to integrate HyperCell with a function evaluation module to support progressive refinement (by evaluating more points on the grid) and possibly computational steering.

To deal with such a problem, we have developed a user interface that: (a) performs the process of instantiating cells, (b) manages the number of possible cells that can be created to investigate the data, (c) informs the user about the available combination of dimensions to form a new cell, (d) supports interactive n-dimensional operations, and (e) helps the user navigate throughout the hyperspace.

This entity is called an *interaction graph*. It is a 2D full connected graph with as many vertices as the function dimensionality. Each edge is regarded as a two-flat, which means a plane defined by any two co-ordinate axes in a high-dimensional space. Each triangle on the graph corresponds to an unique cell composed of the vertices (dimensions) of the sub-graph (triangle). Figure 1 shows this concept applied to the representation of five dimensions.

Each edge can be used **n-2** times to compose a new cell. Consider, for example, the example in Figure 1. If we take the **x1x2** edge, it can be combined with any of the remaining nodes **x3**, **x4**, or **x5**, **{(x1,x2), x3, x4, x5} ⇒ (x1x2x3), (x1x2x4)**, and **(x1x2x5)**.
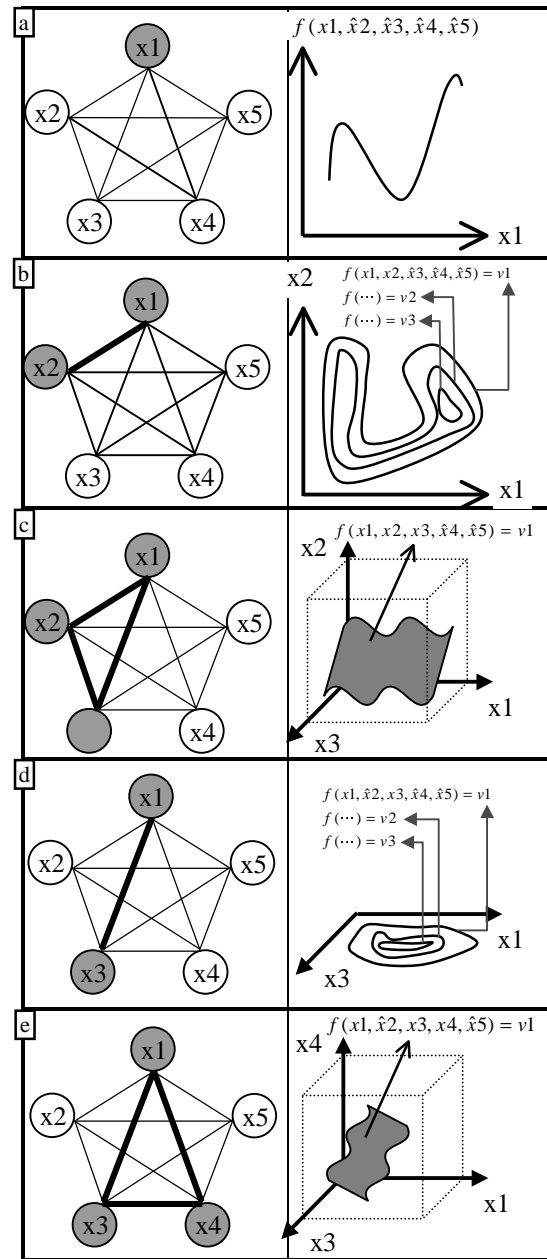

Figure 1*: Progressive selection of subspaces with different dimensionalities using the Interaction Graph (N=5).*

In order to keep track of all cells that might be created, the interaction graph shows, if required, the remaining number of times that each edge could be used to create a new cell. This number is shown on the corresponding edge.

Once a cell is created it is possible to perform direct manipulation such as dragging it around or rotating it in the 3D visual space, in a track-ball fashion.

Hopefully an example will illustrate the interplay between the *interaction graph* and the creation of the *3D dynamic*

cells. Let us consider a 5D function visualized as isovalues and the creation process of one cell, as depicted in Figure 1.

When the user clicks on the first dimension (say, **x1**) automatically *HyperCell* generates a line graph of the function $f(x1, \hat{x}2, \hat{x}3, \hat{x}4, \hat{x}5)$ (Figure 1-a). Next, the user selects another dimension (**x2**), which will generate a transformation of the previous 1D cell into a 2D isoline graph of $f(x1, x2, \hat{x}3, \hat{x}4, \hat{x}5)$ (Figure 1-b). Then the user selects the third dimension (**x3**), which will create the 3D cell containing an isosurface of $f(x1, x2, x3, \hat{x}4, \hat{x}5)$ (Figure 1-c), corresponding to a given function value (chosen by the user). Alternatively, the user could unselect the **x2** vertex (thus generating an isoline graph of **x1** versus **x3**, Figure 1-d) and select **x4**, creating a new cell with the isosurface of the function $f(x1, \hat{x}2, x3, x4, \hat{x}5)$ (Figure 1-e).

