STAR : A general architecture for the support of Distortion Oriented Displays

Paul Anderson, Ray Smith and Zhongwei Zhang

Gippsland School of Computing and Information Technology,
Monash University,
Australia.
{Paul.Anderson, Ray.Smith, Zhongwei.Zhang}@fcit.monash.edu.au

Abstract

Distortion oriented displays (DOD) are an interface approach for supporting navigation through large visual datasets without losing context. They are particularly well suited to applications such as Geographical Information Systems (GIS). Unfortunately such applications are characterised by vast quantities of data requiring substantial time to upload and display. This is completely at odds with the requirements of a DOD, in which the dynamics of the interaction are of paramount importance. STAR (Self Tuning Architecture) addresses this disparity by dividing the interface into a set of tuneable, independently threaded modules responsible for key tasks in the system. The dynamic response of the interface is continually monitored and the modules tuned or detuned to achieve the best display possible within the response constraints of the system.

Keywords

Distortion-oriented presentation techniques, User-interface design, Information visualisation, Geographical Information Systems.

Introduction

Applications such as Geographical Information Systems (GIS) present the user with a large visual dataset (map) through which they must navigate and search out features of interest. The traditional approach to facilitate this navigation is to present the user with a restricted view of the map through a window and allow them to zoom in to obtain more detail (and a more restricted) view. Unfortunately there is some evidence *[Tolsby 1993]* to suggest that users of this type of interface spend significant amounts of time zooming out in order to re-establish their context to the rest of the map.

A solution to this problem in the form of a bifocal display was suggested by Apperley and Spence in 1982 *[Apperley 1982]*. Their solution was to present the user with a movable virtual magnifying glass, within which a detailed view of the point of focus is presented. Surrounding this detail 'focus' region the rest of the map is presented in a visually compressed view with superfluous detail filtered out. The image of this 'context' view has to be distorted in such a way as to ensure feature matching across the boundary with the region of focus. In this way the user is able to see the detail they require presented within a view that is able to convey its context. Since the early work on bifocal displays a number of distortion oriented displays (DOD) have been described *[Leung & Apperley 1993]*. It has been observed that any number of DOD's are possible since an infinite number of continuous and discontinuous magnification functions can be used to define them. An example of a DOD that uses a continuous magnification function is the graphical fisheye display *[Sarkar & Brown 1993]* which produces an image similar to a photographic fisheye lens.

An important property of DOD's is that the user should be able to move the point of focus around the screen and experience no discernible delay in the redisplay of the interface. In order to achieve this fluid interaction the display needs to be redrawn at least as fast as the human visual processor cycle time of 100 msec *[Card, Moran & Newall 1983]* and preferably two to three times faster. Unfortunately the amount of computation required for the redisplay of a DOD is considerable. Apperley and Spence resorted to a hardware based solution to this problem. More recent examples have been software based but have been restricted to fairly small datasets in order to achieve the required dynamic response.

This is unfortunate since it is applications such as GIS's which are characterised by very large datasets that are most in need of this type of navigation aid. It is towards the development of techniques to enable the practical implementation of DOD's for those demanding applications that the work reported in this paper is directed.

Related Work

Some previous work has been done *[Robertson, Card & Mackinlay 1993]* in the development of an Information Visualizer Architecture aimed at providing large workspaces, multiple agents, real-time interaction and visual abstractions. At the heart of their architecture is the Cognitive Coprocessor which is a controlled resource scheduler. The Cognitive Coprocessor has a Governer mechanism that monitors the display cycle time and directs cooperating rendering processes to reduce rendering quality to increase cycle speed when the cycle time is too slow.

The S**e**lf **T**uning **AR**chitecture (STAR) described in this paper differs from the Cognitive Coprocessor in the following ways :

- designed for use with very large complex datasets (eg GIS) and navigation within these using DOD techniques
- uses a range of regimes for reducing image complexity in the context region to adjust re-display time dynamically
- the regimes are independent of the architecture with appropriate techniques being chosen for the type of display being supported
- the range of regimes are controlled through the use of a single encompassing tuning variable

Demanding Applications

In order to indicate the magnitude of the problem involved in implementing a DOD, the following measurements were obtained for a GIS (Infocad) running on a HP9000 715/75 with a graphics accelerator.

