MULTI-RESOLUTION VISUALIZATION OF GEOGRAPHIC NETWORK TRAFFIC

By

BERKAY KAYA

Submitted to the Graduate School of Engineering and Natural Sciences

in partial fulfillment of the requirements for the degree of

Master of Science

SABANCI UNIVERSITY

Spring 2009

MULTI-RESOLUTION VISUALIZATION OF GEOGRAPHIC NETWORK TRAFFIC

APPROVED BY:

Assist. Prof. Dr. Selim Balcisoy ________________________

(Dissertation Advisor)

Assoc. Prof. Dr. Albert Levi ________________________

Assist. Prof. Dr. Burçin Bozkaya ________________________

Elif Ayıter ________________________

Assist. Prof. Dr. Yücel Saygın ________________________

DATE OF APPROVAL: ________________________

MULTI-RESOLUTION VISUALIZATION OF GEOGRAPHIC NETWORK TRAFFIC

Berkay Kaya

EECS, M.Sc. Thesis, 2009

Thesis Supervisor: Assist. Prof. Dr. Selim Balcisoy

Keywords: Geographic Visualization, Network Visualization, Flow Visualization, Level of Detail, Usability Testing

Abstract

Flow visualization techniques are vastly used to visualize scientific data among many fields including meteorology, computational fluid dynamics, medical visualization and aerodynamics. In this thesis, we employ flow visualization techniques in conjunction with conventional network visualization methods to represent geographic network traffic data. The proposed visualization system integrates two visualization techniques, flow visualization and node-link diagram, in a level of detail framework. While flow visualization emphasizes general trends, node-link diagram visualization concentrates on the detailed analysis of the data. Usability studies are performed to evaluate the success of our approach.

COĞRAFİ AĞ VERİLERİNİN ÇOKLU ÇÖZÜNÜRLÜKLERDE GÖRSELLEŞTİRMESİ

Berkay Kaya

EECS, Yüksek Lisans Tezi, 2009

Tez Danışmanı: Yar. Doç. Dr. Selim Balcisoy

Anahtar Kelimeler: Coğrafi Bilgi Görselleştirmesi, Ağ Görselleştirmesi, Akış Görselleştirmesi, Detay Seviyeleri, Kullanılabilirlik Çalışmaları

Özet

Akış görselleştirme teknikleri yaygın olarak; meteoroji, hesaplamalı akışkanlar dinamiği, medikal görselleştirme ve aerodinamik gibi bilimsel verileri görselleştirmekte kullanılır. Bu tezde, akış görselleştirme tekniklerini ağ görselleştirme yöntemlerinden de yararlanarak coğrafi ağ trafiği verilerini görselleştirmek için uyguladık. Öne sürülen görselleştirme sistemi, detay seviyeleri tekniği kapsamında akış görselleştirmesi ve nokta-bağlantı diyagramı tekniklerini entegre etmektedir. Kullanılan akış görselleştirmesi tekniği genel eğilimleri vurgularken, nokta-bağlantı diyagramı görselleştirmesi verinin detaylı analizi üzerine yoğunlaşır. Yaklaşımımızın başarısı, kullanılabilirlik çalışmaları yardımıyla değerlendirilmiştir.

ACKNOWLEDGEMENTS

I would like to express my gratitude to my advisor Selim Balcısoy for his support and guidance during my MSc. period. His thoughtful feedbacks, valuable suggestions and infinite support, have significantly enhanced my thought processes.

I would like to thank all my friends and colleagues, particularly to Çagatay Turkay and Uraz Cengiz Türker, in the Computer Graphics Lab for their friendship and assistance.

I am grateful to my thesis committee members Albert Levi, Burçin Bozkaya, Elif Ayıter, and Yücel Saygın for their valuable review and comments on the dissertation.

I would like to thank to my precious family for their unconditional love and support that made everything possible for me.

Finally, I would like to thank Berfin Dinler for encouraging me at all my times of hesitation. I am also grateful to Burak Karaboğa for his continuous support, especially during my last two weeks.

I also thankfully acknowledge the financial support provided by the Scientific and Technological Research Council of Turkey (TÜBİTAK).

TABLE OF CONTENTS

LIST OF FIGURES

TABLE OF ABBREVIATIONS

LIC	Line Integral Convolution
UFLIC	Unsteady Flow Line Integral Convolution
LOD	Level Of Detail

1. INTRODUCTION

1.1. Information Visualization

The developments in hardware technology made computers able to store tremendous amounts of data. Gathered data is generally collected automatically by monitor systems or sensors. Even some casual daily transactions such as telephone calls or electronic mails are recorded by personal computers. The importance of the recorded data is generally not taken into consideration. Every data is thought as a potential source of valuable information [1]. However, extracting useful patterns and exploring important details in the data is impossible by just reading it in a textual form. The need for information visualization emerges in this step. Information visualization satisfies this need by offering the data in various forms with differing interactions. Information visualization aims to provide graphical representations and user interfaces for interactively exploring large sets of items [2]. A visualization can give an overview of the data, it can filter the data according to user's needs, it can summarize data and help identify important patterns [3].

Exploration of information collections becomes extremely difficult as the volume of data grows. A page of information can be easy to analyze; however, when the information becomes the size of a book or even larger, exploring the information may become a challenging task [4]. As more data are gathered, the need for extracting useful information increases. While data mining aims to uncover hidden patterns and useful information from the raw data, involving human factor into this process can also yield very promising results. Visual data exploration aims at integrating human factor in the data analysis process, applying human perception to gather important information from the large data sets. [1] In the basis of visual data exploration, data are represented in visual form allowing the user to obtain important features from the data. Therefore, the user should be able to directly interact with data. The process of visual data exploration follows three steps, which has been called Information Seeking Mantra in Shneiderman's paper [4]; overview first, zoom and filter, and details on demand.

In today's digital world, one of the most elegantly collected data is network traffic data. Some typical examples of network traffic data are telephone calls, electronic mails, flight traffic, democratic relations, etc. Networks are generally based on simple graphs which are formed by the connecting lines that represent binary connections between nodes. In binary connections, the properties of the relationships between the members are disregarded. Ignoring the important features of the relationships can lead to reduction of potentially valuable information. Traffic density is an appropriate example of a quantitative value that can be assigned to each relationship. Encoding every connection with traffic density value is crucial in analyzing the important features of the network. In a geographic network, node positions are in key importance. Hence, layout of the network should be stable, so that geographic properties are not lost.

The proposed visualization is a multi-resolution visualization of geographic network traffic, offering different visualization techniques in different levels of detail. Edges between nodes are encoded with visual parameters to represent traffic density.

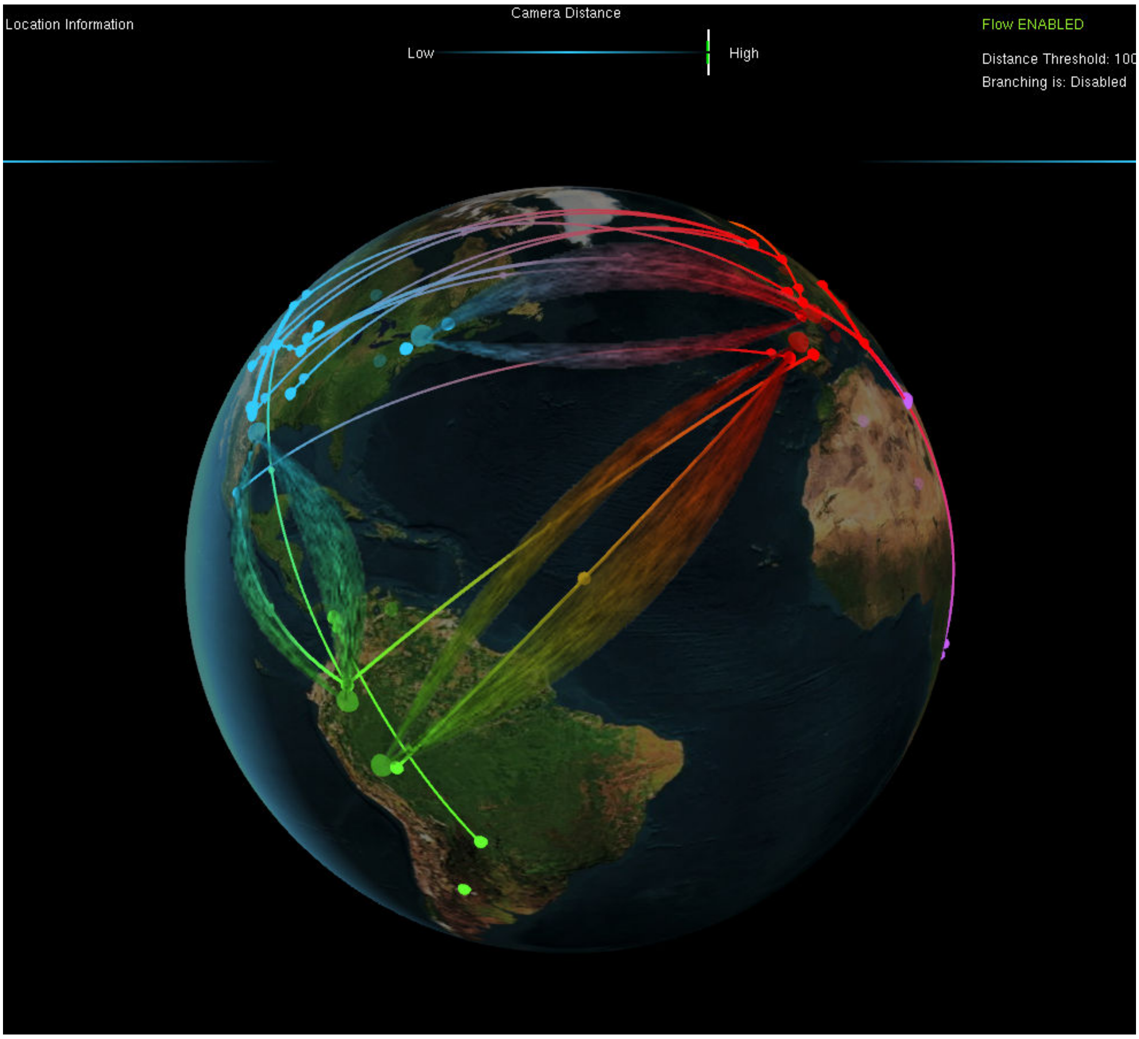
1.2. Problem Definition

Visualizing geographic network traffic is a challenging task for several reasons. In geographic networks, the positions of the nodes play an important role. Although dealing with geographic data on 2D displays seems a trivial process, there are many problematic issues. If the number of records is too high, visualization gets too cluttered due to the limited pixel count of the displays. The cluttered view yields to misleading results, thus clear views should be supplied.

Supporting both global and local views is another significant challenge. Users should be able to comment on global trends while working on the overview mode. On the other hand, specific details must be detected when detailed view is enabled. Likewise, users should be able to examine regions with high level of detail. In brief, successful visualizations must allow users to obtain important, insightful patterns from an overview as well as to investigate the details of each node and links [5].

To clarify the problem definition, the network traffic should be visualized in such a way that, geographic properties are not lost. Moreover, supporting both global and local views is mandatory for data analysis. It becomes easier to detect global patterns in the large scale layout, whereas individual traffic information can only be examined in the detailed view. Therefore, level of detail should be navigable through the program by interactive techniques like zooming.

In this thesis, we employed two visualization techniques, node-link diagrams and flow visualization, in the context of geographic network visualization. We integrated flow visualization techniques to network visualization problem. Our visualization system is based on level of detail framework, which is controlled by the camera distance. In the detailed view, network traffic is visualized by node-link diagrams. In the overview, visualization system benefits from the flow visualization techniques to represent high density traffic. Finally, we evaluated the success of our visualization system with usability testing.

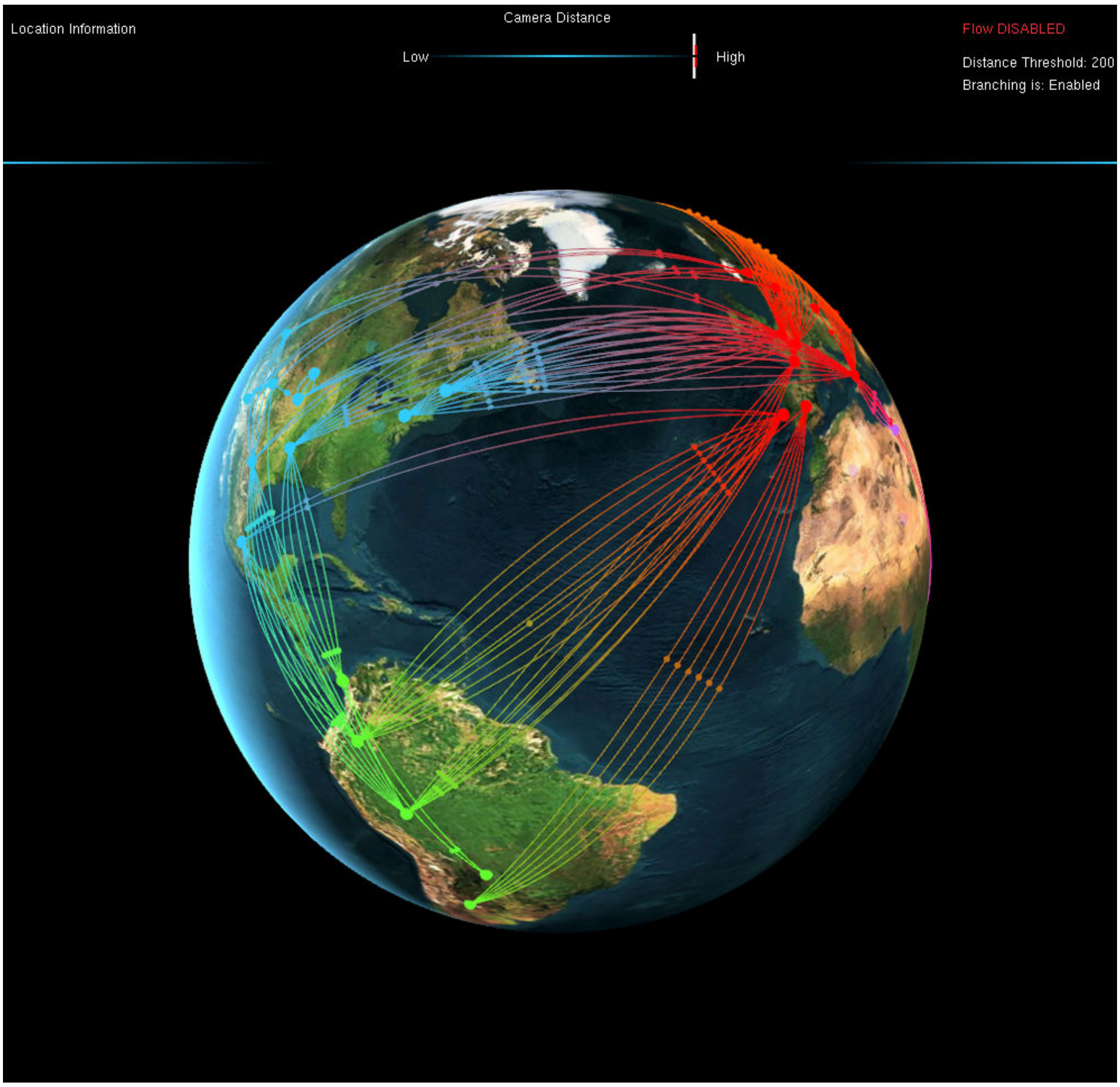


Flow ENABLED

Distance Threshold: 10

Branching is: Disabled

Figure 1.1 Proposed visualization when flow visualization is integrated



Flow DISABLED

Distance Threshold: 200

Branching is: Enabled

Figure 1.2 Proposed visualization when node-link visualization is enabled

1.3. Summary of Contributions

The main contributions of this research can be classified into two categories:

- Integration of flow visualization techniques in the geographic network visualization context.
- Combination of two different visualization techniques, while performing level of detail.