2.2 Region of Interest

Before creating the cells, the user needs to define one or more N-dimensional regions of interest within the N-space. We believe that multiple regions of interest give the user more flexibility in the investigative process. One could create, say, the first three cells based on a specific region of interest and store them in a given workspace, then create a further three cells around a different region of interest in another workspace.

In our initial approach the *region of interest* (RI) follows the same definition as the current point and associated region in HyperSlice. It is defined by a *focal point* $F = (f_1, f_2, ..., f_N)$, and a width defined by a set of scalar values $S = \{s_1, s_2, ..., s_N\}$. Hence the *region of interest* for each dimension i is denoted by $RI_i = [f_i - s_i/2, f_i + s_i/2]$. Only data within the N-dimensional region defined by the *region of interest* are visualized (see Figure 2).
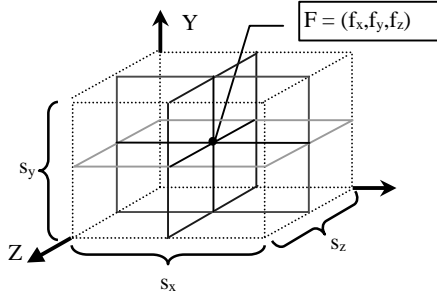

Figure 2: *The concept of region of interest (the 3D dotted box) applied to a 3-dimensional space.*

There are three different ways of modifying the region of interest to explore distinct regions in hyperspace. The first one is to shift the location of the focal point F. This task permits the definition of paths in hyperspace. The second one is achieved by rotating the *region of interest* around the focal point F. This allows the explorer to sweep around it. The third one is to change any scalar value s_i of S.

The *region of interest* plays an important role in the exploratory process. It represents a "vision" of the high-dimensional function and a logical link between subspaces.

It also can be applied to help tracking features present in one subspace throughout other subspaces.

We could outline an interesting parallel involving the cells and the *region of interest*. The cells can be thought of as "mini-universes" which, in turn, are part of a "greater-universe" (data set). Thus the *region of interests*, specifically the focal point F, assume the role of portals through which the explorer (user) can travel from one "mini-universe" to others.

Other operations involving the *region of interest* are discussed in the next section.

3. Extending the HyperCell Method

3.1 Cell Organisation

The use of 3D cells has the side effect of rapidly cluttering the screen, which leads us to the second aspect of the user interface. We have employed the concept of multiple three-dimensional workspaces[7]. This simple mechanism enables the user to logically group related cells together and store them in these workspaces. Then, it is possible to switch back and forth between the workspaces as necessary.

These workspaces are represented as leaves in a tree-like representation, as shown in Figure 3. This approach grants to HyperCell dimensional scalability (c.f. Section 6), since the visual representation does not restrict the number of dimensions that can be visualized on the screen. Nevertheless, the limit might still exist but it would only be imposed by the user's cognitive skill.
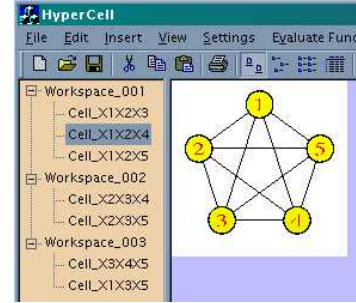

Figure 3: *Current implementation of workspaces through a tree control. Three workspaces have been created.*

3.2 Exploring the N-Space

There are four helpful operations to aid the exploratory process of the multidimensional space. They are: switching views, navigating, sweeping through the hyperspace, and scaling the *region of interest*.

Switching views is achieved simply by changing the focus from one cell to another. It helps in finding features across different dimensions. This procedure is accomplished simply by clicking on any cell displayed in the tree control window that represents the hierarchy of workspaces, as shown in Figure 3.

Navigating can be carried out by translating the *focal point*. This operation takes place in the visual space, hence one interacts directly on the cell's visual representation. The

user indicates the new position within the cell's visual subspace by (a) indicating the numeric visual co-ordinates in 3D-space (*linguistic approach*[14]) or (b) picking a position in 3D-space according to the *3D cursor*, our visual feedback mechanism (*visual approach*[14]).

The *3D cursor* is defined by two semitransparent orthogonal planes called *feedback planes*. The Z_{cur} and Y_{cur} planes are orthogonal to the *Z* and *Y* axes, respectively. The current position in 3D space is represented by a triple intersection between (1) the horizontal corresponding to the intersection of the two feedback planes, (2) a vertical line on the Z_{cur} plane, and (3) a horizontal line on the Y_{cur} perpendicular to the Z_{cur} plane, as shown in Figure 4.