Activity	Time
Upload 25000 vectors	40 sec
Calculation of fisheye display for 25000[1] vectors	~1 sec
Drawing display	25 msec

Admittedly the above example is for a fisheye display which involves considerably more computation than simple DOD such as bifocal. However the objective of this work is to develop an architecture capable of supporting any type of DOD. So how can this enormous disparity between the time taken to produce a DOD and the dynamic response be addressed?

Common to all DOD is the division of the display into a focus region, representing the virtual magnifying glass that the user moves around the screen and one or more context regions that allow the user to view the relationship between features in their current focus with those outside the focus. It follows that in general all the data that can be presented in the focus region should be displayed (since this contains the information that the user is looking for) whereas only a subset of the available data need be displayed in the context region. Extending this idea further, degradation of the context image is more acceptable than degradation of the focus. Hence tuning of the interface should concentrate on the context image. The gains that can be achieved by this may, in some cases, be greater than immediately obvious. The context image needs to be distorted in order to achieve feature matching across the boundary with the focus region. The data transform required for this distorted image may require substantial computation. Reducing the quantity of data that needs to be transformed in this way can produce a large reduction in this computational overhead.

As discussed above, DOD usually divide the display into focus and context regions. In the focus region the data is generally undistorted and it is relatively easy (fast) to calculate the data mappings in this region. The focus region presents the highest magnification of the display. Some data will be associated with items that are too small to be visible in this region. It follows that these data items would be too small to be visible anywhere on the display, and consequently can be filtered at source. Our experience with real datasets has shown that this typically reduces the quantity of data by ~80 %. In addition to this general size filtering some areas of the map offer very high levels of detail. Within the context region such detail is beyond the resolution of the display. Hence a further size based filtering can be applied based on data population densities.

The context region is there to show context. It follows that much of the detail in this region not only could but should be removed. Within STAR this is achieved by selecting data based on user defined layers. The calculation of data mappings in the context region is often complex and incurs a major time penalty, hence any reduction in this region can produce considerable gains in performance.

Apart from these static filtering regimes, a number of dynamic filtering approaches are possible and it is these that form the backbone of the STAR architecture. The basis of this approach is the dynamic adjustment of the quantity of data displayed based on measured system performance. Each drawing request is manifested as a time stamped event in a queue. The time taken to service requests can hence be monitored and the

[1]A single vector in the original data can translate into a number of vectors within the fisheye display.

quantity of data displayed modulated in line with the difference between the actual response time and the optimal (ideal) response time. This dynamic adjustment is referred to as tuning.

Tuning Variables

A number of parameters can be monitored to adjust the system performance. These include:

i) Polynomial Term Limit
Within some DOD a straight line in the original map translates into a curved line within the distorted region of the display. One approach to achieve this mapping is by fitting a polynomial to the data. Obviously the greater the number of terms calculated, the better the mapping. On the other hand the calculation of more terms incurs a greater calculation overhead, and a greater drawing overhead since the resulting data is then mapped onto an increased number of vectors. Reducing a limit on the number of terms calculated can produce substantial performance improvements. Such approximations within a distorted image used purely for establishing context are entirely reasonable.

ii) Size Filtering
Static filtering of data based on size was described above. Increasing the minimal size of data items within the context region is simple to implement and has a low filtering overhead. An upper limit, beyond which the minimal size cannot be extended needs to be set otherwise all of the data could be removed in preference to the application of other tuning regimes, which would be unreasonable.

iii) Layer Based Filtering
Data in the GIS is categorised on the basis of user defined levels (the more important the level the lower the level number). This level information is used for static filtering of context region information. If two or more user defined data levels are displayed in the context region then the maximum level number can be dynamically adjusted to filter out whole categories of data. The overhead incurred in this approach is very low.