We extended our research by doing usability tests to discuss the success of the proposed visualization.

1.4. Thesis Outline

This chapter briefly focuses on the importance of information visualization in data analysis. The benefits of visual data exploration are discussed in conjunction with the challenges in geographic network visualization. Moreover, a summary of the proposed visualization and the main aspects of the system are given. Finally, it concludes with a listing of the summary of contributions.

Motivation And Related Work – The second chapter gives a background on network visualization and flow visualization. Moreover, an overview of geographic visualization techniques is presented. Level of detail framework is also briefly discussed.

Flow Visualization for Geographic Networks – The third chapter concentrates on the proposed flow visualization technique and its application to geographic network visualization phenomenon.

Level of Detail for Geographic Networks – This chapter briefly focuses on the level of detail approach studied. Besides, it analyzes the node-link diagram visualization technique used in the visualization system.

Experiments and Usability Studies – The fifth chapter demonstrates the experiments designed. Moreover, usability studies are analyzed.

Discussions – In this chapter, the results from the evaluation of usability studies are discussed in detail. It also elaborates on the success of the proposed flow visualization technique.

Conclusions – Finally, an overall discussion of the research is presented in this chapter.

2. MOTIVATION AND RELATED WORK

2.1 Stages in Visualization

Data visualization process is based on four stages. These four stages are as follows:

- The gathering of the data itself
- Preprocessing of the data into something that we can understand
- The display hardware and visualization algorithms that create an image to the screen
- Human perceptual and cognitive system [6]

Collection of the data is the first step in designing a visualization system. The data type to be visualized plays the most important role in the design process of a visualization system. Each record of data consists of a number of variables. Moreover, each record corresponds to an observation, a relationship or a measurement. For instance, network traffic data can be thought as a collection of relationships, whereas sensor output from physical experiments can be regarded as observation or measurement [1]. The dimensionality of the dataset can be defined by the number of variables. Data sets can be one-dimensional, two-dimensional, multi-dimensional or complex structures like graphs. It is a common strategy to denote network relations with node-link diagrams; however, using node-link diagrams for temporal one-dimensional data sets like time series of stock prices may not be appropriate. Thus, the convenient visualization technique can only be determined, if the analysis of the dataset is completed.

2.2. Network Visualization

Networks have been recognized as effective tools for visualizing datasets involving relationships. Networks are widely used in computer science to illustrate data structures, object relationships, hardware architecture, control flow and data flow [7]. Visualization of large data sets including relationships is in growing need in many fields such as social networks, telecommunication, and internet networks [8]. Graphs are the most common visualization element for representing networks. As data sets become larger, graph structures become more complex. Depending on the structure of the network, there are different types of visualization techniques to represent graphs, which are discussed below.

Node-link diagrams are the most popular method for visualizing networks in general. In the recent years, a high percentage of network visualization papers are based on node-link diagram representations [9]. Many networks can be applied to node-link diagrams. For instance, in social networks, nodes (i.e. vertices) represent actors and links (i.e. edges) represent relationships. While in geographic networks, nodes represent locations and links represent traffic.

Network layout is a fundamental point in the network visualization. For a visualization method to convey information as effectively as possible, a good layout is crucial. The literature on network layout has been dominated by force-directed strategies because they produce clever spreading of nodes and reasonable visibility of links [5]. Nodes are laid out as if there were electrical forces between them, where links determine the attraction between connected nodes. Eades et al. [10] proposed this idea; however, the most common reference is to Fruchterman-Reingod algorithm [11]. Variations are called spring-embedding to describe the connections between connected pair of nodes. [12][13]

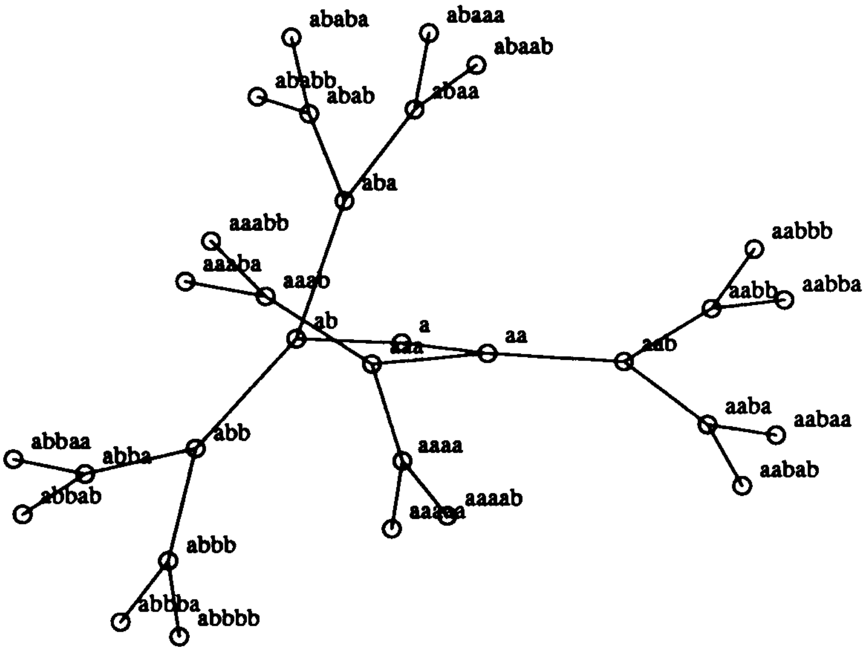


Figure 2.1 Force-directed graph (Spring-embedding) [12]

A second common layout strategy, which generates familiar and comprehensible layouts, uses geographical maps, in which the node locations are fixed [14]. Edges are drawn between locations denoting the relationships between the entities.

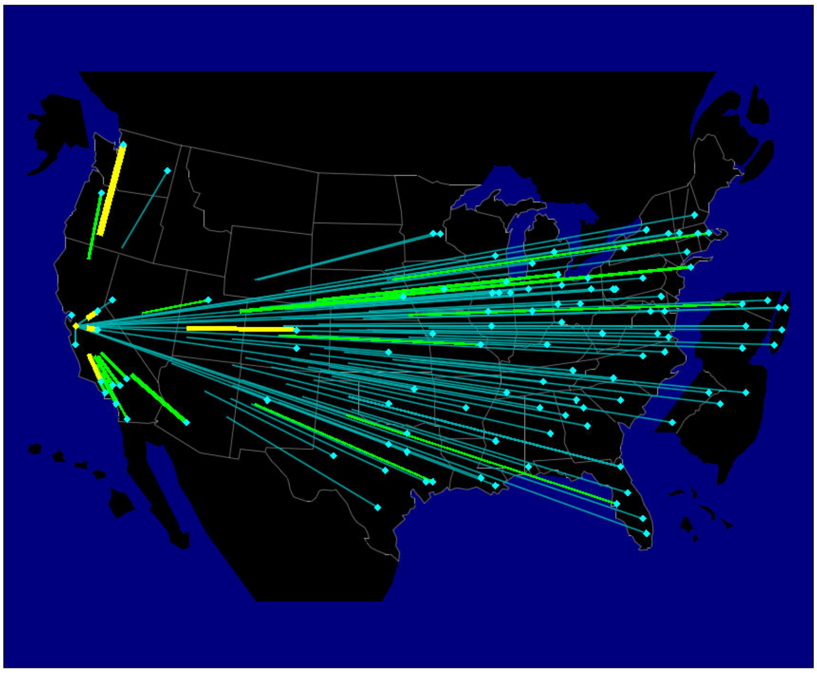


Figure 2.2 Linkmap visualization. Links denote relationships between nodes. [14]

A third strategy uses circular layout for nodes. This visualization places nodes in a circular layout and it produces an elegant presentation with crisscrossing lines through the center of the circle [15][16].

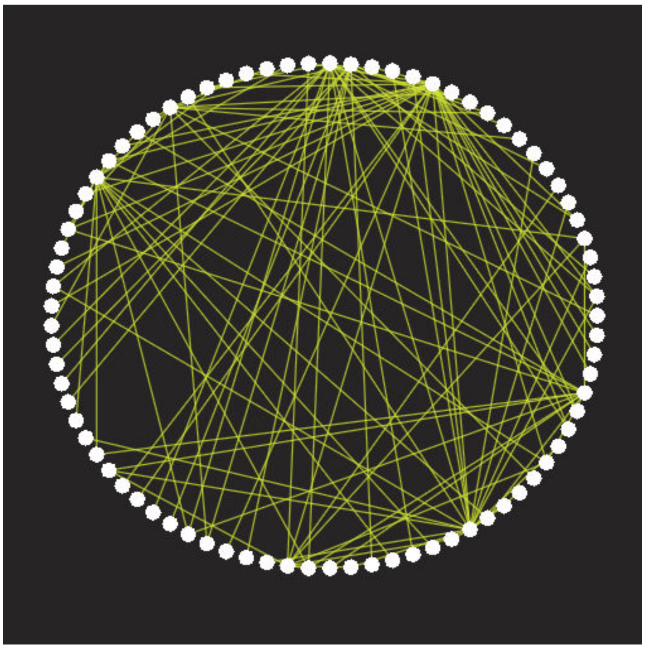
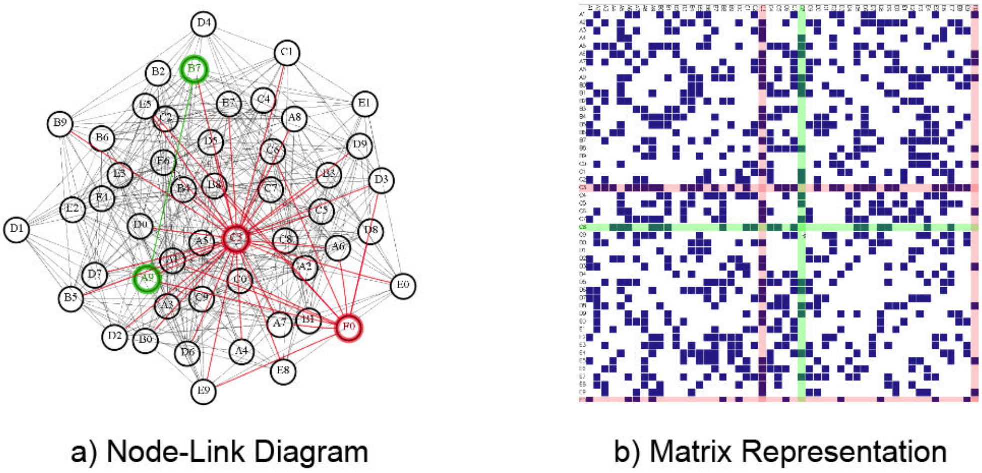


Figure 2.3 Circular Layout [15]

Another strategy uses matrix-based representations instead of node-link diagrams [17]. In this representation, some problems like edge crossings, node occlusions are avoided. However, spatial characteristics become harder to perceive, such as finding nodes on a path and identifying clusters.



a) Node-Link Diagram

b) Matrix Representation

Figure 2.4 Two visualization methods representing the same dataset. Node-link diagram on the left, matrix representation on the right [17]

To sum up, network visualization requires a deep analysis of the data to be visualized. Depending on the data type, network layout should be decided in a proper way to express important features of the data. In geographic networks, node positions are important, therefore, layout cannot be altered in a straightforward fashion. Small changes in the layout can be tolerable if they don't significantly perturb the perception of relationships.

2.3. Flow Visualization

As a subfield of scientific visualization, flow visualization is one of the popular fields in information visualization. Flow visualization covers a variety of applications such as weather simulation, meteorology, computational fluid dynamics and medical visualization [18]. The coverage of flow visualization solutions spans multiple technical challenges like 2D vs. 3D solutions. The data type also plays an important role on designing the flow visualization application. Visualization of time-dependent unsteady data may face different challenges than steady data representations. The remainder of the literature survey will expand flow visualization in two sections, direct flow visualization and dense texture-based flow visualization.

2.3.1. Direct Flow Visualization

This subset of visualization techniques uses a translation of data that is as direct as possible. The resulting image is generally an overall picture of the data [18]. Commonly, flow visualization methods are based on 2D or 3D vector fields defining the characteristics of the data. The techniques examined in this section are the studies made on vector field visualizations.

Arrow Plots

Arrows plots are the most fundamental visualization for 2D vector fields. In this visualization, arrows are representative icons for vectors. This visualization is also called "hedgehog" or "hedgehog plot" [19]. Instead of using arrows, other glyphs that could represent direction of the vectors in a vector field can be used.

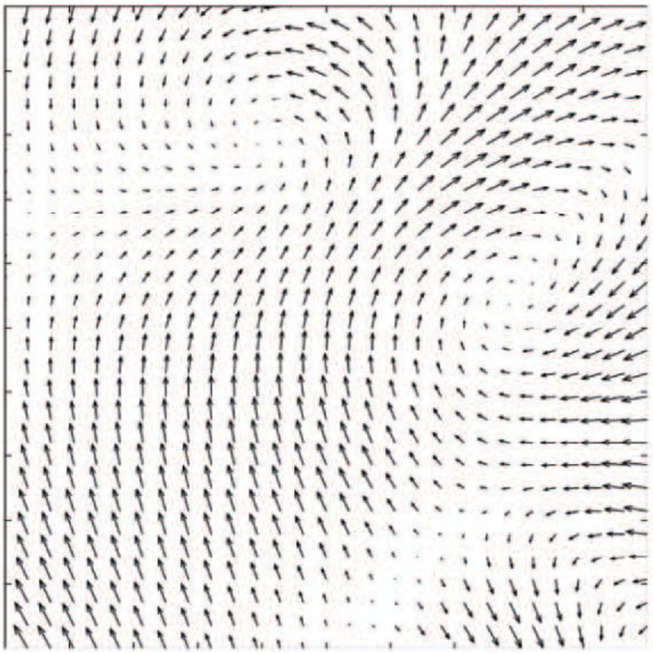


Figure 2.5 Arrow Plot Visualization. Each arrow represents a vector in the 2D vector field [21]

There exist some variations of the arrow plot visualization. Study done by Dippe et al. [20] places icons on a jittered grid, in which, critical points of the vector field can be depicted easily.