The feedback planes are activated in turns. Mouse movements are mapped to movements on the active feedback plane, which changes the current position on that plane. By clicking the mouse's right most button the user switches between the planes. Hence this mechanism allows 3 degrees of freedom (left, right, up, down, forward, backward). By clicking the mouse's left most button the current 3D point is selected.
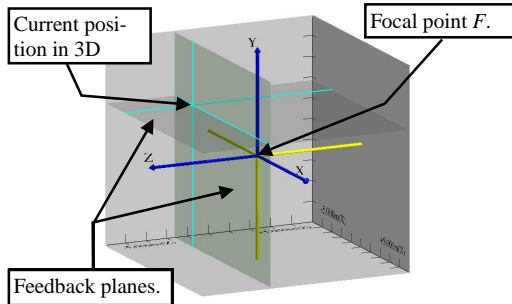

Figure 4: *Choosing a new position through the 3D cursor.*

Sweeping through the hyperspace means to perform a N-dimensional rotation around the *focal point*. Note that this rotation takes place in the N-space rather than in the visual space. In the N-space there are *N(N-1)/2* distinct pairs of co-ordinate axes that can be used as the plane of rotation.

A matrix $R_{ij}(\theta_{ij})$ that implements a rotation by an angle θ_{ij} in the plane defined by the pair of axes *(x_i,x_j)*, *i,j = 1,...,N*, has the following rule of formation: (1) all the elements ($e_{l,k}$ *,l=k*) from the main diagonal are equal to 1, except the $e_{i,i}$ and $e_{j,j}$, which are equal to *cos(θ_{ij})*, and (2) all the remaining elements ($e_{l,k}$ *,l≠k*) are equal to zero, except $e_{i,j}$ = *-$e_{j,i}$ = -sin(θ_{ij})*.

A rotation in 4D, for example, can be implemented by the following matrix concatenation

$$M=R_{12}(\theta_{12})R_{13}(\theta_{13})\ R_{14}(\theta_{14})\ R_{23}(\theta_{23})\ R_{24}(\theta_{24})\ R_{34}(\theta_{34})$$

The *N(N-1)/2* degrees of freedom are parameterised by *{θ_{ij}}*[16]. This rotation is triggered by direct manipulation of the cell through a mechanism called *control triangle* (see Figure 5) using, say, a mouse-like device. Each cell allows three different rotation planes.
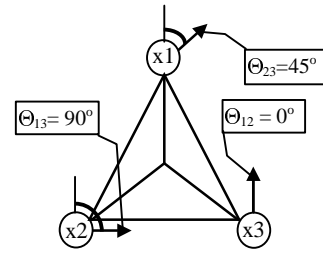

Figure 5: *Control triangle showing the rotation angles for a cell **x1x2x3**.*

Scale means to **expand** or contract the subspace defined by a cell. This can be done by either providing new values scalar values for the set *S* or by interacting directly on the cell's *control triangle*.

3.3 Brushing Operation

Brushing consists in defining an N-dimensional region of interest on the data. The data is selected in a given subspace (cell) and this action is reflected on all other subspaces.

Brushing in HyperCell is accomplished by (a) highlighting the brush region in all the related cells using, say, different colour scheme, (b) showing only the brush region, or (c) showing all the data except the brush region. All these operations are useful, for example, if we consider the exploration of *extrema*. By selecting the region of interest in on cell we can follow this feature in the others.

The selection and manipulation of the brushing region is done inside a *Brushing Graph* (a special version of the *Interaction Graph*), as seen in Figure 6.
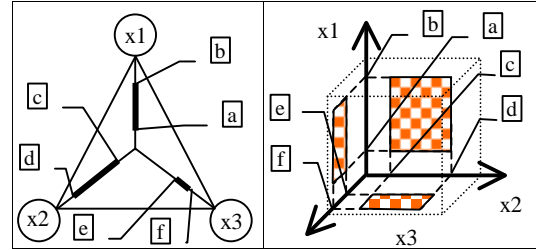

Figure 6: *Correspondence between Brushing Graph and cell for N=3.*

3.4 Cell Splitting

Cell splitting consists in putting several subspaces together to seek common features. The user selects a cell and the corresponding vertices on the *Interaction Graph* are highlighted. Then, the user selects one position inside the 3D space of the cell and an orthogonal plane (X, Y, or Z) that passes through the selected point. This is done through a slightly modified version of the *3D cursor*. The chosen plane is the *splitting plane*.

