

Automatic Camera Path Generation for Graph Navigation in 3D

Adel Ahmed

Peter Eades

National ICT Australia Ltd.,
Locked Bag 9013,
Alexandria NSW 1435, Australia.
Email: {adel.ahmed, peter.eades}@nicta.com.au

Abstract

This paper addresses the Focus+Context issues involved in navigating very large graphs in 3D Euclidean space. The main aim of the approach presented in this material is to preserve and perhaps enhance the user mental map during the transition phase from one focus object to the other. This approach, *NavAssist*, accomplishes this task by filling the users view port with the maximum amount of information without compromising user orientation. A simplified 3D view point path-generating technique is presented that reveals the largest number of graph components during 3D space navigation.

Keywords: *NavAssist*, Graph Navigation, 3D Virtual Environment, Focus+Context, Visualisation, Virtual Camera.

1 Introduction

Despite the advances in computer technology including human-computer interface in recent years the display space used to exchange information with humans is finite. No matter how fast computers get and regardless of the storage capacities of modern computers, the communicating channel is still narrow and small. Information structures are becoming so complex that there has been considerable interest in the question of whether a 3D visualisation will reveal more information than a 2D visualisation. No matter which method is chosen for visualisation the concept of *focus+context* stresses that in multi-level data visualisation, the user must be able to zoom in and focus and maintain an overview of the context of the data focused on all at the same time. In many cases one is compromise to achieve the other. Distortion of space is one solution (Carpendale 1999). Another solution is to offer multiple representations at different scales simultaneously.

Early research in display space optimisation suggested that comparatively more information can be easily displayed in three-dimensions without visual clutter (S. Card, G. G. Robertson & J. Mackinlay 1991). Research evidence suggest that 3D drawings of graphs are more effective in providing more insight of large information spaces (C. Ware, G. Franck 1996), (Y. Xiao, P. Milgram, A.G. Ryman 1993). Sophisticated and specialised methods are needed to navigate visualised data in 3D space due to the increased degree of freedom in movement through information-rich 3D virtual environment (Stephen Hughes, Joseph

Manojlovich, Micheal Lewis, Jeffrey Gennari 2003). The ability to manipulate the viewpoint is the essential task in any visualisation package (Hix, D., E. Swan, J. Gabbard, M. McGee, J. Durbin and T. King 1999). In this paper we propose *NavAssist*, a constrained navigation method for a graph with a pre-computed layout in 3D. *NavAssist* extends previous work on smooth animation from one focus point to another proposed by Wijk *et al.* (Jarke J. van Wijk, Wim A. A. Nuij 2003) into 3D space. The initial objective is to preserve the mental map of the user while he/she shifts focus from one piece of information to another. The technique exploits motion parallax phenomenon, an essential cue to perceive an immersed 3D environment (Colin Ware 2004). The remainder of this paper is organised as follows. Previous work and literature survey is presented in section 2. Then section 3 describes the basic *NavAssist* model used in this paper to describe the assisted navigation algorithm. Section 4 explains the details of *NavAssist* navigation algorithm. The two phases of the algorithm are presented in 4.2. Section 4.3 discusses implementation issues due to the discrete nature of graph navigation problem. The paper is concluded with some observations about the algorithm in section 5.

2 Background

A fair amount of effort has been put in viewpoint position finding. This is also referred to as camera position finding. A model for distinguishing between good viewpoint and bad viewpoint was proposed by Webber (1998) where multiple viewpoints can be tested against a bad viewpoint arrangement in 3D space to determine whether they are good or bad (Richard J Webber 1998). The problem of automatic camera positioning and automatic camera path generation in the context of historical data visualisation was addressed by Stoev *et al.* (Stanislav L. Stoev, Wolfgang Starber 2002). It is based on maximising the projected area, depth and the inverse of the normalised absolute difference between them in a 3D scene. *StyleCam* (Nicholas Burtnyk, Azam Khan, George Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach 2002) is yet another recent application that employs automatic camera path generation, which takes the virtual camera on an automatic calculated path from one predefined constraint surface to another. Experimental results conducted by Bowman *et al.* listed the advantages and disadvantages of three assisted navigation techniques of virtual environments (Bowman, D., Koller, D., and Hodges, L.F. 1997).