Other tuning variables may be possible for particular types of DOD. What is important with a tuning variable is that the overhead incurred in applying the filter is less than the performance gain obtained by applying it.

The Tuning Process

In practice, a number of tuning variables can be identified in the system. In this section we will consider a simplified system that has only a single tuning variable T_V, that represents the percentage of the available data that is to be presented within the context region of the display. Practical values for T_V might be 10 through 100.

The tuning process monitors the time taken to redraw the screen. Let us assume an optimal redraw time of 50 msec. A redisplay time of more than this means the system is not responding fast enough and needs to be detuned, ie T_V needs to be reduced. Conversely, if the redisplay time is less than 50 msec then it would be possible to produce a better display within our time constraint, so T_V can be increased.

Let us now consider how the user is likely to perceive these changes in performance. Users do not continually move the focus region around. As is the case with all computer systems the interaction between user and system is characterised by bursts of action interleaved with periods of complete inactivity. Focusing on the response of the system at the start of one of these activity bursts, consider the following two cases :

- T_V is too high, and the resulting system response is too low (Figure 1a).
- T_V is too low, and the resulting system response is more than acceptable (Figure 1b).

In the first case the lack of performance will result in the tuning process reducing T_V to compensate. However, from the user viewpoint the interaction will initially seem to be sluggish. In the second case no such change in system performance will be perceived by the user. It follows from this that it is better to start with T_V too low, and tune the system up than to have the converse case.

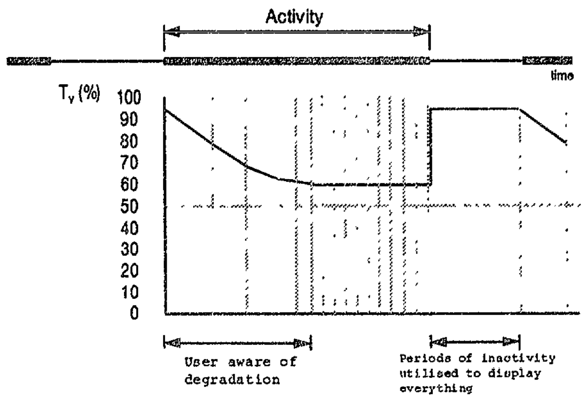
An additional benefit of starting with T_V too low is that the system will tune itself to optimal performance quicker since adjustment of T_V can only occur at the completion of each display interval.

The scenario presented above makes the following assumptions:

1. Burst mode activity is the norm.
2. The data distribution across the screen is uniform.

Point 1 is probably valid, however 2 is probably not. If data distribution is not uniform (i.e., point 2 not valid) it will result in a constantly changing re-display rate, hence T_V would oscillate up and down rather than following a straight line as depicted in Figure 1. The question then arises as to how the user perceives this changing performance. The answer is surely that if the user perceives any change in dynamics of the system, then that is a bad thing.

(a)



(b)

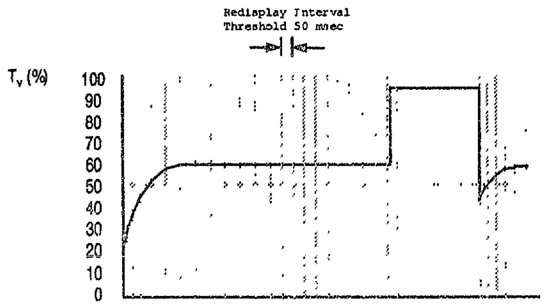


Figure 1: Alternative tuning scenarios.

Since users will only perceive changes if the redisplay interval becomes greater than a certain threshold (~50 msec) it is arguably better to tune the interaction interval to a period shorter than the threshold so intervals that give rise to detuning are not perceived by the user.

Periods of user inactivity need to be considered as a special case since during these periods the display should be presented in its most complete form. The rate at which the user is moving the point of focus can be easily monitored. If this scan speed is found to be zero, the T_V could be set to 100 to achieve optimal image quality. However, the next burst of activity would be likely to give rise to a redisplay interval greater than the redisplay threshold with the result being that the user would become aware of a change in performance. The solution is for the tuner process to monitor scan speed and at the start of a new burst to set T_V at a low level (Figure 2).