Now the user adds an extra dimension, dividing the cell into two different subspaces, as shown in Figure 7. Note that the entire splitting plane shares the same function values between the newly created subspaces. The goal in this op-

eration is to observe the transition from one subspace to another, looking, perhaps, for continuity in N-Space.
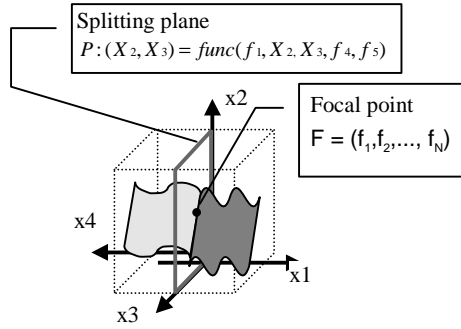

Figure 7: *Splitting the cell x1x2x3to accommodate the new added dimension x4 in a schematic visualization of a 5D function. The modified region is on the left hand side of the cell.*

4. Conceptual Discussion

In this section we put into perspective the research problem addressed in this paper, followed by the conceptual issues we have considered during the design of our suggested solution to it.

4.1 The Research Problem

To quote Earnshaw[4]:

> The goal of scientific visualization is to promote a deeper level of understanding of the data under investigation and to foster new insight into the underlying process, relying on the human's powerful ability to analyse.

This sentence stresses one of the most important objectives of a visualization, which is acquiring insight about the data. Hence we could summarise our research challenge in one sentence: *How could one **visualize** a high-dimensional function so that one would be able to acquire **meaningful insight** about it?*

The term *meaningful insight* in this context leads us straight to the issue of ascertaining whether a visualization method could be regarded as a successful (useful?) tool. This evaluation process itself is a complex theme and involves delving into topics such as human computer interaction, perception factors, and qualitative measurement. Therefore we have decided not to address these questions here because the initial intention of this paper is to describe the HyperCell methods as well as to present some preliminary results. In fact, according to Grinstein et al.[15] and Wong and Bergeron[32], the evaluation and perceptual issues are two of the fundamental problems still facing *multidimensional multivariate visualization*.

The other term is *visualize*. Considering that the visualization process refers to a cognitive activity in which the human being engages (or interacts) in order to achieve insight about the subject under study[24], one could conclude that the visualization activity involves both *visualization* and *interaction* issues.

These very same issues have conducted us towards two sub-problems. The first one is to design a satisfactory visual method that would help the user build a cognitive map of the N-dimensional data based only on the graphical representation provided. Finding an answer to this question means solving the complex problem of graphically representing multidimensional entities through a two-dimensional device. This representation must be "good" enough to be interpreted and understood by a human mind accustomed to deal with the four-dimensional space-time world.

The second one is to design a visualization method that affords a powerful interaction mechanism to help overcome the limitation imposed by the low dimensionality of the display devices.

4.2 General Approaches

Indeed the dimensionality issue exerts a great influence in the design of visualization methods. When data to be visualized are defined in an N-dimensional space, where N is lower than four, there exist a great number of well-established techniques and procedures to deal efficiently with them[23]. As pointed out by Hibbard in [18]

> Visualization has been successful because so much computer data is produced that describe the four-dimensional space-time world that our eyes and brains evolved to see.

The problem arises when the visualization complexity or dimensionality, represented by N, is greater than four. In this case the range of available solutions starts to lessen.

In a general way, there are four approaches to visually represent both multivariate and multidimensional data and, consequently, try to answer this research's challenge:

1) Reduce the amount of data presented to two or three dimensions (e.g. statistical technique[2,6]). This includes also the use of multiple views of the data, each encompassing a subset of the dimensions (e.g. scatterplot matrix[10] for multivariate, and HyperSlice for multidimensional[29]).

2) Rearrange the axes to be non-orthogonal and display data along all axes simultaneously (e.g. such as in parallel co-ordinates[19], glyphs[9]/icons[26], and Andrews' Plot[1]).

3) Embed or combine data dimensions to form composite spatial dimensions, such as in dimensional stacking (e.g. worlds within worlds[13] or hierarchical axis[25]).

4) Map each data values from large datasets to a pixel of the screen, arranging them adequately (e.g. Pixel-oriented techniques[22], and natural texture mapping[20, 17]).

4.3 Why Reduction Philosophy?

Of particular interest is the first approach. It comprises those techniques that have as the central idea the reduction of the amount of data presented. The reduction process is achieved by either taking into account only a subset of the original data set (thus ignoring completely the rest of it) or by means of some mathematical treatment, originating a new data set

derived from the original one. This is fairly similar to the philosophy of "divide to conquer".

Assuming that the mind is not adept at processing large amounts of information but prefers to simplify complex information into patterns and easily understood configurations[27], we may hypothesise that such philosophy represents a valuable approach towards a feasible solution to our research problem.