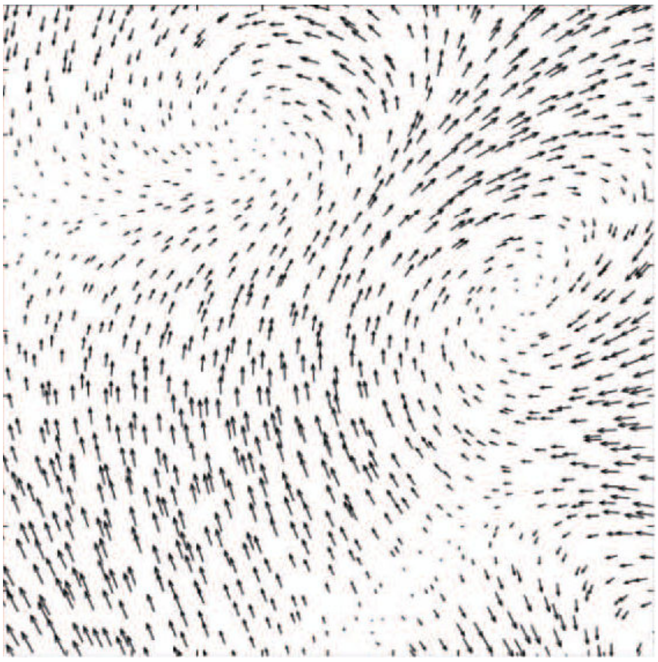


Figure 2.6 Icons placed on a jitter grid. Image courtesy of [21]

The survey by Laidlaw et al. [21] expands the study of Kirby et al., which places icons using one layer of a visualization method that is based on concepts borrowed from oil painting [22]. Triangle-shaped wedges are used in the example. Strokes are carried out with regard to the spaces between wedges to maintain a clear image. Wedges that don’t satisfy the spacing property are not drawn.

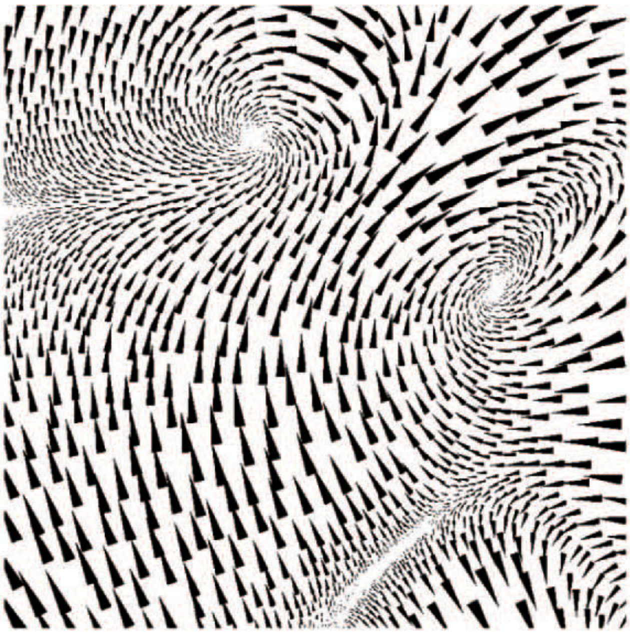


Figure 2.7 Study of Kirby et al. [22]

Streamlines

A streamline is a smooth curvature through the vector field at a given time. Streamlines are tangent to the vector field in every place. There are a variety of representations in both 2D and 3D vector field visualizations. Turk and Banks [23] studied image-guided streamline placement for representing streamlines based on 2D vector fields.

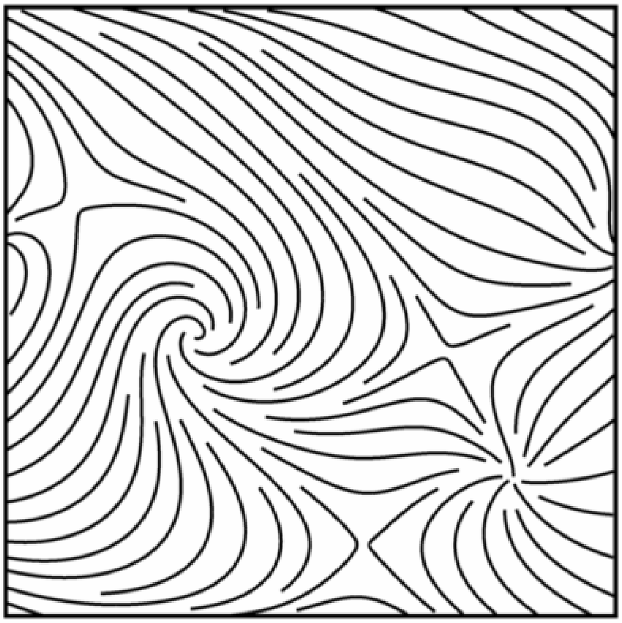


Figure 2.8 Streamline visualization of randomly generated 2D vector field [23].

Another research done by Jobard et al. is applied to unsteady flow data. The movement of the flow is handled by moving a one-dimensional texture along the streamlines [24]. In order to satisfy smooth animation, consecutive frames are correlated. This approach gives full control on the density of the representation. In high density settings, this visualization can produce results similar to dense texture-based flow visualizations. In the following figure, three consecutive frames of a time-varying flow data are visualized.

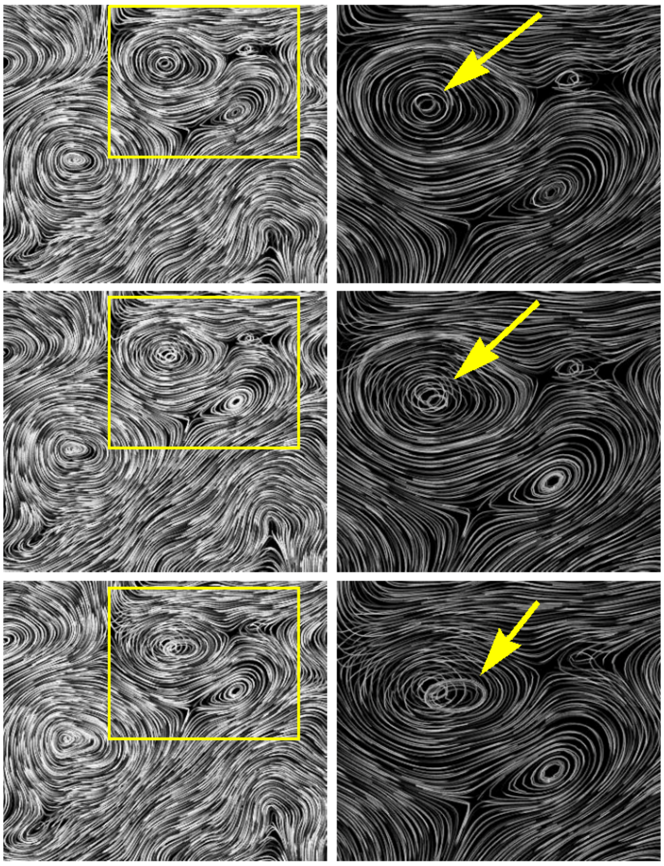


Figure 2.9 Unsteady flow data visualization using streamlines Jobard et al. [24]

A 3D vector field visualization that is based on streamlines is studied by Fuhrmann and Gröller. They implement 3D streamlines with animated opacity textures, called dashtubes. [25]

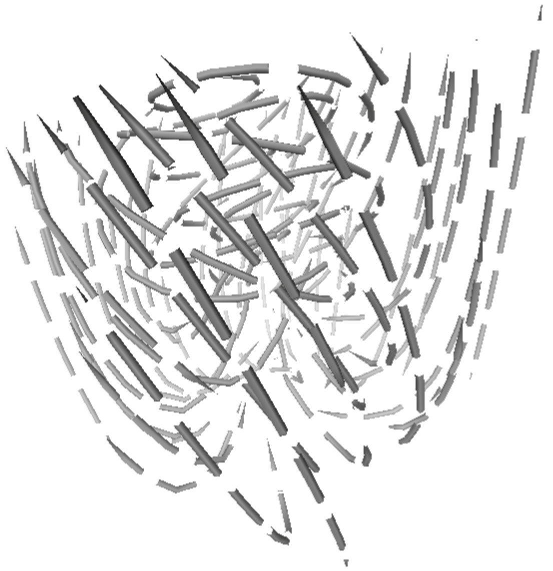


Figure 2.10 Dashtubes by Fuhrmann et al. [25]

Stream ribbons and stream surfaces

Stream ribbons are based on the concept of a streamline that is extended to a surface. If two adjacent stream lines are connected by many polygons, a stream ribbon emerges [26]. A stream surface is the two-dimensional locus of an advected initial seed curve. In other words, the sheet is the locus of all streamlines with a seed curve [27].

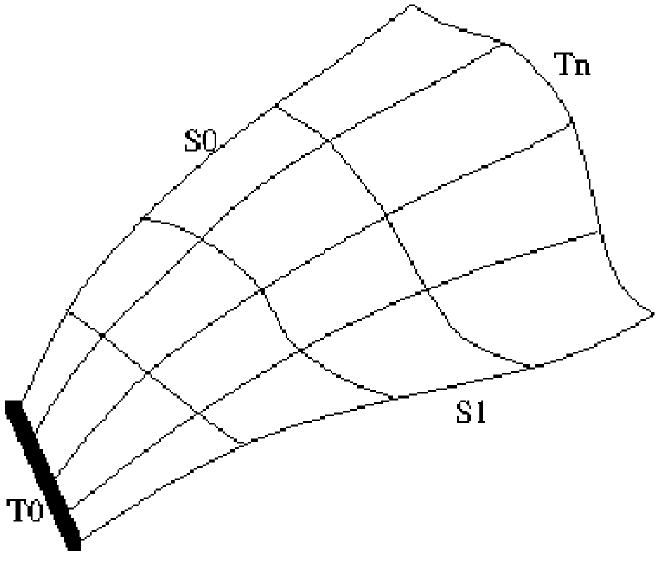


Figure 2.11 Stream surface visualization [27]

2.3.2. Dense Texture-Based Flow Visualization

This category of visualizations is built on the concept of computation of texture values to generate a dense representation of the flow [18]. Flow motion is incorporated through the related texture values along the vector field. These methods provide full spatial coverage of the vector field.

Spot Noise Based Methods

Spot noise, first introduced by Van Wijk [28], is one of the first dense texture based methods in the flow visualization literature. The resulting texture is generated by distribution of a set of intensity functions called spots. These spots represent a particle that warps over a small step in time. After a period of time, these particles form a streak in the direction of the local streamline.

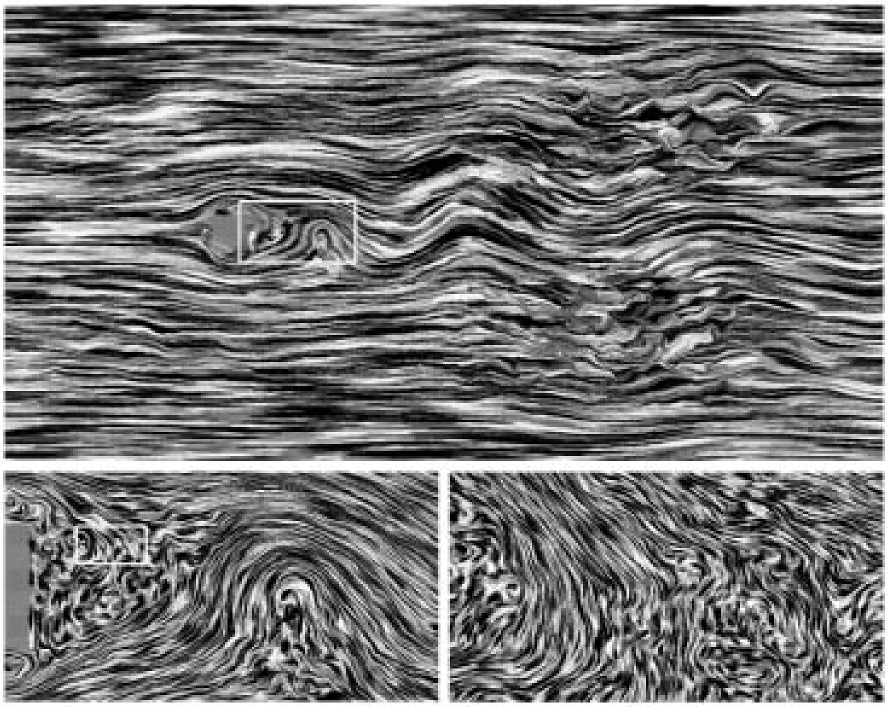


Figure 2.12 Spot noise algorithm. [29]

For further details on spot noise based methods, a state of the art report can be found in [18].

Line Integral Convolution

First introduced by Cabral and Leedom [30], a large community of flow visualization researchers dedicated themselves for developing better methods. The figure below illustrates the developments made in visualization techniques based on line integral convolution (LIC).

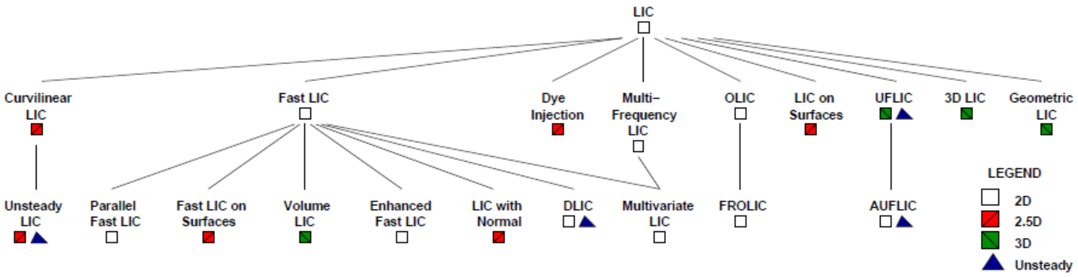


Figure 2.13 Research fields based on LIC [18]

In the original LIC algorithm a white noise image is taken as input. Texels are convolved along the path of streamlines. The convolution process is actually a low-pass filter, which creates smoothened textures. In order to create a dense representation, post-processing steps like high-pass filtering is used.