Navigation, according to Tan *et al.*, can be broken into three subtasks, exploration to gain survey knowledge; search to locate an object; and inspection to maintain a particular view of the object (Desney S.

Tan, George G. Robertson, Mary Czerwinski 2001). For the exploration and search subtasks, studies suggest that the six-degree of freedom for view point control offered by the 3D virtual environments imposes huge disadvantages on navigation (Stephen Hughes, Michael Lewis 2002). The experimental results obtained by Hughes *et al.* (Stephen Hughes, Joseph Manojlovich, Micheal Lewis, Jeffrey Gennari 2003) also suggest that controlled navigation, where some of the navigation control is done by the computer, is more desirable especially when it provide hints about the environment while taking the movement burden through the virtual space off of the user. Further more, these techniques assist users in accomplishing the tasks more effectively and efficiently. Geometry-driven navigation was proposed by Hanson *et al.* (A. J. Hanson, H. Ma 1995) and further extension work was carried out by Hanson *et al.* (Andrew J. Hanson, Eric A. Wernert, Stephen B. Hughes 1999). Other studies (Melanie Tory, Torsten Möller, M. Stella Atkins, Arthur E. Kirkpatrick 2004), (Springmeyer, R. R., Blattner, M. M., Max, N. L. 1992) have explored the effect of combining two-dimensional and 3D views for better orientation and relative position tasks, but the method proposed was applicable to relatively small 3D virtual models.

3 Navigational Model

Most 3D visualisation environment provide a mechanism to control and manipulate the viewpoint. This viewpoint is defined by a virtual camera in the 3D virtual space. A brief description of the camera model adopted for NavAssist is presented next.

3.1 Camera Model

Many models have been proposed for the virtual camera. We adopt the simple three-parameter model which consists of the following:

1. Camera position denoted by the three coordinates (c_x, c_y, c_z)
2. Camera target-point position (t_x, t_y, t_z) and
3. The up-direction represented by the vector $u = \langle u_x, u_y, u_z \rangle$

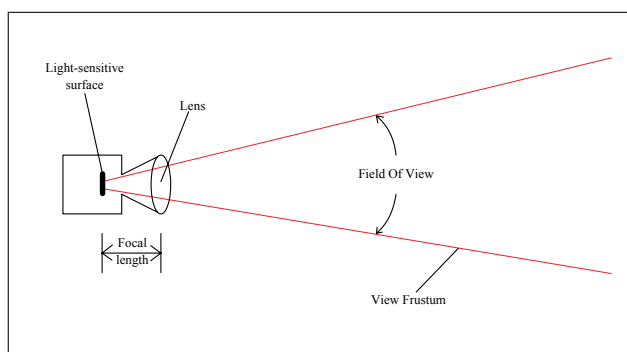


Figure 1: Camera Focal length, Field of View, and View Frustum

Fig 1 shows some properties of the camera model. The camera target-point defines the direction of view for the camera, in other words the camera always “looks at” the camera target-point. The content of the view port depends on the values of these three parameters at a given time. Perspective projection defines the camera view frustum. Objects lying inside the view frustum will be visible to the user and vice-versa.

Other secondary parameters that are taken into account in this simple camera model are the directly linked properties *Focal length* and *field of view*. In brief, the distance between the lens and the light-sensitive surface, whether film or video electronics, is called the focal length of the lens. Focal length affects how much of the subject appears in the picture. It is measured in millimetres. A 50 mm lens is common standard for photography. The field of view, which is directly linked to the focal length defines the width of the view as an angle with its apex at the viewpoint and the ends at the sides of the view. It is measured in degrees of the horizon because it represents the arc of the camera’s horizon. A 50 mm lens shows 46° of the horizon. The perspective associated with 50 mm lenses appear normal, partly because it is close to what the eyes see, and partly because such lenses are widely used in photography.

To extend the two-dimensional zooming and panning idea proposed by Wijk *et al.* (Jarke J. van Wijk, Wim A. A. Nuij 2003) into 3D space, zoom is replaced with camera dolly, which simply refers to the movement of the camera along its *line of sight* or *direction of view* axis. In this context zoom-in/zoom-out would mean dolling toward/away from an object in the virtual space. Further more, panning is replaced with the coupled transformation of camera and camera target-point positions altogether in 3D space. The relative position of camera and camera target-point in virtual space will determine the roll/pitch/yaw parameters of the virtual camera. Perspective projection is considered in the remainder of this material.