This hypothesis is also supported by two facts: (1) sometimes a high-dimensional phenomenon can actually be governed (thus best described) by a few simple variables (called "hidden causes" or "latent variables")[24], hence the investigation of a reduced set of dimensions at a time might be helpful in finding hidden causes for a given phenomenon; (2) each method results in the distortion of the n-dimensional relationship between data points in order to map the data to the display. However, dimensionality reduction techniques explicitly attempt to preserve these relationships[8].

4.4 The Importance of the User Interface

As noted by Grinstein et al.[15] and Wong and Bergeron[32], there still exist many fundamental problems facing high-dimensional visualization. The former stresses that the research effort should shift away from the design of yet more visual displays towards a more rigorous evaluation of experimental visualization techniques. The latter argues that the three cornerstones for further research are the *geometric* issues (the data and its representation), *perceptual* issues (the human and its capabilities), and *evaluation* issues (the system and its effectiveness).

We believe that a fourth issue should also be considered: the *user interface* issues. We assume this by evidences found throughout several accounts of successful visualization methods[8,32]. They have systematically relied on a careful design of an intuitive user interface.

Therefore this evidence suggests that a visualization technique, regardless whether it is high dimensional or not, achieves efficiency and usefulness when its roots are based on a delicate balance between visual representation and interaction capabilities. Indeed, when dealing with high-dimensional representation this relation is far more critical.

4.5 Visual Representation: 2D *versus* 3D

The experimentation with a 3D visual representation (rather than a 2D one) has led us to challenge the three previously listed problems (c.f. Section 1.1) given by the *HyperSlice's* authors not to adopt 3D representation.

Today we can say that the first point, *slow volume rendering algorithm*, does not pose a difficulty as there have been several improvements in both algorithms and hardware necessary to generate volume rendered images[5].

The second problem, *difficulty in interpreting 3D visual representation rather than simpler ones*, is harder to solve than the previous one and it is also heavily dependent on the application domain. However, it has been addressed in HyperCell by employing a combination of two measures:

- Employing (standard) 3D visualization techniques, such as isosurfacing, volume rendering, etc., rather than creating new visual representations (what would probably increase the cognitive load on the user). This grants the user freedom to stick, for example, to those techniques s/he is accustomed to. Besides, Hibbard[16] has stressed the importance of the user being able to interactively select and combining different techniques;

- Allowing the user to control the incremental changing of the cell, from 1D up to 3D (and the reverse order). This might help user not to lose context, because s/he can start with only one dimension in one cell and gradually builds up to three dimensions and several cells. Furthermore it is always possible to return to a more familiar visualization (e.g. 1D).

The third problem relates to the *nontriviality of 3D interaction*. For this matter we have decided to make use of a two-dimensional user interface to handle the tasks of cell interaction and navigation in the hyperspace. Consequently most of the problems of three-dimensional interaction are avoided.

As said before, all the issues discussed in this section have driven the design decisions taken during the creative stage of our approach to the main research question. In the next section we present some implementation issues as well as the first test performed with the HyperCell prototype.

5. Implementation Issues and Preliminary Tests

In order to assess the methods found in the HyperCell proposition, we have developed a computational system that implements the basic principles behind it. The prototype is designed to work as a tool for visualizing scalar functions of many variables. Possible goals are: (a) Exploration of the surroundings of local/global extrema; (b) Comparison of functions; (c) Find topology (insights into the global structure of scalar fields of any dimension); (d) Search for patterns (discovering hidden patterns within the data) and structure; (e) Investigate subsets of data; (f) Visualise complex processes (mathematically modelled by multidimensional equations); (g) Theory formation (modelling a new theory that describes or predict the phenomenon from which the data set was obtained);

The prototype in its current form allows the dynamic creation of multiple workspaces and individual cells. It also affords basic interaction operations such as picking, rotation and translation performed in the visual space. However, no evaluation has been done at this stage.

The first prototype was built using C++ with the OpenGL™ API. Currently we are implementing a more complete version of HyperCell on the IRIS Explorer™[21] platform. Adopting such a platform has the following advantages: (a) several visualization algorithms are already present in the system, avoiding the time-consuming task of coding them, (b) we can take advantage of native features like collaboration[31], (c) it is possible to have a simulation module integrated with IRIS Explorer, creating the opportunity of computational steering, and (d) it is a good way of

making the work available to the scientific community that already uses the system.

The primary goal of this first prototype is to evaluate qualitatively the degree of efficiency of both the visual representation and interface features. As a test application, we have chosen the problem of visualizing a four-dimensional hyperspherical field centred at the origin and with an interior hypersphere of a fixed value, greater than that of the field..

The equation for the four-dimensional sphere centred at the origin with a radius r is $(x_1)^2 + (x_2)^2 + (x_3)^2 + (x_4)^2 = r^2$. Thus we can rearrange the terms to get the function $f(P) = (x1)^2 + (x2)^2 + (x3)^2 + (x4)^2 - r^2$, where P is 4D point.