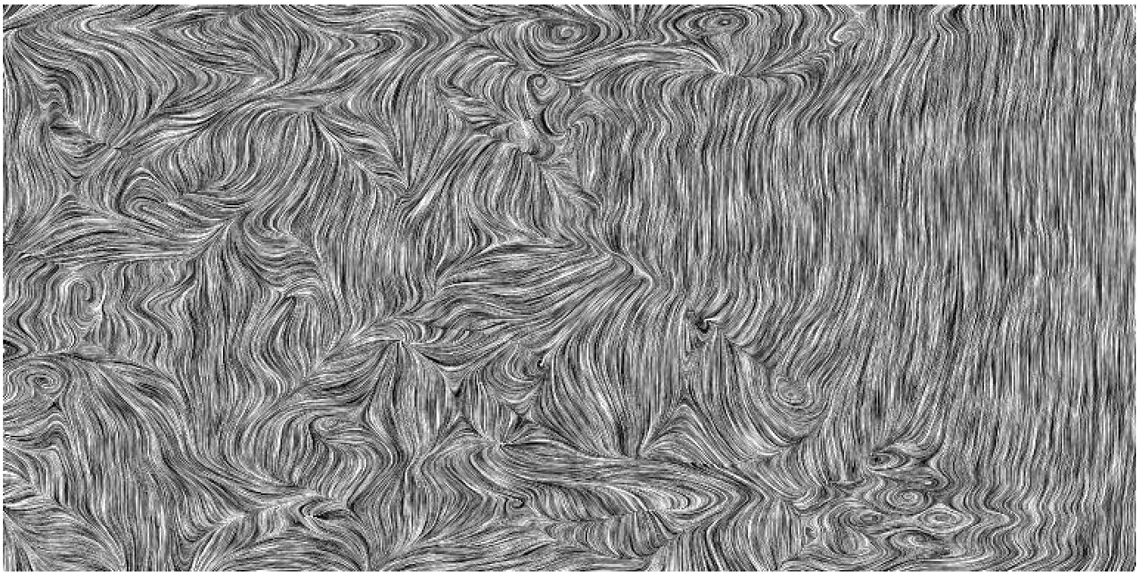


Figure 2.14 LIC output of a white noise texture convolved with fluid dynamics vector field [30]

Several extensions are developed in several directions such as extending LIC to 3D, applying LIC to unsteady flow data, improving the performance of LIC to real time and etc. These extensions will not be covered in detail. Details can be found in the survey paper published on dense texture-based flow visualization by Laramee et al. [18]

In the design process of our flow visualization technique, we reviewed several constraints in network visualization such as traffic direction and density. In order to represent traffic direction, providing an understandable flow animation was crucial. Furthermore, dense representations should be preferred in order to highlight traffic density. With these motivations, we decided on building our algorithm on dense-texture based methods. The algorithm used for generating flow textures in our approach is mainly based on UFLIC (Unsteady Flow Line Integral Convolution). UFLIC maintains the coherence of flow animation by successfully updating convolution results. UFLIC uses LIC as the underlying approach. Therefore, LIC and UFLIC are covered in detail in the third chapter, Flow Visualization for Geographic Networks.

2.4. Level of Detail

Keeping an overview of the entire data while allowing users to discover and analyze patterns in detail is a fundamental challenge in information visualization. There are several premises to address this problem. First of all, the user needs both views at the same in a simultaneous way. In other words, the user should be able to switch contexts by his/her will and the application should serve both views. Secondly, information needed in the overview side could be different from the detailed view. In addition to these principles, global and local views may be combined in a single display [31]. Displaying every single item in the global view is costly and unnecessary. Level of detail algorithms attempt to provide meaningful views in both contexts in an efficient manner, while satisfying multi-resolution property.

Level of detail discipline, LOD for short, attempts to balance complexity and performance by managing the amount of detail presented in the visualization [32]. The complexity of 3D models seems to grow faster than the ability of the hardware. In addition to 3D models, terrain visualizations also benefit from LOD techniques. One example of level of detail study on terrain visualization is implemented by Engin et al. [33].

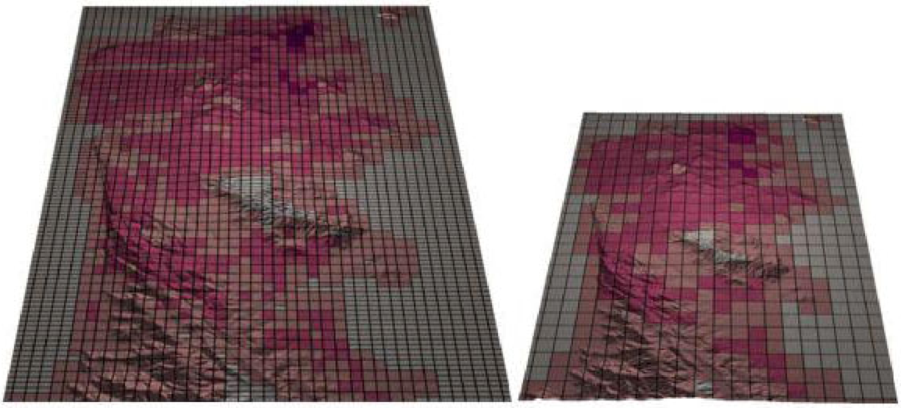


Figure 2.15 Terrains with different level of detail and their subdivisions [33]

They present a system that employs level of detail for data abstraction with respect to camera movements. They introduce automated details-on-demand feature to thematic maps allowing the user to extract detail information about his/her interest.

The fundamental concept of LOD is depicted in Figure 2.16.

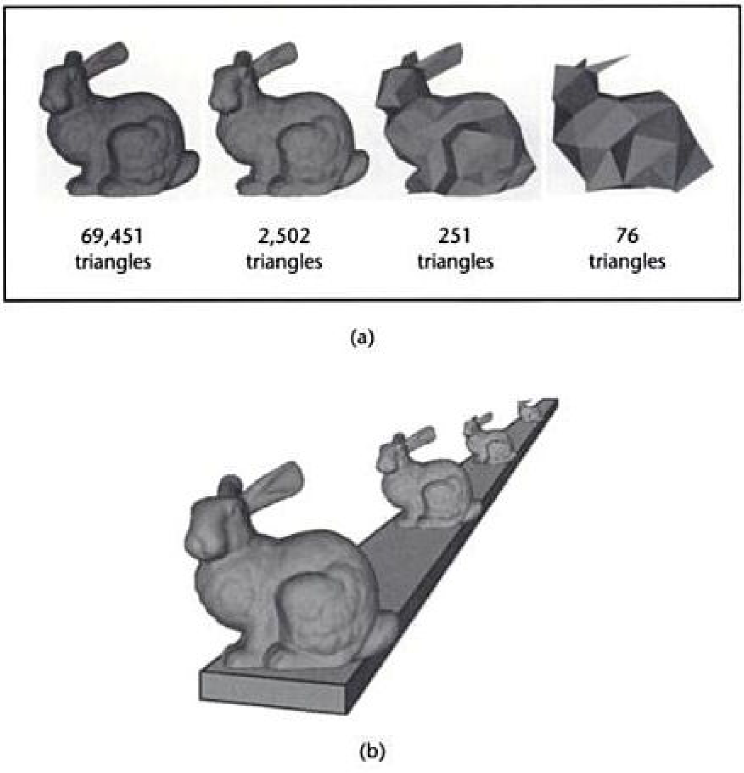


Figure 2.16 LOD applied to meshes via polygon reduction [32]

In this process, a complex object is simplified in terms of number of polygons, generating different levels of detail. With regard to the camera distance, object with the corresponding level of detail is drawn. Performance is increased by eliminating unimportant geometry. Controlling the detail level as the viewing parameters change, is referred to view-dependent level of detail control [34].

In network visualization problem, level of detail can be maintained by regulating the underlying graph structure. In the thesis, level of detail is controlled by the camera distance. When camera distance reaches to predefined threshold value, edges carrying high density traffic are clustered into flows and individual links are hidden. Flow locations that emerge from clustered edges are integrated to the current graph structure; however, cluster forming edges and corresponding nodes are removed. As a result, graph structure is morphed. Apart from the changes in graph structure, visualization technique is also modified by employing flow visualization in the existing node-link diagram visualization.

2.5. Geographic Visualization

Geographic visualization is a sub-field of information visualization in which principles from cartography, geographic information systems (GIS), Exploratory Data Analysis (EDA) are integrated in the development and assessment of visual methods that facilitate the exploration and analysis of the geography referenced information. [35] The datasets having geospatial components are relevant to large number of applications such as weather measurements, telephone networks, use of connecting nodes in internet backbone, air pollution in cities etc [1]. Nowadays, with the developments in the communication technology like the increased trends in mobile applications and extended use of GPS, geographic properties are included in almost every dataset. The visualization of geospatial data is clear-cut.

There are several approaches that deal with geospatial data. One widely used method visualizes data points on map regions. There are various commercial products based on this method like ArcView [36] and ADVIZOR [37]. In addition to placement of data on map regions, more information can be merged via visual components like glyphs or bars.

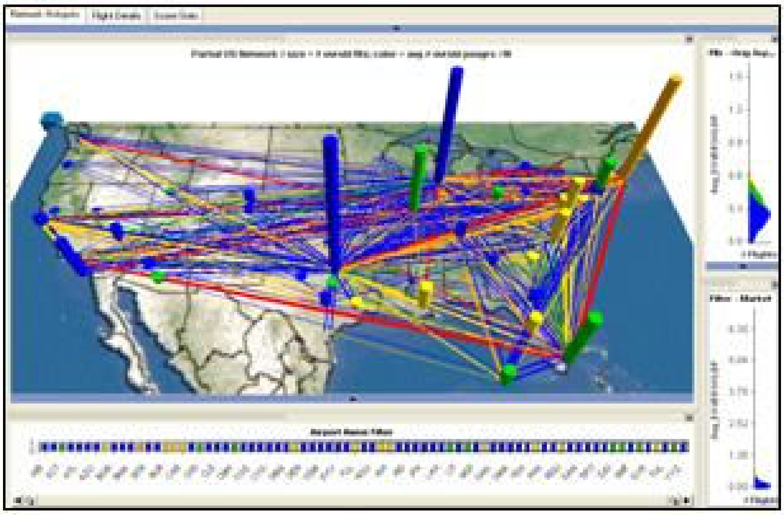


Figure 2.17 Visualization of geospatial data points in ADVIZOR [37]

Munzner et al. studied visualization of MBone, Internet's multicast backbone, by drawing arcs over the globe [38].

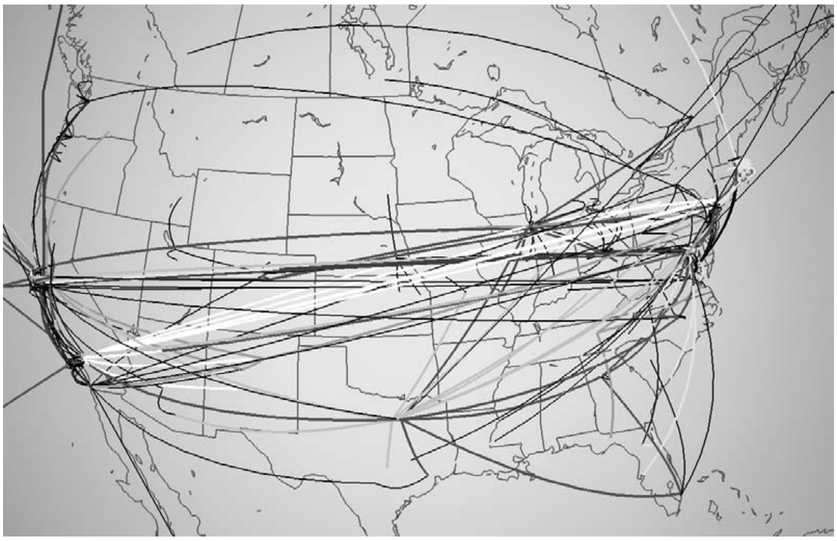


Figure 2.18 Munzner's visualization of MBone. Arcs denoting links between locations [38]

Thematic maps are also widely used in the geographic visualization field. A thematic map is used to display the spatial pattern of a theme. They emphasize one or more attributes like population density or family income, regarding geographic information.

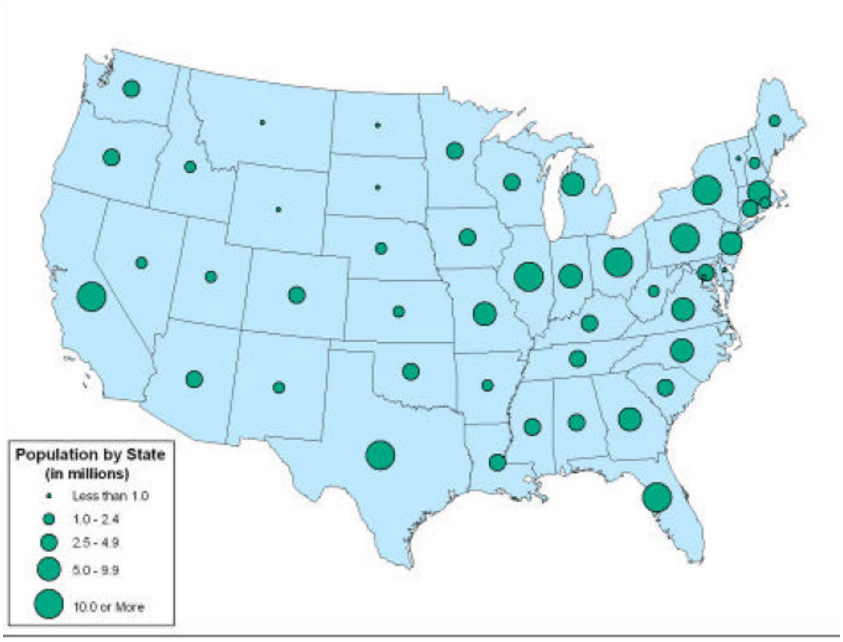


Figure 2.19 Population in USA by States. [39]

Cartograms are map visualizations that are based on the motivation of distorting the map according to the thematic mapping variable. Distortion of the map is handled in such a way that the geography is preserved. Area cartograms are commonly used visualizations for geographic distribution of a variety of disciplines involving social demographics, epidemiology and business. Areas of the map regions are scaled to represent the data. [40] A good example would be Worldmapper Project. Territories are resized corresponding to the subject of interest. [41]

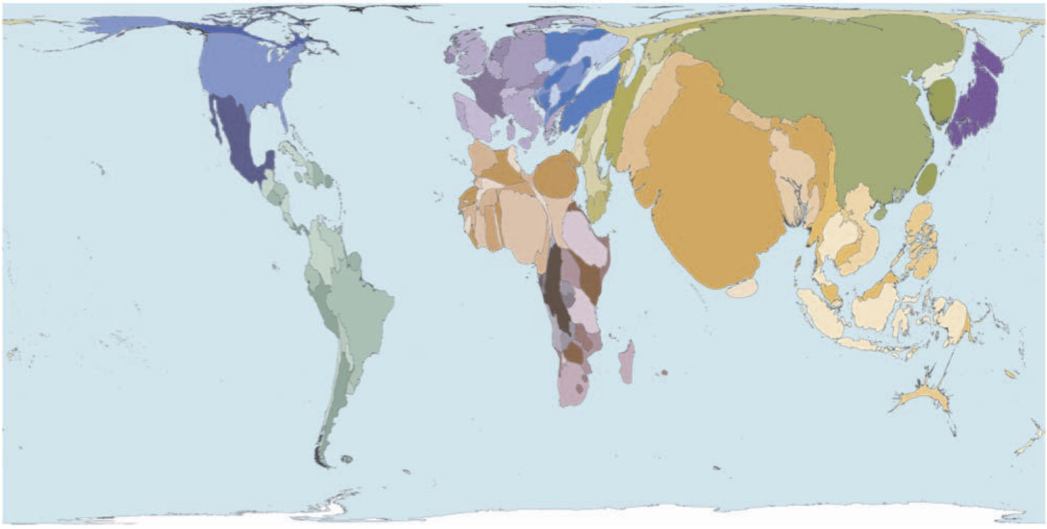


Figure 2.20 Visualization of World Population in Worldmapper [41]

3. FLOW VISUALIZATION FOR GEOGRAPHIC NETWORKS

The proposed visualization system in this study is based on two different visualization techniques. The first approach, which is a variant of node-link diagrams applied to geographic networks, achieves high level of detail. In this view, each relationship in the geographic network is represented as an individual edge. The second approach incorporates flow visualization into network visualization framework when level of detail is low. The regions having very high density traffic are represented as flow textures. In order to make clearer textures, transparency of the globe is decreased to 50%.