In the algorithm presented in Figure 2, T_V is adjusted in increments of a fixed size. An obvious improvement in this would be to adjust the size of the increment in line with the magnitude of the difference between *redisplay_interval* and *redisplay_threshold*, hence forcing the *redisplay_interval* to converge to the threshold more rapidly. (This assumes a linear relationship between *redisplay_interval* and T_V. If T_V simply controls the quantity of data being displayed then this may be true).

In the discussion presented above there was only a single tuning variable. In practice there is the potential for any number of tuning variables that control parameters such as the minimum size of vectors displayed, the number of polynomial terms calculated in mapping vectors to a context surface and so on.

In dealing with a set of tuning variables the question arises as to which should be adjusted, and by how much. In the general case this cannot be defined since different types of DOD are likely to have different sets of tuning variables. What can be done is:

- suggest a scheme for managing tuning variables
- suggest approaches for identifying which tuning variables should be adjusted, and by how much.

```
TVINC = 5
stationary = TRUE
REPEAT FOREVER
  GET scan_speed
  IF stationary THEN
    IF  scan_speed NE 0   THEN
      stationary = FALSE
      Tv = lower_limit     // eg 10
    ENDIF
  ELSE                      // not stationary
    IF  scan_speed EQ 0   THEN
      stationary = TRUE
      Tv = 100
    ENDIF
  ENDIF

  IF NOT stationary  THEN
   GET redisplay_interval
    IF redisplay_interval<redisplay_threshold THEN
      IF  Tv < 100   THEN
        Tv = Tv + TVINC
      ENDIF
    ELSEIF redisplay_interval>redisplay_threshold  THEN
      IF  Tv > lower_limit THEN
        Tv = Tv - TVINC
      ENDIF
    ENDIF
  ENDIF
END REPEAT
```

Figure 2 : Tuning Algorithm

The scheme used for managing variables in STAR relies on maintaining a simple primary tuning variable T_p and defining a function mapping the value of T_p to the value of each specific tuning variable in the system. In this system, changing the value of T_p (between 0 .. 100) results in one of the more specific tuning variables being adjusted in line with the tuning function. (In the implementation the tuning function simply maps onto a table, permitting the new values for specific tuning variables to be rapidly located). Given this scheme the tuning system of STAR can be defined in the following way.

The tuning system $S = \{ T_p, T, F_t, N_t \}$
where

T_p	=	the primary tuning variable
T	=	a set of specific tuning variables
F_t	=	a function mapping $T_p \rightarrow T_{(x)}$
	=	$\left\{ \forall x \in 1 \cdot\cdot N_t \bullet \left(T_p \rightarrow T_{(x)} \right) \right\}$
N_t	=	the number of specific tuning variables

Any number of tuning variables (F_t) could be defined. However a set of criteria for identifying acceptable tuning functions can be stated, reducing the number of alternatives. The first criteria is that the relationship of T_p to the *redisplay_interval* (R_i) should be linear. This is desirable to facilitate rapid convergence of run-time tuning. The second criteria is that the display should be the best possible. Given two displays produced in the same R_i, the one containing the most data is likely to be the best.

The question then arises as to how an acceptable tuning function is found. Since the performance of the interface is closely related to the size and distribution of the visual dataset, any tuning function will have to be a compromise. However an empirical approach based on a large uniform artificial data set is possible. Given N_t tuning variables, each would have a specific number of possible values (between 1 .. N(x)). The number of permutations is the product of N(x) for x : 1..N_t. Hence given 4 tuning variables, each with 10 possible values (a worst case scenario), the number of permutations is 10000. An artificial dataset with a dense, uniform data distribution could be automatically scanned with the specific tuning values set to each one of these permutations and the redisplay interval recorded for each. If R_i is 0.1 msec, with 10 samples being taken for each permutation, the complete scan could be carried out in 10000 sec or about 3 hours.