Hence this function represents a hyperspherical field in 4D where the value increases as the square of the distance from the origin. In our example, $r = 1.0$, and P is defined over the range [-1.0;+1.0] in each of the four co-ordinate directions. The data is composed of 21 points sampled over the range of P, using an incremental step of 0.1 The interior ball has been assigned a high value (4.0) in order to be distinguishable from the hypersphere field data itself, which ranges from – 1.0 to 3.0.

A snapshot of the HyperCell prototype is shown in, Figure 8-a, in which we can see the hole depicted by the small reddish sphere at the centre of each cell. This little sphere disappears of course if it is not within the range of data spanned by the cell, as shown in Figure 8-b. Here we have applied a volume rendering technique to visualize the data, where the opacity reflects the value.
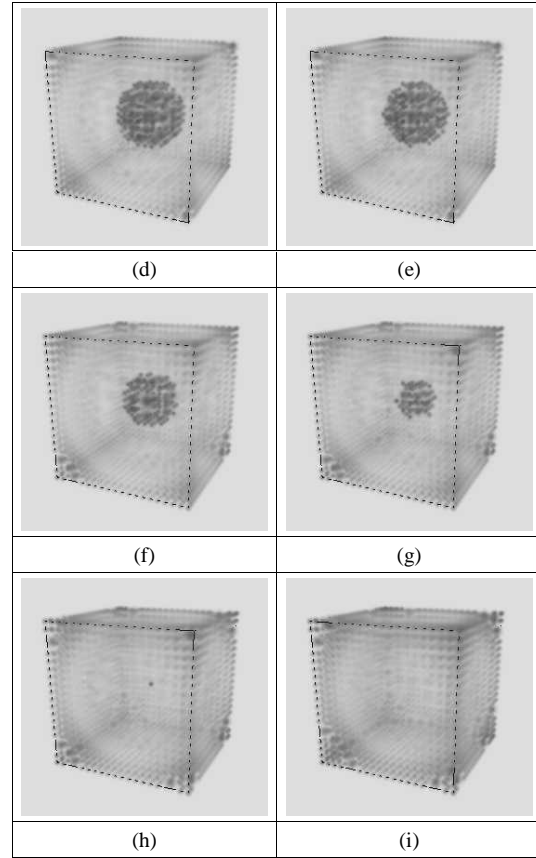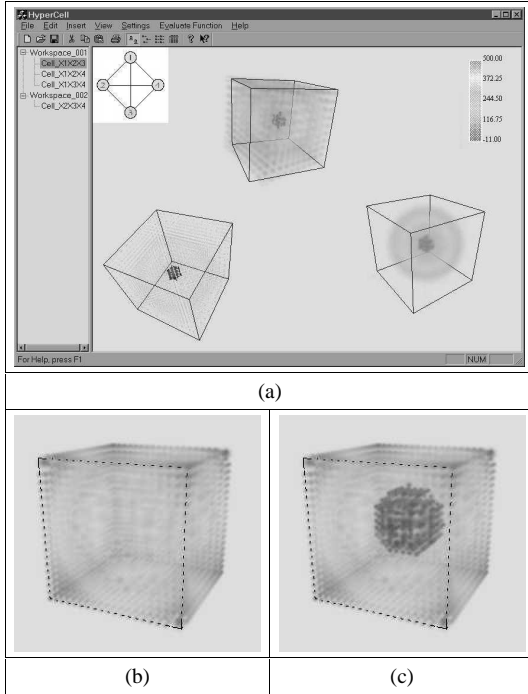

(a)

(b)

(c)

(d)

(e)

(f)

(g)

(h)

(i)

Figure 8: (a): *HyperCell's main window showing the visualization of the 4D hyperspherical field with a ball of radius 0.5 (top image). Three different cells have been created in the active workspace using a volume rendering technique. The cell located in the left corner has the ball depicted through an isosurface. Pictures (b) to (i) correspond to a cell built from x1, x2, and x3 dimensions. (b):Visualization of a cell without a ball. (c) to (i): The same cell with a ball inside (the red region). In each following image the focal point has been moved along the x4 axis from position (0,0,0,0) to (0,0,0,0.6), with steps of 0.1. In the last image (i) the ball is no longer visible.*

6. Analysing the Results

By experiencing with the prototype we have identified some positive aspects in the HyperCell method:

- **Scalability:** Conceptually there is no limit on the function's dimensionality. Because the user is in charge of the cell instantiation process, the data necessary to put flesh on a cell is required only when the visualization takes place. Thus it is unnecessary to store in the main memory the whole dataset necessary to drawn all the possible cells. Consequently no limit is imposed on the function dimensionality supported by HyperCell.