As discussed in Chapter 2.3, flow visualization approaches are built upon 2D vector fields. We claim that it is not convenient to involve every pixel in the flow area. In the case of network visualization, some nodes may not have any traffic at all. Therefore, we reject the idea of defining global vector field to denote network structure. Instead, we present small texture patches representing edges in the network.

3.1. Line Integral Convolution

Our algorithm for visualizing flow is based on Line Integral Convolution (LIC) algorithm [42]. LIC is a well-known texture synthesis technique proposed by Cabral and Leedom. In this algorithm, an input image, which is generally a white noise image, is convolved along streamlines that are retrieved from the underlying vector field. In the following figure, flow of the LIC pipeline is illustrated.

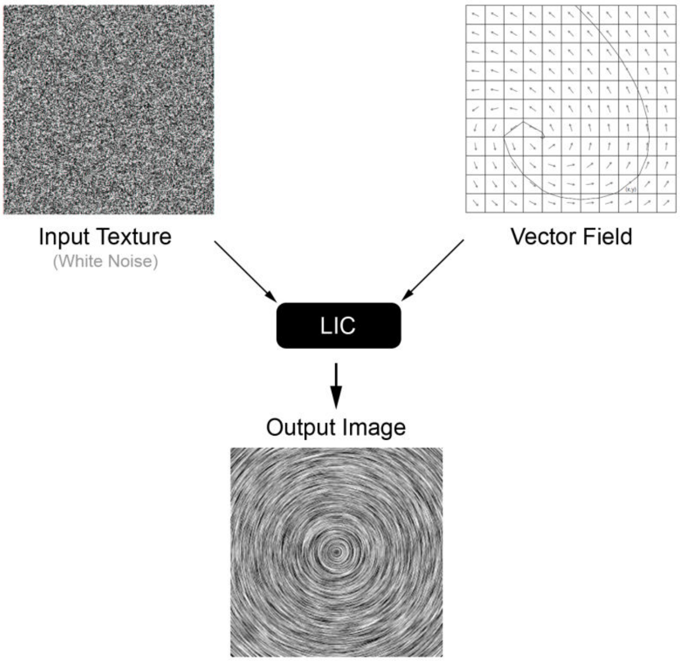


Figure 3.1 LIC Pipeline

LIC algorithm is based on value gathering scheme, where each pixel value of the output image is dependent on the local streamlines that originate from a given pixel. In the process of calculating the output image's pixel values, the contributing pixels are gathered from bi-directional advected streamlines. The length of these local streamlines defines the convolution length. Increasing the convolution length leads to the reduction in contrast; thus, flow lines can hardly be depicted. On the contrary, if the convolution length is too small, flow patterns disappear due to insufficient filtering. Therefore, it should be defined proportionally to the size of the vector field.

Only directional component of the vector field is used in the advection process. Values are gathered from the corresponding pixels and averaging is used to get the convolved value. Figure 3.2 demonstrates the convolution path of a given pixel, when convolution length is 4.

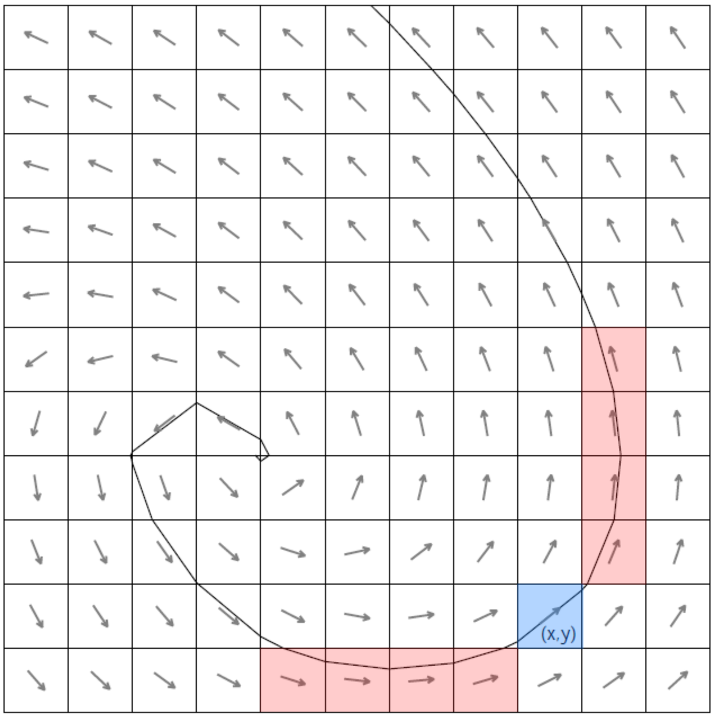


Figure 3.2 Convolution path. Source pixel (blue) and visited pixels (red)

In this figure, selected pixel is colored with blue. Visited pixels are colored with red. Note that there are 4 red pixels in opposite directions.

The LIC output pixel value for a given point *p* is given by the equation:

$$F(p) = \frac{\int_{-L}^{L} F\big(P(p,s)\big)k(s)d_s}{\int_{-L}^{L} k(s)d_s}$$

where *L* is the convolution length, *P(p, s)* is a parametric curve defining the local streamline, *s* is some distance between *(0,L)* in both directions and *k(s)* is the weighting function.

3.2. UFLIC (Unsteady Flow Line Integral Convolution)

There are many extensions to LIC method, including UFLIC [43], FastLIC [44], EnhancedLIC [45], and etc. UFLIC (Unsteady Flow Line Integral Convolution) forms the basis of our flow visualization technique, the main reason being that it has the ability to effectively produce animations with spatio-temporal coherence. Unlike the basic LIC, UFLIC is based on value scattering scheme. Instead of gathering pixel values from opposite directions on the local streamlines, each pixel scatters its value along its path. Paths are defined by the corresponding streamlines. This natural phenomenon of leaving footprints around their flow traces creates the output image by successively scattering and convolving the values.

UFLIC algorithm consists of two steps; time-accurate value scattering process and successive feed forward scheme. Given an initial input texture image, every pixel serves as a seed particle. Every active seed particle advects forward in the vector field, following a pathline that originates from the center of the pixel. The pathline is computed with the equation:

$$p(t+\Delta t) = p(t) + \int_{t}^{t+\Delta t} v(p(t),t)dt$$

where $p(t)$ is the position of the particle at time *t*, $p(t+\Delta t)$ is the new position after time Δt, and $v(p(t),t)$ is the velocity of the particle at $p(t)$at time t.

In the time-accurate value scattering process, each pixel deposits its value and the current integration step along the pathline it travels. Due to the depositing seed particles, every pixel keeps a buffer structure called C-Buffer (Convolution Buffer) for the purpose of convolving scattered values. Each C-Buffer holds several buckets which correspond to different integration steps. The number of buckets that a C-Buffer holds is dependent on the number of integration steps. Each bucket in a C-Buffer has a field of intensity value and a field of the accumulated weight. According to the computation time, corresponding bucket is selected and the computed intensity value is written to the output image. Accumulated intensity and weight values in a C-Buffer are computed in a cumulative fashion as follows:

$$I_{accum} = I_{accum} + I_{current}$$

$$W_{accum} = W_{accum} + W_{current}$$

Each seed particle has a global life span, which in fact defines the life time of a pixel in terms of integration steps. This parameter, *Life Span*, also defines the number of buffers in a C-Buffer. When computing the convolution, the necessary buffer index is obtained from the equation.

Buffer Index = Computational Time % Life Span

Since each pixel advects its value along their path during their life span, there exist some buckets with future timestamps. These buckets will be used when their time comes. Animation is carried by successively increasing buffer indexes, which satisfies the iterative convolution of values from the C-Buffers.

Successive feed forward scheme defines the work flow of the algorithm. Initially, the input image is given as input to value scattering process. After value scattering process advects and convolves the input image, it generates an output image for the current time step. For the further steps, instead of using the initial white noise texture, the output image that is computed from the previous time step is fed forward as an input image to the current value scattering process.

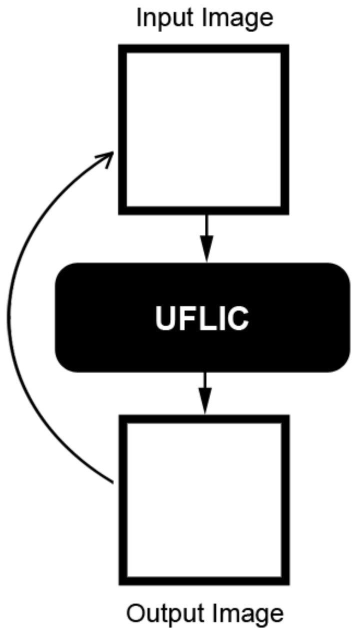


Figure 3.3 Successive Feed Forward Scheme

This successive feed forward scheme is a kind of low-pass filtering process. Therefore, after some time, since we use low-pass filtered output images as inputs, the resulting images get blurred. As a result, the contrast among the flow lines will be lost. One solution for this problem is to apply high pass filter to input image before scattering process.

3.3. Path Generation

Being a sub-field of scientific visualization, conventional flow visualization techniques are built on predefined vector fields. In our flow visualization technique, only regions that have flow properties are filled with vectors. This is supported by using an alpha mask. If a pixel on the input texture is not an active seed, its alpha value is set to 0. In path generation process, visited pixels are defined as active seeds. Using this alpha mask also benefits the system by increasing performance in the convolution process. Consequently, only active seeds are visible and they are included on the convolution phase.

A single record in a geographic network dataset is specified by two entities, source and destination nodes. Since flow textures are mapped onto 2D plane, the position of source and destination nodes are also transformed into 2D coordinates and they are mapped to the corresponding texture coordinates. We define paths over curves in order to give 3D feeling. We used Quadratic Bezier curves for this task. A parametric form of a quadratic Bezier curve is defined as:

$$P(t) = (1-t)^2 P_0 + 2(1-t)tP_1 + t^2 P_2, t \in [0,1]$$

where P_0 and P_2 define the end points and P_1 represents the control point [46]. Parameterization is controlled by the value *t*. The curve passes through P_0 at $t = 0$ and P_2 at $t = 1$. Points between P_0 and P_2 are calculated with the equation above. In our case, the end points are represented by source and destination nodes. Calculation of the control points is illustrated in Figure 3.4. In this figure, source and destination nodes are denoted with red circles. The orange line is the line connecting the source and destination nodes. Orange circle is the point which stands at equal distances from the red circles, computed by taking the mean of the end points. In order to give the feeling of 3D, instead of using orange circle as the control point, the median point is computed in terms of geospatial coordinates, i.e. latitude and longitude. The blue circle denotes the geospatial median point. The blue curve denotes the curvilinear path of the expected flow.

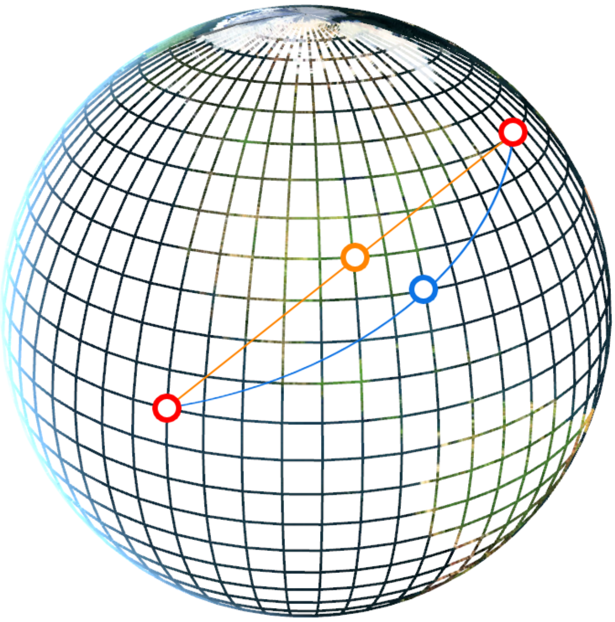


Figure 3.4 Calculation of control points

Similar to source and destination points, control point is also mapped to 2D coordinates. After projecting the control point to 2D coordinate system, a number of curves are defined to generate the path of the flow. The number of curves generated is dependent on the flow width. For instance, if the width of the flow is 10, 10 different curves are generated with their respective control points. Source and destination points remain the same; however, control points are altered in opposite directions. New control points stand on the line that is orthogonal to the line connecting the end points. The illustration of new control point generation is found in Figure 3.5.

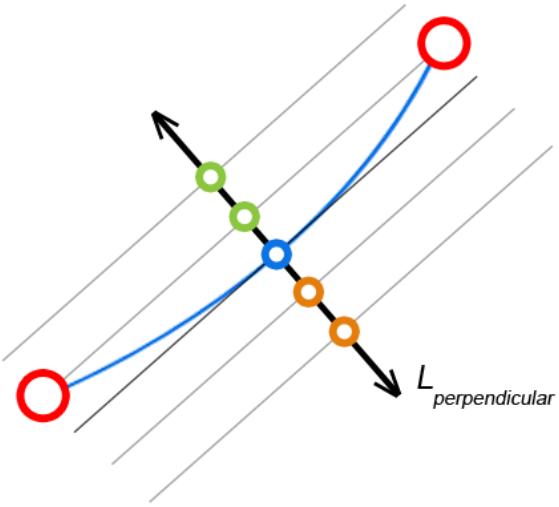


Figure 3.5 Calculation of new control points depending on the width of the flow

In this figure, the black thick line ($L_{perpendicular}$) denotes the line that is orthogonal to the line connecting the end points. Depending on the width of the flow, each new control point is placed in opposite directions with incremental steps along $L_{perpendicular}$. Green and orange circles represent the new control points generated in opposite directions. When the end points and control points are defined, quadratic Bezier curves are formed. Specific positions in a path are calculated as follows. In the parametric form of Bezier curves, t varies between 0 and 1. Starting from 0, we increment t value by $1/L$ in each step, where L denotes the path length. Finally, we obtain an array of positions, called PathLine, in which the size of the array is equal to path length. In the following pseudo-code, path index is denoted by i. i varies between 0 and L. Paths are defined in a recursive fashion such that pixels jump to first position after they reach the end. This precaution is taken for the continuous animation of flows.

```
FUNCTION GeneratePath
step = 1/Length
t = 0
i = 0
while (t <= 1)
  CurrentPos = QuadBezier(Src,CtrlPt,Dest,t)
  PathLine[i] = CurrentPos
  i = i + 1
  t = t + step
end while
```

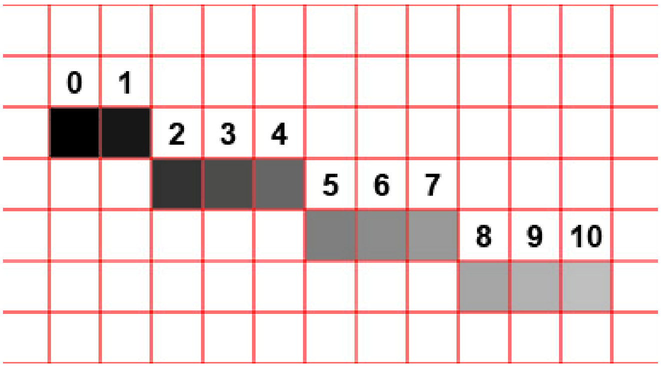


Figure 3.6 Example path in terms of pixel positions. Numbers above pixels denote the path indexes.