3.2 Graph Model

Navigation of graphs in 3D Euclidean virtual space is considered. A graph $G = (V, E)$ consists of a set of nodes V and a set of edges E where each edge $e = (u, v)$ represents a relationship between two nodes $u, v \in V$. The navigation is more effective if the graph has a pre-computed layout, thus a pre-computed suitable 3D-layout of the graph is assumed. Furthermore navigation of a graph, in general involves transferring user focus from one graph element, usually a graph node, to another. The automatic path generation technique is based on such structured scenario.

4 Automatic Path Generation

For simplicity, the automatic path calculation and generation on a two-dimensional plane is presented first. The generated path may later be mapped into 3D space through a series of translations and rotations. It is assumed that data is visualised at several levels of scale. Each node of the graph, when zoomed into, displays those extra visualised details. One good example of such visualisation are MoireGraphs (T. J. Jankun-Kelly, Kwan-Liu Ma 2003).

A simple and trivial method to shift user focus from one node to the other, say from P_1 to P_0 in Fig 2, would be to make the virtual camera and its target-point follow the straight line path defined by P_1P_0 . But during the entire journey the direction of camera view frustum would be fixed. No extra information would be displayed in the view port.

An alternative method would be to swing the camera view frustum back and forth during the camera journey so that the user may become more aware of the surroundings. But changes in the view do not necessarily help in cognition and spacial awareness. Certain constraints must be applied on the motion of the virtual camera frustum that will enhance spacial awareness.

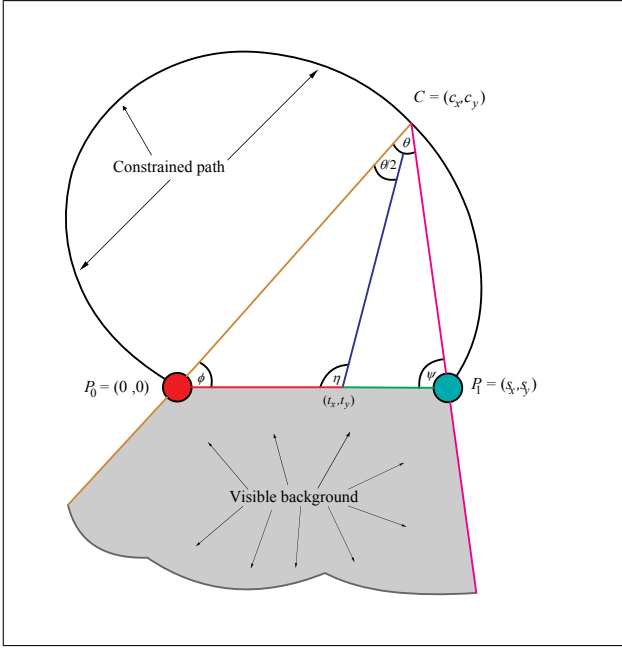


Figure 2: $C = (c_x, c_y)$ is the camera position, P_0 is the current focus node and $P_1 = (s_x, s_y)$ is the next focus node. $\theta = \text{FOV}$, which is constant.

4.1 Requirements and Constraints

The objective of NavAssist is simple. Smooth transition from one focus point to another is sought such that perceptual visual flow in the view port remains constant. In other words, we seek a smooth movement of the camera view frustum that reveals as much information about the surrounding as possible. The aim is to fill the user view with as much background information as possible during the virtual journey of the camera from the start point to the end. It was observed by Bowman *et al.* that sudden changes in the viewpoint position, also known as *teleportation*, reduces the user's spatial awareness (Bowman, D., Koller, D., and Hodges, L.F. 1997). Therefore sudden change in the scene should be avoided. For this purpose a smooth continuous curve is desired that defines a path in the plane. To further reduce the probability of user disorientation during navigation it is desirable to make sure that the current focus node and the next focus node are always visible to the user. This means that both nodes must be inside the camera view frustum throughout the journey of the camera. Throughout the remainder of this material frame based animation terminology is adopted for describing events that happen in a certain time interval. To summarise the objectives in brief, NavAssist produces a camera path in 2D that:

1. Shows as many graph nodes possible to build user awareness during the camera journey, and
2. Keeps the current focus node and the next focus node in the camera view frustum to avoid user disorientation.