To locate the appropriate permutations from the resulting data the shortest and longest redisplay interval within the range *0..redisplay_threshold* would be found and the corresponding permutation recorded. Next the interval covered could be divided into an appropriate number of steps (10 or 100) and the nearest mapping to each step interval located in the database. These would then represent the appropriate mappings for T_p corresponding to each interval. This empirical approach to defining the tuning function would only need to be carried out for each new type of DOD, the same mapping being used for all dataset examined henceforth.

Other techniques for finding appropriate mappings may be possible. For example, rather than having a separate tuning phase, the system could monitor its own performance as it is used and devise its own tuning functions. Genetic algorithms, neural network or fuzzy logic techniques may be applicable for this purpose.

The STAR Architecture

The STAR architecture can functionally be divided into three distinct divisions, responsible for data acquisition (uploading), data transformation and display, and performance monitoring and tuning (Figure 3). Changing from one form of DOD to another only affects the data transformation and display module.

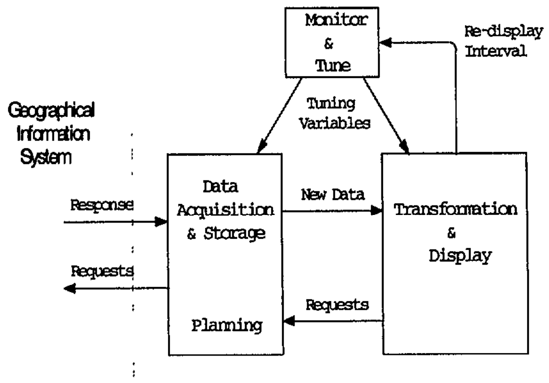


Figure 3: The STAR architecture.

In addition to the basic capabilities of the data acquisition modules, STAR also provides for a planning function to be incorporated into this module. The planning function monitors each request for data and attempts to make intelligent predictions as to what data will be required in the very near future. These requests are then added to the request queue with a low priority. In this way it may be possible to achieve improved performance.

Conclusion and Discussion

The STAR architecture relies on a combination of static and dynamic filtering in order to achieve the dynamic response required of a DOD. A working prototype has been implemented that has demonstrated the feasibility of this approach. So far only display output issues have been addressed, since it was felt that the problems of achieving the required level of performance were of the greatest importance for demanding applications. Having tackled this issue, we now plan to investigate user input through the display. The quantities of data involved are relatively small, hence it is felt that this side of the interaction should not present any great difficulties.

Finally we plan to investigate a number of different types of DOD. Our initial investigations indicate that DOD based on continuous transform functions are better suited to practical use, since their apparent distortion is less than that seen with simple DOD types such as bifocal. Released from the shackles of performance constraints, we will have a largely free rein to assess any form of distortion transform that seems promising.

References

Tolsby, H. : *Navigating in a Process Landscape.* HCI Proceedings of 3rd International Conference EWCHI '93 Moscow Russia. Selected Papers, August 1993 pp 141-151

Leung, Y. and Apperley, M. : *A Unified Theory of Distortion-Oriented Presentation Techniques.* Massey University School of Mathematical and Information Sciences Report Series A No 93/1.

Apperley, M.; Tzavaras, I. and Spence, R. : *A Bifocal Display Technique for Data Presentation.* Eurographics '82 pp 27-43., DS Greenaway, EA Warman (eds) North-Holland Publishing Company

Card, S.; Moran, T. and Newell, A. : *The Psychology of Human-Computer Interaction.*, Erlbaum Ass., Hillsdale, New Jersey, 1983

Robertson, G.; Card, S. and Mackinlay, J. : *Information Visualization Using 3D Interactive Animation.*, Communications of the ACM April 1993 Vol 36 No 4, pp 57-71.

Sarkar, M. and Brown, M. : *Graphical Fisheye Views.* Technical Report CS-93-40, Dept. of Computer Science, Brown University September 1993.