- **Efficiency:** The same user's ability to control the creation of cells (and used to justify the scalability feature)

also supports an efficient use of the computer processing power. As said before, the data necessary to plot a cell is required only when the user triggers the visualization. Hence there is no need of performing the computational processing necessary to drawn all the possible cells (combination of three dimensions) before the user requires so. Consequently the computational effort is greatly reduced.

▪ **Intuitive interface:** Although the user interface has not been fully evaluated, evidence from prior tests has led us to believe that the *interaction graph* bears an intuitive appeal. It allows direct manipulation using simple point devices to control the cells and perform interactive operations;

▪ **Cognitive load:** HyperCell has not introduced any new visual representation. Rather, it takes advantage of the users' supposedly cognitive baggage, by employing standard 3D visual algorithms such isosurfacing and volume rendering. These algorithms are already present in the platform in which HyperCell is being developed: IRIS Explorer ™.

▪ **Improved exploratory power:** The ability to create multiple *regions of interest* and organise them into distinct workspaces improves the exploratory tools present in the method. This fact becomes specially evident if we compare it with the case where only a single *region of interest* is available. Multiple *regions of interest* allow comparison of different locations in N-Space without losing the initial reference position.

Despite the claim that HyperCell possesses dimensional scalability, there might be two limitations to the function dimensionality. The first one is found on the user interface, which might become difficult to handle when the number of dimensions increases. This is because the space on screen reserved to it is restricted. The second one resides in the limited user's cognitive ability to deal with several cells simultaneously.

7. Final Remarks and Future Work

This paper has presented the current status of this research, which aims at investigating solutions to the problem of visualizing multidimensional scalar functions.

We have introduced our approach to the problem, called HyperCell. This method fits in the category of reduction techniques, which has as the basic idea to represent a N-dimensional (N>4) function as a set of lower dimensional subspaces (1D, 2D, or 3D), called cells.

In our approach the user is responsible for creating as many subspaces as necessary to investigate the function. Each cell employs standard three-dimensional techniques to visualize the data sub-space defined by cells.

The exploratory firepower relies on the manipulation of one or multiple *Regions of Interest*, which is a N-dimensional entity. Through operations such as *switching views*, *translation*, *rotation*, and *scaling* in N-Space, the user has the ability to analyse the multidimensional function.

The two-dimensional interface called *Interaction Graph* allows direct manipulation, using a mouse-like device, in order to create, manage, and perform interactive operations with the cells.

Currently we are experimenting with the prototype, performing preliminary evaluation, and assessing its effectiveness by applying it to artificial functions.

However, we are aware that important tasks lie ahead and need to be tackled shortly. The evaluation of usability factors for complex task (effectiveness, efficiency, and satisfaction) is one of them.

We need also to collect more evaluation measurements and quality indicators from the application of HyperCell to real-life problems. Two possible case studies are (1) Visualizing the Activation Extinction Model for Platelets Adhesion[10], and (2) Visualizing sensitivity analysis in parameter optimisation.

Further research will investigate possible improvements in our mechanism for navigating throughout the many subspaces. The main objective is to ensure that our navigation device will not let the user get lost during the exploration of the N-Space. This may involve (a) guiding the users to the next cell, or; (b) allowing the users to understand the transition between the cells. We feel that one possible realisation for this navigation mechanism should be based on the *Interaction Graph*.

Nevertheless, the preliminary conclusion suggests that the method may be considered a valuable complement to the previous approaches in the field of high dimensional visualization.

References

1. D.F. Andrews. "Plots of High-Dimensional Data", *Biometrics*, volume 29, pp. 125-136, 1972.

2. D. Asimov. "The Grand Tour: A Tool for Viewing Multidimensional Data". *SIAM Journal on Scientific and Statistical Computing* **6**(1):128-143, 1985.

3. C.L. Bajaj, V. Pascucci, G. Rabbiolo and D.R. Schikore. "Hypervolume visualization: a challenge in simplicity". *ACM Press* (*Proc. of IEEE/ACM Symposium on Volume Visualization*), pp. 95 -102, 172, 1998.

4. K.W. Brodlie, L.A. Carpenter, R.A. Earnshaw, J.R. Gallop, R.J. Hubbold, A.M. Mumford, C.D. Osland, and P. Quarendon. *Scientific Visualization - Techniques and Applications*, Springer-Verlag, 1992.

5. K.W. Brodlie and J. Wood. Recent Advances in Volume Visualization. *Computer Graphics Forum*, **20**(2), The Eurographics Association and Blackwell Publishers Ltd, pp. 125-148, 2001.

6. A. Buja, D.F. Swayne and D. Cook. Interactive High-Dimensional Data Visualization. *Journal of Computational and Graphical Statistics*, **5**(1):78-99, 1996.