The path computation algorithm is presented next.

4.2 Path Computation

The path computation process can be broken down into two phases. In the first phase a simple curve defining the camera path is constructed. The second phase involves choosing one curve from a family of curves that define a 2D manifold shown in Fig 3. The

chosen curve must maximise a certain optimal function.

4.2.1 Phase 1: Path Calculation in 2D

For the first phase let the current focus node be P_1 and the next focus node be P_0 as shown in Fig 2. According to the camera target-point concept, shifting focus can be represented as transformation of the camera target-point position, $T = (t_x, t_y)$ from the location of P_1 to the location of P_0 . If $P_0 = (0, 0)$, $P_1 = (s_x, s_y)$ and $C = (c_x, c_y)$, where C denotes the location of the camera in the plane. Then we are looking for those values of $C = (c_x, c_y)$ such that the angle $\widehat{P_1 C P_0}$ in the triangle $\triangle P_1 C P_0$ is always constant. Let $\theta = \widehat{P_1 C P_0}$ denote the field of view of the camera. In general $\theta = 45^\circ$, which depicts a 50 mm camera lens and remains constant throughout the navigation period. In general variable values of θ may also be considered if desired. For the time being fixed value of θ is assumed. From Fig 2 it is clear that $\eta = \phi + \frac{\theta}{2}$ and $\psi = \phi + \theta$. Then for values of $\phi \in [0, 2\pi]$, the position of the camera $C = (c_x, c_y)$ that satisfies the constraints can be computed by

$$c_x = \frac{t_x \tan \eta - s_x \tan \psi}{\tan \eta - \tan \psi}$$

$$c_y = (c_x - t_x) \tan \psi$$

where t_x can be calculated by the parametric line equation that joins P_1 and P_0 . It is worth noting that the path generation depends on the values of s_x , t_x and ϕ when $|t_x| \leq |s_x|$. Thus the path generation function can be defined as

$$f : \mathbb{R}^2 \times [0, 2\pi] \mapsto \mathbb{R}^2$$

where $s_x, t_x \in \mathbb{R}$ and $\phi \in [0, 2\pi]$. It is not difficult to verify that values of c_x and c_y are defined for all values of s_x , t_x , and ϕ .

In summary, the first phase computes a suitable path in 2D that conforms to some constraints.

4.2.2 Phase 2: Choosing a Context Aware Path

After calculating the 2D path, this second phase chooses a path that will display maximum background information. After mapping the 2D path into 3D using $g : \mathbb{R}^2 \mapsto \mathbb{R}^3$ where g represents a combination of translation, rotation, and/or scaling transformations in 3D space, the aim is to make use of the shaded area in Fig 2 to display the context. In order to conform with the path constraints in 3D space, the motion of the camera must be confined to the surface of the 2D-manifold shown in Fig 3 after applying g on the path. To find the appropriate path on this surface a criterion is required with which one path among the infinite number of paths is chosen. This is an application dependent condition and it varies from application to application. Let h be the path evaluation function defined as $h : \mathbb{R}^3 \mapsto \mathbb{N}^+$. The objective of this phase is to maximise $\int \int \int_V h dV$.

In the case of graph navigation, one suitable candidate for h can be the number of nodes the shaded area in Fig 2 swaps during the journey of the camera along the path. This method exploits the structure-from-motion perception cue (Colin Ware 2004) which may help the user to determine both the 3D structure of the surroundings and to identify or at least be aware of the large-scale layout of objects, in our case, nodes in space.