In the path generation process, the length of the path has an important role. Path length is a predefined value which is defined with respect to the resolution of the texture. In our application, the resolution of the 2D plane is 600 x 600. There are two parameters that affect the path length, which are life span and convolution length.

$$\textit{Path Length} = \textit{Life Span x Convolution Length}$$

By experimental results, Life Span is set to 30 and Convolution Length to 10, making path length equal to 300. Further details on the meaning of these parameters are discussed in the next chapter.

3.4. Value Scattering Process

In this step, every seed pixel scatters its intensity value along its path. At each time step in their life span, they leave their footprints, i.e. deposit their values in the visited pixel's buckets. For each seed pixel, their alpha value is set to a predefined transparency value. This value is set to 80% transparency but if more opaque flows are desired, higher opacity values can be set. Each seed pixel follows its path in a structure called PathLine. This PathLine structure is simply an array of 2D vectors, representing 2D coordinates in the texture. *Life Span* and *Convolution Length* parameters play an important role in this process. As each pixel moves along its path, it deposits its value via structures called buckets. Depositing time is incremented from 0 to *Life Span*. At each time step *t*, the number of pixels visited is equal to *Convolution Length*. After depositing values with the same time step, time step is incremented by 1. Same procedure is followed for further steps. Illustration can be found in the next figure.

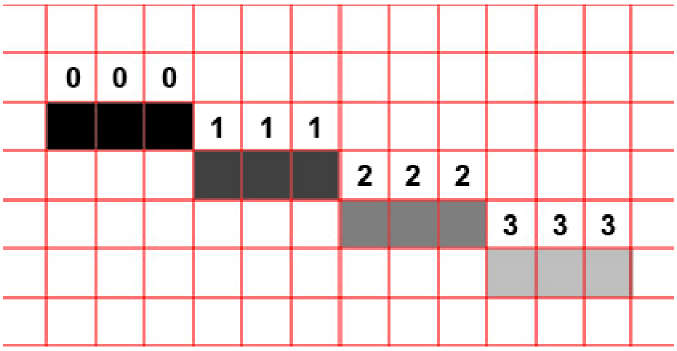


Figure 3.7 Value scattering process. Numbers above pixels denote depositing timestamps.

Each bucket holds accumulated intensity (I_{accum}) and accumulated weight (W_{accum}). When a pixel deposits its value in the visited pixels' bucket, time index is used for reaching the corresponding bucket. In a pixel's C-Buffer, there are n buckets with time indexes i from 0 to *Life Span*. Consequently, the size of each C-Buffer is equal to *Life Span*. The following figure illustrates these structures.

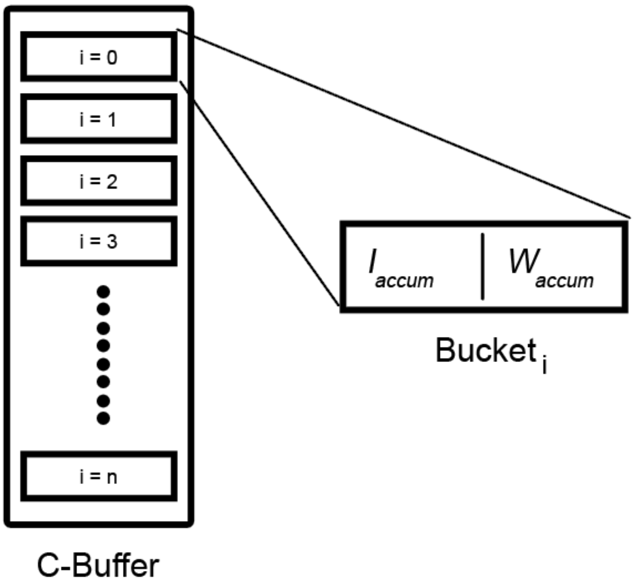


Figure 3.8 C-Buffer and Bucket Structures

C-Buffers are used in the convolution process. After every seed pixel deposits its value, not only for the current step, but for the future steps as well, value scattering process is terminated.

3.5. Convolution Process

Convolution process is triggered after value scattering process terminates. Value scattering process is executed when flow locations are altered. Unlike the value scattering process, convolution process is called during the program. The main reason is that the animation of the flow is carried out in this step. The output of the convolution process is a new 2D texture with the new values acquired from the scattering process. Therefore, this convolution function has to be called with increasing time steps, in order to animate during runtime. The time step parameter of the convolution process is calculated by:

$$Current\_Time\_Step = Buffer\_Index \pmod{Life\ Span}$$

Buffer_Index is increased by 1 after each convolution process. The convolution is initialized by reaching the bucket with timestamp value equal to *Current_Time_Step*. After reaching the corresponding bucket, accumulated intensity and weight values are read. Convolution is performed by dividing the accumulated value by the accumulated weight. The computed value is then written to the output texture.

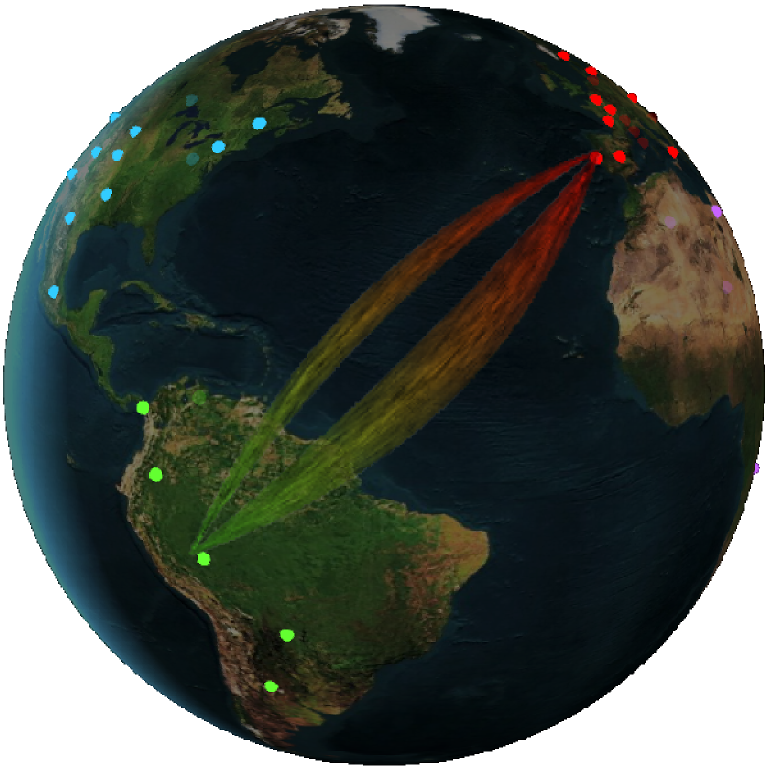


Figure 3.9 A bidirectional flow drawn between South America and Europe

4. LEVEL OF DETAIL FOR GEOGRAPHIC NETWORKS

We performed our level of detail study depending on the camera distance. In high level of detail, flow visualization is disabled. Geographic network traffic is visualized by node-link diagrams in this state. When the camera distance reaches or exceeds the threshold value, flow visualization is enabled revealing high density traffic. In both levels of detail, node-link diagram visualization is enabled. We discuss the significant details of our implementation of node-link diagrams in the next section.

4.1. Node-Link Diagram Visualization

In this visualization technique, each edge in the network is represented as a trail, which is an arc drawn over the surface of the globe. Nodes in the network, which are locations with geographic properties, are represented as spheres. Elements drawn in this visualization are drawn in 3D coordinate space.

Trails are drawn as Quadratic Bezier curves over the surface of the Earth. After depicting the nodes' 3D coordinates, a median point is calculated at the midpoint between these nodes. This is done by following three steps. First, the mean of these nodes' 3D coordinates is calculated. Secondly, this vector is normalized to the unit vector. Finally, it is scaled to fit on the surface of the Earth; hence it is multiplied by the radius of the Earth. The resulting point is referred to the median point. You can see the illustration of this process in the following figure.

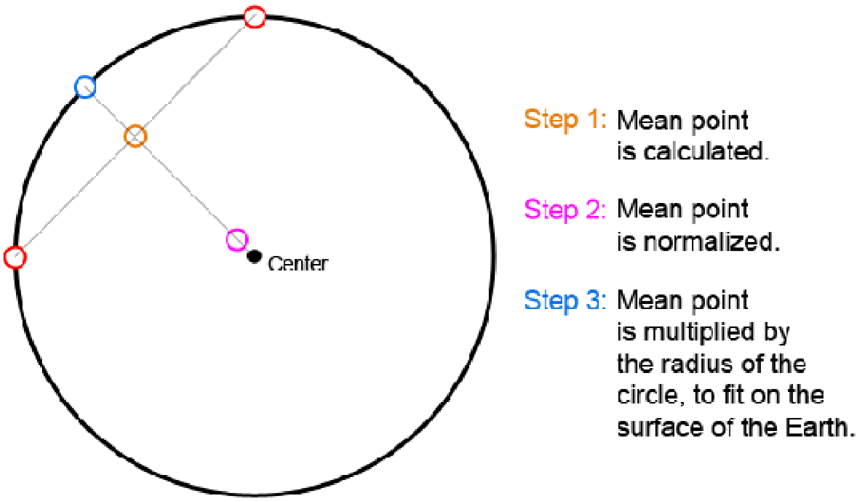

Center
Step 1: Mean point is calculated.
Step 2: Mean point is normalized.
Step 3: Mean point is multiplied by the radius of the circle, to fit on the surface of the Earth.

Figure 4.1 Path Generation of Trails.

After calculating the median point, a quadratic Bezier curve is drawn in the 3D coordinate space. This curve denotes the path of the trail. When there is a bidirectional traffic relation between two nodes, paths are modified to avoid overlapping. In this case, the median points of the two trails are moved in opposite directions along the orthogonal vector defined at the median point. The algorithm used for calculating new control points is the same as in the case of path generation of flows, see Figure 3.5.

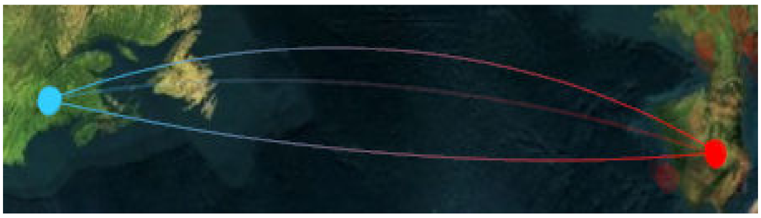


Figure 4.2 Bidirectional trails

In Figure 4.2, the trail at the center is the original path defined between two locations. Since there is bidirectional traffic between them, new paths are defined in opposite directions. Bidirectional paths are drawn with higher opacity.

The length of the trail defines the number of pieces (line segments) that are needed to draw this curve. In other words, a curve is consisted of an arbitrary number of line segments. For animation purposes, these line segments can be encoded with several visual parameters like opacity value and color.

Trails are designed to support many useful visual parameters. Visual attributes that are supported in our representation are explained below in subheadings. Properties of the traffic relations are encoded with various visual parameters.

Coloring

Trails are encoded with gradient colors. Nodes in the same region (i.e. continent) are colored with the same colors.

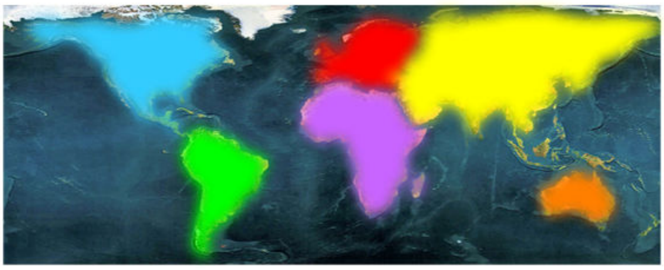


Figure 4.3 Region Coloring Scheme

In Figure 4.3, the region coloring scheme is shown. Locations that belong to the defined regions, i.e. continents, are colored according to this setting. The following figure shows a singular trail that is colored with gradient coloring option. Source location in this traffic relation is in North America; therefore it is colored as blue. Destination location is in Europe; thus making it red. Gradient coloring is done by linearly interpolating colors between source and destination colors.

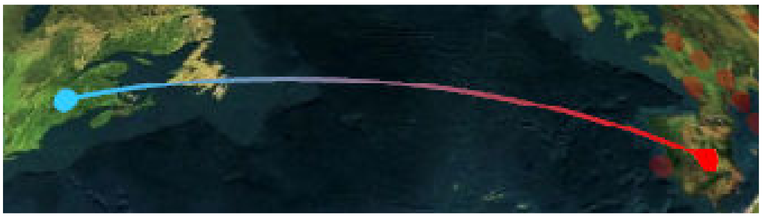


Figure 4.4 Gradient Coloring of Trails

Width

Width is one of the visual parameters that can be used to encode traffic density. As traffic density increases, the width of the trail increases. Figure 4.5 shows two trails width different width settings.

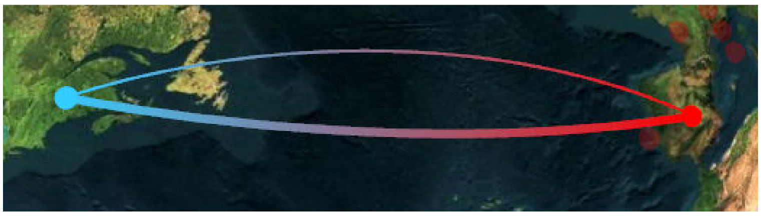


Figure 4.5 Trails with different widths, 1(top) and 5(bottom)

Branching

Branching is the other visual parameter that is used for representing traffic density. Instead of drawing a singular trail, the trail is split into a number of branches. The number of branches, i.e. thin singular trails, is directly proportional to the traffic density.

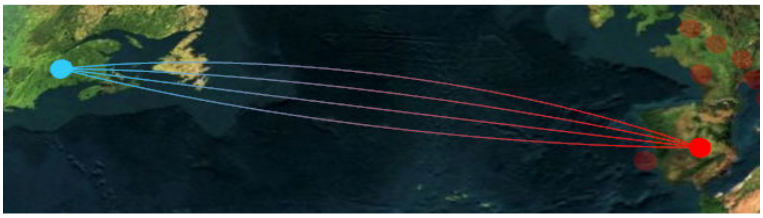


Figure 4.6 A trail split into 4 branches

Animating Bubbles

Animating bubbles are used to stress traffic direction. During the execution of the application, bubbles are animated from source node to destination node. Speed of the animating bubbles is not encoded with a variable in the dataset.