7. M. Büscher, P. Mogensen and D. Shapiro. Spaces of Practice. Accepted for ECSCW 2001, 2001.

8. M.A. Carreira-Perpinan. "A review of dimension reduction techniques". *Technical report CS-96-09*, Dept.

Of Computer Science, University of Sheffield, UK, 1996

9. H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, volume **68**, pp. 361-368, 1973.

10. W.S. Cleveland. *Visualizing Data*, Summit, N. J.: Hobart Press, 1993.

11. T. David and P.G. Walker. "Activation and Extinction Models for Platelet Adhesion", *School of Mechanical Engineering*, University of Leeds (yet to be published), 2001.

12. B.S. Everitt. An Introduction to Latent Variable Models. *Monographs on Statistics and Applied Probability*, Chapman & Hall, London, New York, 1984

13. S.K. Feiner and C. Beshers. Worlds within worlds: metaphors for exploring n-dimensional virtual worlds. *In Proceedings of the Third Annual ACM SIGGRAPH Symposium on User Interface Software and Technology*, pp. 76 – 83, 1990.

14. J.D. Foley, A. van Dam, S.K. Feiner, J.F. Heghes. *Computer Graphics – Principles and Practice, second edition in C*, Addison-Wesley publish company, 1997.

15. G. Grinstein, A. Inselberg, S. Laskowski. Key Problems and Thorny Issues in Multidimensional Visualization. Proceedings of Visualization '98, pp. 505 –506, 1998

16. A.J. Hanson. Rotations for N-Dimensional graphics. *Graphics Gems V*, Academic Press, Cambridge, MA, pp. 55-64, 1995.

17. C.G. Healey, and J.T. Enns. Large Datasets at a Glance: Combining Textures and Colors in Scientific Visualization. *IEEE Transactions on Visualization and Computer Graphics*, **5**(2), pp. 145 –167, 1999.

18. B. Hibbard. Top Ten Visualization Problems. *SIGGRAPH Computer Graphics Newsletter – VisFile*, ACM Press, **33**(2), 1999.

19. A. Inselberg and B. Dimsdale. "Parallel Coordinates: A Tool for Visualizing Multi-Dimensional Geometry". Proceedings of the First IEEE Conference on (Visualization `90), pp. 361-370, 1990.

20. V. Interrante. Harnessing natural textures for multivariate visualization. *IEEE Computer Graphics and Applications*, **20**(6):6 –11, 2000.

21. IRIS Explorer 5 (2000), Technical Report, Silicon Graphics Computer System, Mountain View, <http://www.nag.co.uk/visual/IE/iecbb/DOC/Index.html>

22. D.A. Keim. Designing Pixel-Oriented Visualization Techniques: Theory and Applications. *IEEE Transactions on Visualization and Computer Graphics*, **6**(1):59 –78, 2000.

23. P.R. Keller and M.M. Keller. *Visual Cues, Practical Data Visualization*, IEEE Press, 1993.

24. A. M. MacEachren. *How Maps Work, Representation, Visualization and Design*, New York, The Guildford Press, 1995.

25. T. Mihalisin, J. Timlin, and J. Schwegler. Visualizing multivariate functions, data, and distributions. *IEEE Computer Graphics and Applications*, **11**(3): 28 –35, 1991.

26. R.M. Pickett and G.G. Grinstein. Iconographic Displays For Visualizing Multidimensional Data. *In Proceedings of the 1988 IEEE International Conference on Systems, Man, and Cybernetics*, volume **1**, pp. 514 – 519, 1988.

27. R. Spence. *Information Visualization*, ACM Press Books, 2001.

28. R. van Liere and J.J. van Wijk. Visualization of multidimensional scalar functions using Hyperslice, *CWI Quarterly*, **7**(2):147-158, 1994.

29. J.J. van Wijk and R. van Liere. HyperSlice – Visualization of scalar function of many variables. *In Proceeding of the IEEE Conference on Visualization (Visualization '93)*, pp. 119 –125, 1993.

30. E.J. Wegman. Hyperdimensional Data Analysis Using Parallel Coordinates. *Journal of the American Statistical Association,* **85**, 411, Theory and Methods, pp. 664-675, 1990.

31. J. Wood, H. Wright and K.W. Brodlie. Collaborative Visualization. *Proceedings of IEEE Visualization 1997 Conference*, pp 253-260, ACM Press (1997).

32. P.C. Wong and R.D. Bergeron. "30 Years of Multidimensional Multivariate Visualization". *In Gregory M. Nielson, Hans Hagan, and Heinrich Muller, editors, Scientific Visualization - Overviews, Methodologies and Techniques*. IEEE Computer Society Press, pp. 3-33, 1997.