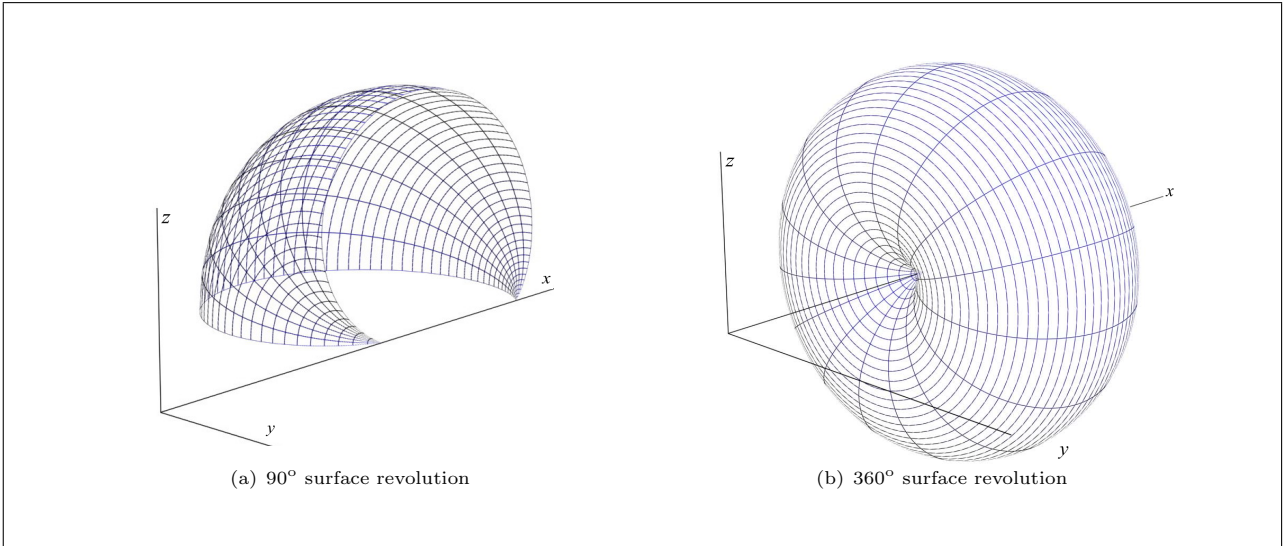


Figure 3: Possible paths on a 2D surface revolution manifold

Let us call this the *visibility* criterion. It is obvious that in this case h is bounded by the number of nodes in the graph. In this case the path that displays most number of nodes during the journey of the camera would be the one chosen by this second phase.

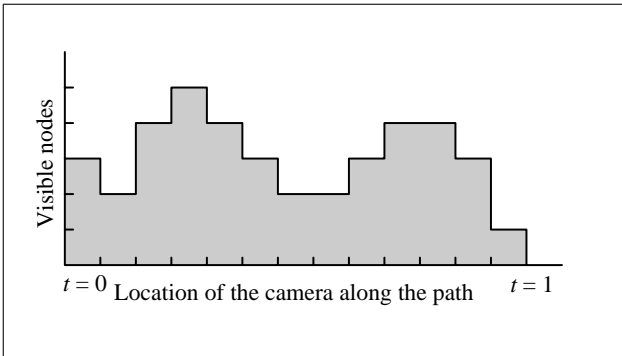


Figure 4: One instance of the visibility function \mathcal{F}_i for some i . The value t represents the position of the camera along the corresponding path C_i .

Another criterion that may seem significant to context awareness can be the *depth-of-view*, which is the virtual distance of the furthest visible object from the virtual camera at a certain time. Furthermore, to benefit from both of the above mentioned criteria, one may use $L_i = \lambda \mathcal{F}_i + (1 - \lambda) D_i$ to choose a suitable path C , where \mathcal{F}_i denotes the visibility function for some path i , D_i denotes the depth-of-view function of that same path, and $0 \leq \lambda \leq 1$. The value of λ depends on the nature of the application.

4.3 Implementation Remarks

Consider the graph navigation problem where only the graph nodes are the objects of interest. The graph edges do not carry any navigation importance. At a certain animation frame a graph node is either inside the camera view frustum or outside. In other words visible or not visible. the problem of automatic path calculation for navigating graphs in 3D becomes discrete in nature. Which makes the implementation fairly easy.

For the first phase, once the positions P_0 and P_1 are established, and since the camera target-point follows a straight line path, the values T_i for the camera target-point can be interpolated between P_0 and P_1 .