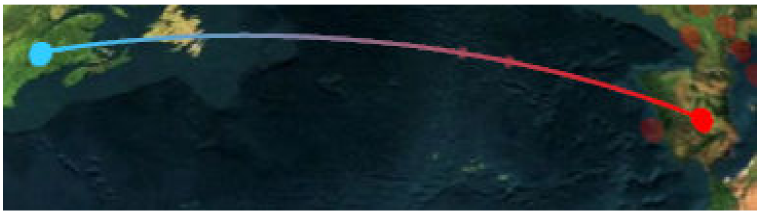


Figure 4.7 A trail with animating bubbles

4.2. Level of Detail Framework

The details of flow visualization technique used in our approach can be found in Section 3. Similarly, node-link diagram visualization technique is analyzed in Section 4.1. We used a randomly generated flight data in our approach. Entries in the data set indicate number of flights occurred between locations. Each record in the traffic data set is consisted of source and destination indexes and the density of the traffic between them. Scaling of density parameter is a crucial step in defining trail properties. When traffic density is below 10, single trail is drawn. If traffic density is in between 10-100, trails are encoded with width or branching factor, depending on the density mapping mode. For example, if traffic density is 50, depending on the mapping scheme, trail width is set to 5, so 5 branches are drawn. When camera distance exceeds the threshold distance value, flow visualization is enabled. Trails having density above 50 are clustered into flow locations, if they satisfy the clustering property. In order to form a cluster, locations that are close to each other have to exceed the cumulative density threshold value, which is 150. After defining the flow locations, flows are drawn while cluster forming trails are hidden. In other words, trails are replaced with flows to emphasize high density traffic.

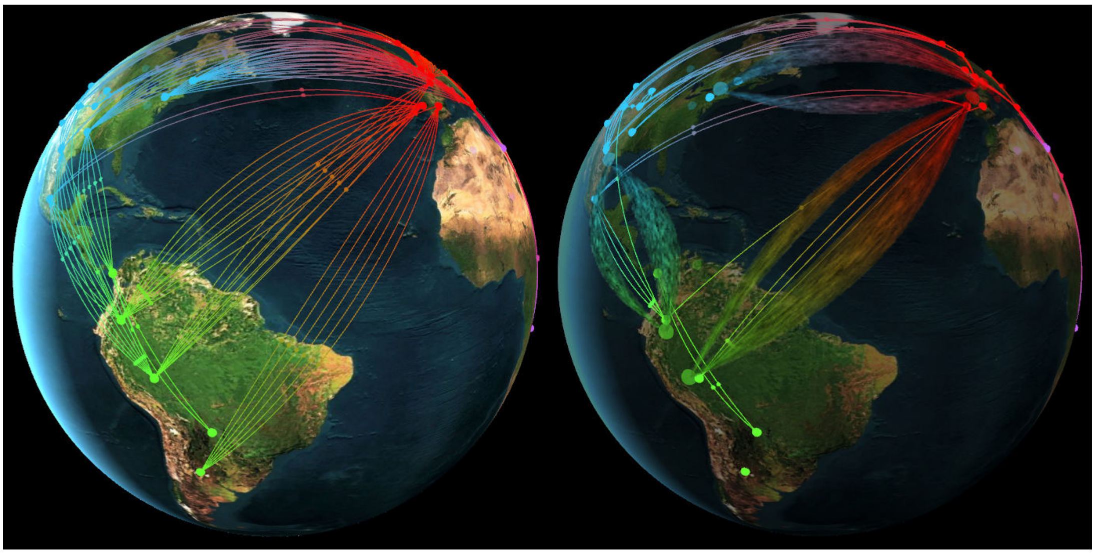


Figure 4.8 Comparison of two visualizations in low level of detail. Node-link diagram visualization with branching enabled (left) and flow visualization (right)

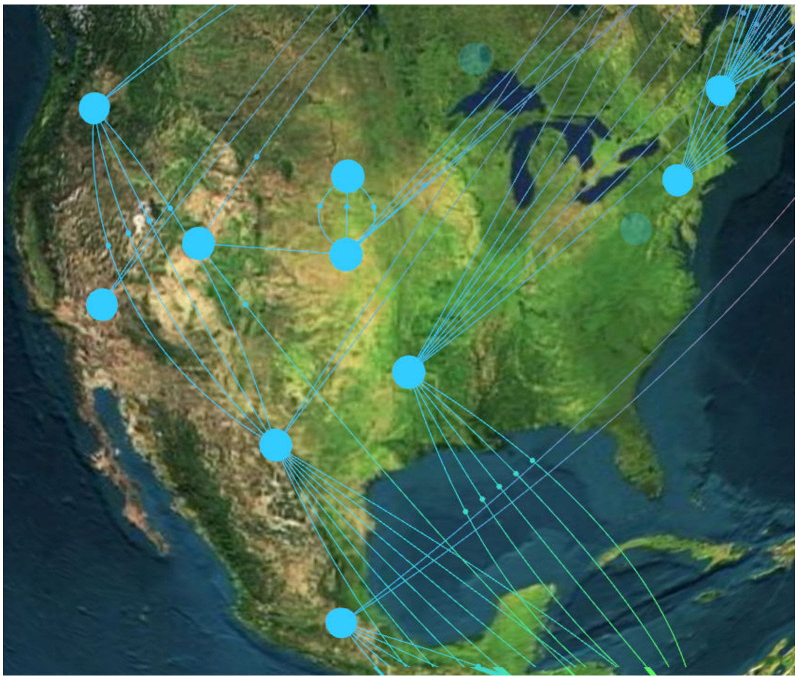


Figure 4.9 Traffic in North America in high level of detail. Node-link visualization (branching is enabled)

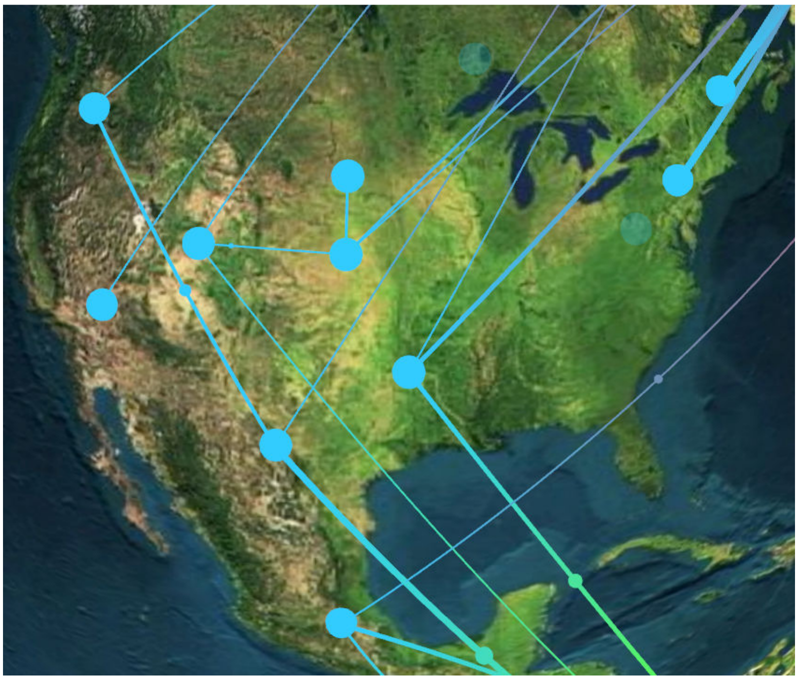


Figure 4.10 Traffic in North America in high level of detail. Node-link visualization (width coding is enabled)

5. EXPERIMENTS AND USABILITY TESTING

5.1. Introduction to Usability Testing

What makes something usable is the absence of frustration, as Rubin et al. clearly states [47]. A product or service can be defined as usable, if it satisfies such concerns like usefulness, efficiency, effectiveness and etc. Clarifying some of these concerns is essential before focusing on usability testing. Usefulness concerns whether the product enables users to achieve their goals and examines their willingness whether they want to use the product or not. If a product is easy to use and easy to learn, but hinders and prevents users to achieve their goals, it will not be used in any circumstances. Therefore, usefulness is the most crucial term in evaluating usability. Efficiency measures the quickness of the user while reaching his or her goal. Unlike efficiency, effectiveness concerns the correctness of the results achieved. It also refers to the error rates acquired from the tests. In an efficiency test, results obtained are in this form: “95% of the users achieved the goal in less than 10 minutes”. However, effectiveness deals with the correctness in a sense that “95% of the users achieved the goal correctly.” [47].

It is clear that usability testing directly focuses on the users. Users are generally busy people; therefore, products should be usable in an efficient way. Finally, users should not face difficulties while using the products. Easy to use applications are always preferable to the difficult ones [48].

Usability evaluation refers to the analysis or empirical study of the usability of a product or system. The goal behind usability evaluation is to provide feedback in software development or other industries. Designers of these systems benefit from usability evaluations in terms of problem detection. They can easily detect underlying causes of the problems in the system and correct the problems accordingly [49].

5.2. Usability Tests

The usability of the designed system is tested with different aspects. In the initial setup of the system, randomly generated data are loaded. Although the data that are fed to the system are the same, visualization techniques and visual representations vary in different cases.

Main goals of the usability tests are the readability of the flow visualization technique, identification of locations that are involved in high density traffic, perception of global trends and subjective comments about the visualization techniques. 20 users are tested for each of these goals.

5.2.1. Readability of Flow Visualization Technique

In the first case study, effectiveness of the flow visualization technique is discussed. Trail visualization is not taken into consideration. The main focus of this case is the readability of the flows regarding the distances between source and destination nodes.

In the visualization system, visibility of a flow is decided by its length with respect to the threshold value, in terms of screen coordinates. If the length of the flow is smaller than the threshold length, it will be hidden. In this study, users are expected to rate the understandability of the flow direction with respect to the different threshold values like 25, 100 and 200 pixels. Rating of the understandability is on a scale from 1 to 5, 1 being the most difficult and 5 being the easiest. In the first step, they are presented with a flow having length equal to 25 pixels. Afterwards, same procedures are followed with length values 100 and 200 pixels.

5.2.2. Identification of Locations with High Traffic Density

In the second section, node-link visualization is analyzed in terms of density encoding. Since the scope of this study is node-link visualization, flow visualization is disabled at all zoom levels. In node-link diagrams, edges may cross over nodes or other edges. This situation may lead to visual clutter that would in fact reduce the readability of relationships. In this study, users are expected to count the number of locations with high traffic density in Europe.

Density encoding in our node-link representation is managed by two different methods; branching and width coding. These methods yield to different views in terms of different levels of detail. Test is performed in four steps. In the first step, width coding is enabled and camera stays on the maximum distance. After the user counts the number of locations with high density traffic, camera is zoomed into nearest perspective, which forms the second step. These two steps are shown in Figure 5.1.

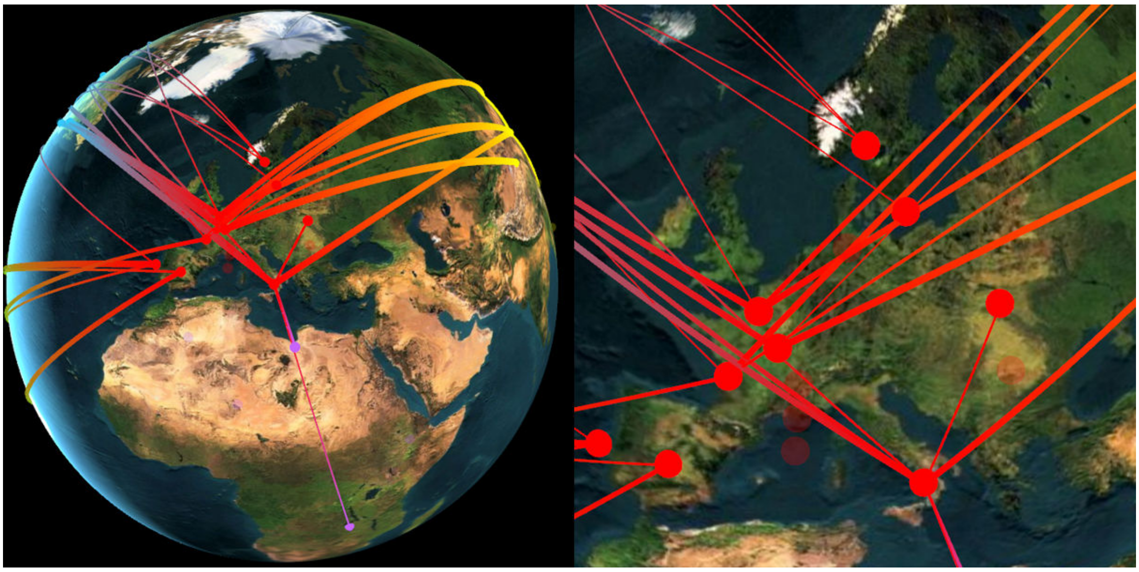


Figure 5.1 Width coding is enabled. Camera distance is set to maximum distance (left), camera is zoomed into nearest perspective (right)

Following these two steps, remaining steps are carried out when branching method is enabled. Same zoom levels are used in this state as well.

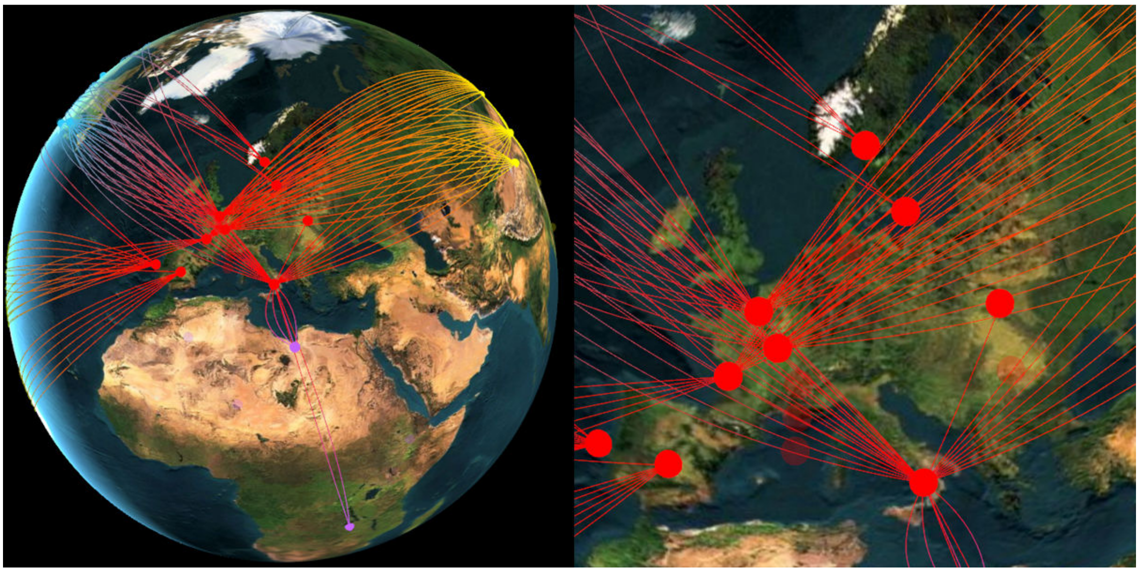


Figure 5.2 Branching is enabled. Camera distance is set to maximum distance (left), camera is zoomed into nearest perspective (right)

It is important to note that, users' personal judgments are regarded in these studies. Discussion of this study clarifies this concept.

5.2.3. Recognition of Global Trends

In the third case study, the perception of global trends around the world is evaluated. Users are expected to analyze high density traffics in the global scale. In other words, they are expected to count the number of regions that imply high density traffic such as from Europe to Asia, etc. Initially, all trails are drawn with width coding enabled. After completing this task, same users are expected to repeat same procedure while branching method is enabled.