Then camera locations C_i along the path are computed. It is sufficient to compute the camera positions C_i for discrete values of $0 \leq \phi \leq \frac{13}{10}\pi$ then use these positions on the plane to defining a smooth spline curve, because $|P_0C| \approx 0$ when $\phi \approx \frac{13}{10}\pi$ and negative values of $|P_0C|$ are not significant to the task.

For the second phase, once a single camera path C_1 is calculated in the first phase, and considering the visibility function one can compute the set $C = \{C_2, C_2, \dots, C_k\}$ by simply rotating C_1 around the $P_0 - P_1$ axis in discrete steps and compute the corresponding \mathcal{F}_i s. Then it is just a matter of identifying the path $C_{optimal}$ which corresponds to $\max(\mathcal{F}_i)$ for $1 \leq i \leq k$.

As mentioned earlier the visibility function is discrete in nature. A typical plot of the visibility function is shown in Fig 4 where $0 \leq t \leq 1$ is the camera location parameter on a particular path. Not surprisingly, the plot for all criteria mentioned above are similar to the visibility due to the discrete nature of the graph navigation problem.

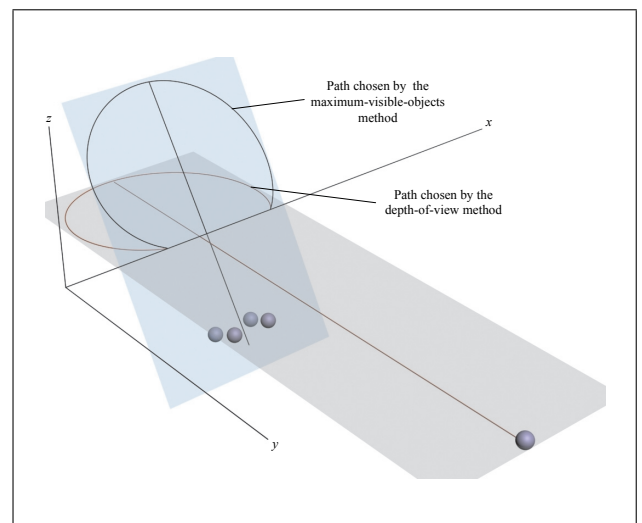


Figure 5: Two paths chosen by the two different context optimisation functions

5 Observations and Conclusions

A simple constrained navigation technique to navigate a graph visualisation in 3D virtual space was proposed. The technique involves moving the user's view port along a simple pre-computed path in 3D. An algorithm was given to generate the simple 3D camera path. The computed path can be used in assisted graph navigation tasks. Initial experiments indicated that by using the visibility function more objects were swapped by the virtual camera view frustum. In other words, more graph-structured-information was revealed. In the limited number of experiments that were carried out, the visibility function was preferred for navigation. Fig 5 explains the reason where two paths are shown. One selected by the visibility function and the other by the depth-of-view function. The straight line drawn from a path curve to an object indicates the direction of view. It is clear that the path chosen by the visibility function reveals more information about the structure of the neighbourhood than the path chosen by its counter part. Also, nodes that were away from the camera turned out to be less significant to the task in the experiments.

In extension to the Focus+Context concept discussed by Wijk *et al.* (Jarke J. van Wijk, Wim A. A. Nuij 2003), the camera navigation method proposed here combines the ability to focus on graph details and the process to build user awareness of the context. During the entire journey, the camera moves along the generated path while the camera target-point moves along a straight line from P_1 to P_0 as shown in Fig 2. Camera target-point motion is considered as focus transfer, whereas camera motion along the path contributes in building and enhancing the user cognitive mental map.

A full evaluation of NavAssist requires a user study. This is currently under the design stage. NavAssist is being planned to be implemented on GEOMI which will be used for evaluation experiments. GEOMI (GEOmetry for Maximum Insight) is a cutting edge new generation visual analysis tool that combines network visualisation techniques with network analysis methods. GEOMI is developed by NICTA IMAGEN VALACON research team for the visualisation and analysis of large and complex networks such as social and biological networks.

The navigation method presented here can be extended to applications other than graph navigation provided that a suitable evaluation function h is identified.

6 Acknowledgments

The authors would like to thank Nicola Nikolov for his valuable input by reviewing the final draft. They would also like to thank Tim Dwyer for proof reading the contents.

WilmaScope was used to find 3D layouts for the experimental graphs, see <http://www.wilmascope.org>.

References

A. J. Hanson, H. Ma (1995), 'Space walking.', *Proceedings of Visualization '95* pp. 126–133. IEEE Computer Society Press.

Andrew J. Hanson, Eric A. Wernert, Stephen B. Hughes (1999), 'Constrained navigation environment', *Scientific Visualization: Dagstuhl '97 Proceedings* pp. 95–104. 0-7695-0505-8.

Bowman, D., Koller, D., and Hodges, L.F. (1997), 'Travel in immersive virtual environments: an evaluation of viewpoint motion control techniques', *Proceedings of the 1997 Virtual Reality Annual International Symposium* pp. 45–52.

C. Ware, G. Franck (1996), 'Evaluating stereo and motion cues for visualizing information nets in three dimensions', *ACM Transactions on Graphics (TOG)* **15**(2), 121–140.

Carpendale, S. (1999), A Framework for Elastic Presentation Space, PhD thesis, School of Computing Science, Simon Fraser University.

Colin Ware (2004), *Information Visualization: Perception For Design*, second edition edn, Morgan Kaufmann.

Desney S. Tan, George G. Robertson, Mary Czerwinski (2001), 'Exploring 3D navigation: Combining speed-coupled flying with orbiting', *Proceedings of the SIGCHI conference on Human factors in computing systems* **3**, Issue No. 1.

Hix, D., E. Swan, J. Gabbard, M. McGee, J. Durbin and T. King (1999), 'User-centered design and evaluation of a real-time battlefield visualization virtual environment.', *IEEE Virtual Reality* pp. 96–103.

Jarke J. van Wijk, Wim A. A. Nuij (2003), 'Smooth and efficient zooming and panning', *Proceedings of IEEE 2003 Symposium on Information Visualization* pp. 15–22.

Melanie Tory, Torsten Möller, M. Stella Atkins, Arthur E. Kirkpatrick (2004), 'Combined 2D and 3D views for orientation and relative position tasks', *Proceedings of the 2004 conference on Human factors in computing systems* **6**(1), 73–80.

Nicholas Burtnyk, Azam Khan, George Fitzmaurice, Ravin Balakrishnan, Gordon Kurtenbach (2002), 'StyleCam: Interactive stylized 3D navigation using intergrated spatial & temporal controls', *Proceedings of the 15th annual ACM symposium on User interface software and technology*.

Richard J Webber (1998), Finding the Best Viewpoints for Three-Dimensional Graph Drawings, PhD thesis, The Department of Computer Science and Software Engineering, The University of Newcastle, Australia.

S. Card, G. G. Robertson & J. Mackinlay (1991), 'The information visualizer, an information workspace', *Proceedings of the SIGCHI conference on Human factors in computing systems: Reaching through technology* pp. 181–188. ACM Press / Addison wesley.

Springmeyer, R. R., Blattner, M. M., Max, N. L. (1992), 'A characterization of the scientific data analysis process.', *Proceedings of the 3rd conference on Visualization '92* pp. 235–242.

Stanislav L. Stoev, Wolfgang Starßer (2002), 'A case study on automatic camera placement and motion for visualizing historical data', *IEEE Visualization* pp. 545–548.

Stephen Hughes, Joseph Manojlovich, Micheal Lewis, Jeffrey Gennari (2003), 'Camera control and decoupled motion for teleportation', *IEEE*. 0-7803-7952-7/03.

- Stephen Hughes, Michael Lewis (2002), 'Attentive interaction techniques for searching virtual environments', *Proceedings of the Human Factor and Ergonomics Society* pp. 2159–2163.
- T. J. Jankun-Kelly, Kwan-Liu Ma (2003), 'Moire-Graphs: Radial focus+context visualization and interaction for graph with visual nodes', *IEEE Symposium on Information Visualization* **0-7803-8154-8/03**, 59–64.
- Y. Xiao, P. Milgram, A.G. Ryman (1993), 'Implementation and evaluation scheme for stereoscopic display of networks', *IBM Canada Laboratory Tech. Rep.* **74.126**.