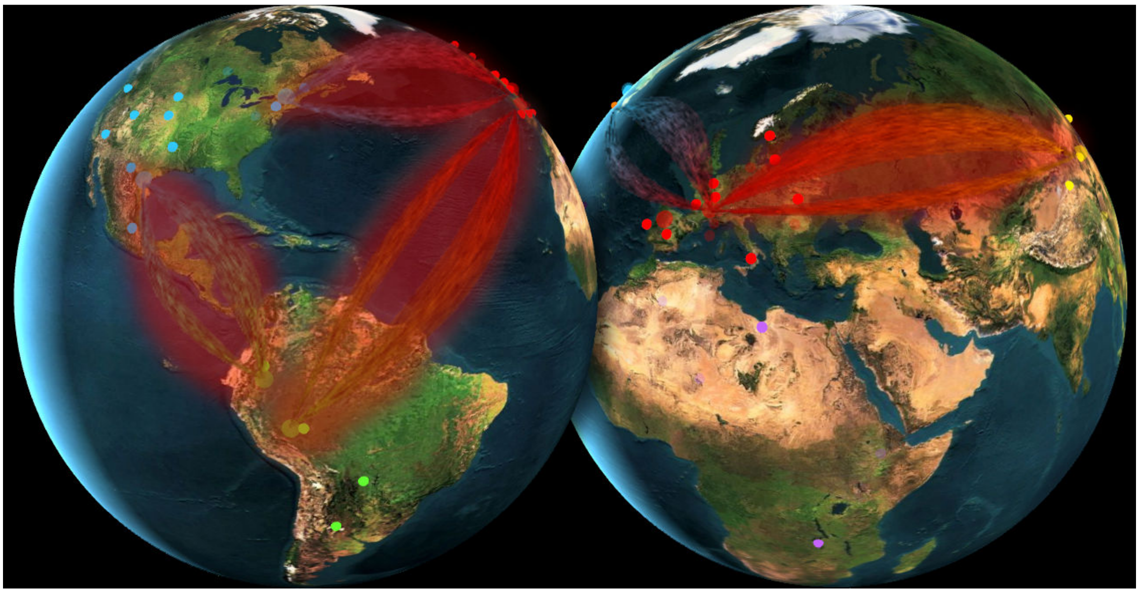


Figure 5.3 Global trends highlighted with red

Following these two tasks, users are encouraged to analyze the global trends when flow visualization is enabled.

5.2.4. Subjective Comments

In the last step of the experiment, users are encouraged to give subjective comments about the visualization techniques that they have experienced in previous steps. Instead of asking their opinion on pre-defined subjects, they are expected to evaluate the success of different visualization techniques based on their experiences. Critical discussion issues from the users' feedbacks can be summarized as:

- Identification of locations in different levels of detail
- Perception of traffic direction and density
- Understanding of global trends
- Readability, visual clarity

The results of the usability tests are discussed in the next paragraph.

We were able to extract significant results from the employed usability studies. Tests were applied to 20 students, of which 75% percent were non-experts.

In the first study, readability of the flow direction is addressed. In all cases, majority of the users were able to detect flow direction without difficulties. Before discussing the results on different flow lengths, it is mandatory to recall that readability is on a scale from 1 to 5, where 1 denoting the most difficult and 5 denoting the easiest. The readability feature is not a quantitative measure; therefore, results may vary deeply.

When flow length is 25 pixels, 9 people out of 20 (45%) voted for 5, 3 people out of 20 (15%) voted for 4, 6 people out of 20 (30%) voted for 3, 1 person out of 20 (5%) voted for 2 and 1 person out of 20 (5%) voted for 1. In the second case, when flow length is set to 100 pixels, results are much clearer. 13 people out of 20 (65%) have chosen 5, where 6 of them (30%) have chosen 4 and 1 person have chosen readability as 3. In the last case, 20 people out of 20 have agreed that the readability of flow direction is 5 (easiest) when flow length is 200 pixels. The following chart summarizes the results. We concluded that the readability of flow direction is proportional to the flow width.

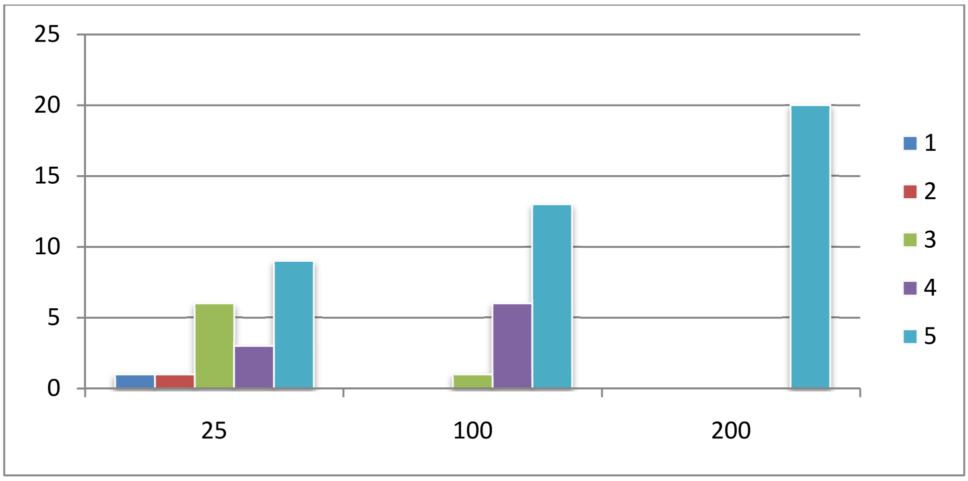


Figure 5.4 Readability of flow direction. The horizontal axis represents the flow length in pixels and vertical axis shows the number of people that agreed on the readability scores.

In the second study, identification of the locations with high traffic density in Europe is analyzed. Different density mapping methods are used in this analysis process. Each method – branching and width coding – is tested with different zoom levels. Figure 5.5 shows the number of locations predicted by the users when branching method is used. Figure 5.6 indicates the same situation when width coding method is used. In both figures, the columns in the front represent the results when zoom level is high. Similarly, the horizontal axis represents the number of locations counted and the vertical axis shows the number of people. Columns in the front represent high zoom level and vice versa.

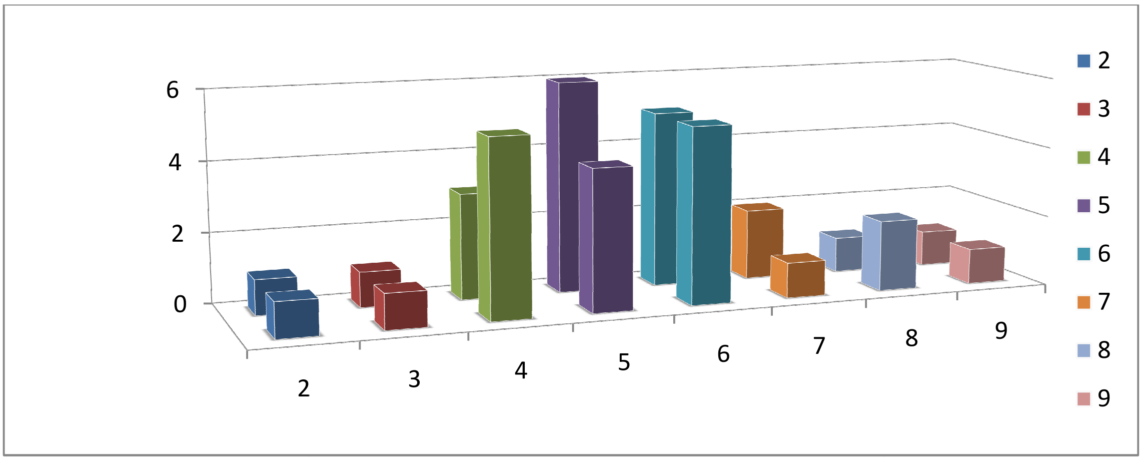


Figure 5.5 Identification of the number of locations with high density traffic in Europe when branching method is enabled.

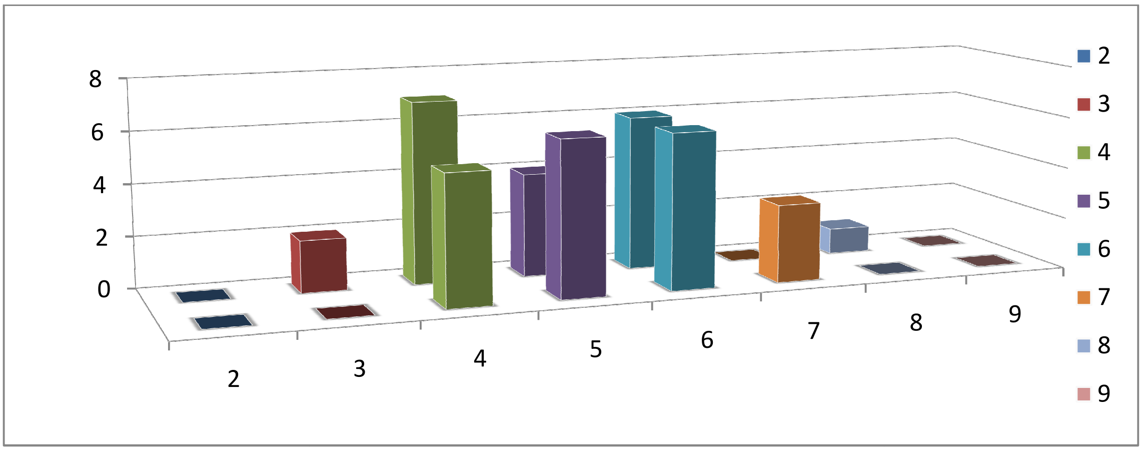


Figure 5.6 Identification of the number of locations with high density traffic in Europe when width coding is enabled.

Although it is impossible to derive specific conclusions from these results, it is evident that in both methods, majority of users have agreed that the number of locations varies between 4 and 6. In both methods, it's improbable to analyze the effect of zoom level. In branching method, some of the users have counted more locations when zoom level is increased. However, the opposite situation, where users counted fewer locations when camera stands on maximum distance, holds as well. When width coding method is enabled, thick lines overlap and the locations below those edges become hidden. The most important result obtained from this study is that, changes in the zoom level cannot be directly related to the identification of locations.

The third study focuses on the global trends in such a way that users are expected to find the number of flow areas with high density traffic. There are 4 flow areas detected by the clustering algorithm used in the application. These flow areas are the traffic from Europe to Asia, from North America to Europe, from North America to South America and from Europe to South America, as also shown in Figure 5.3. It is beneficial to remind that clustered regions form flow locations and flows are drawn between these locations to denote very high density traffic. Multiple trails having traffic density value above 50 are clustered into flow locations. In the application, these regions are calculated automatically and flows are drawn; however, users are expected to find the flow areas with very high density traffic when flows are not visible. In other words, users are encouraged to detect those regions in node-link visualization with branching and width coding methods. Figure 5.7 shows the results. In this figure, the horizontal line represents the number of flow areas counted, the vertical axis show the number of people. Width coding is represented with the columns in the front, branching is represented with the columns in the back.

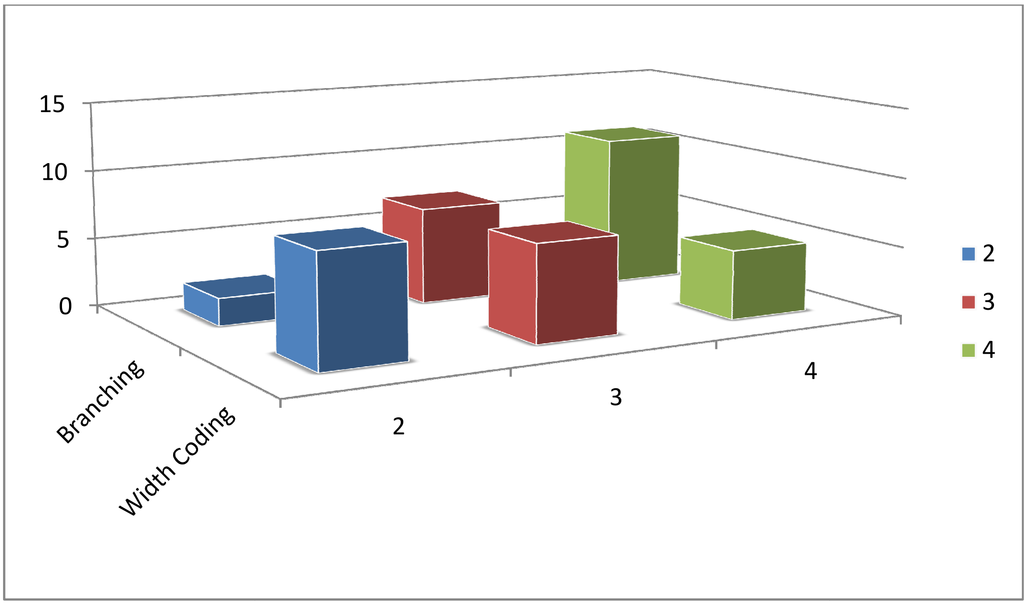


Figure 5.7 Identification of global trends.

It is reasonable to deduce a conclusion that users were more comfortable in detecting the regions with high density traffic when branching method is enabled. 11 out of 20 people (55%) found 4 regions in branching method; whereas in width coding method, only 5 out of 20 people (25%) identified 4 regions.

In the final study, flow visualization is demonstrated and users were encouraged to draw conclusions from the tasks they have accomplished in previous steps of the experiment considering all visualization methods presented. The key subjects that were introduced by users are remarked at section 4.2.4. The results can be summarized as:

- Majority of the users (60%) preferred flow visualization technique for evaluating global trends. They claimed that node-link visualization technique suffer from visual clutter in terms of readability of global structure. Using a completely different visual technique for displaying very high density traffic also attracted users' attention.
- Readability of traffic direction is best perceived by flow visualization technique. The response time needed to analyze traffic direction is too low in flow visualization technique when compared to node-link diagram technique.
- Traffic density is best examined by flow visualization and branching enabled node-link diagram.
- Users experienced difficulties on detecting locations in node-link diagrams when width coding is enabled, due to the fact that some locations were exposed to

overlapping edges. Some users also faced this uncertainty when branching method is used; however, those cases were negligible when compared to the cases when width coding is enabled.

- In high levels of detail, users favored branching method over width coding method. Their main concern was the comparability of width parameter. They argued that making decisions between trails having different numbers of branches is much easier than comparing their widths, especially if their density values are close.

6. CONCLUSION

In this thesis, we employ a visualization tool that incorporates flow visualization techniques in the context of geographic network visualization. An overview of the whole data while encouraging users to discover and analyze patterns in detail is achieved by performing level of detail. Level of detail is managed by altering the layout of the network structure and modifying the visualization techniques. In high level of detail, node-link visualization is enabled to provide detailed information. When level of detail is low, general trends can be easily depicted in flow visualization. We evaluate our study by applying usability studies. The results of the usability studies have shown that some visualization techniques performed better than others on certain aspects. One conclusion we derived is that our flow visualization technique gave promising results in representing network traffic. Users have agreed that our technique was able to convey valuable information about traffic direction and density. Moreover, users have preferred flow visualization technique over conventional node-link diagram visualization in terms of visual clarity. It is also essential to mention that clustering algorithms are not in the scope of this thesis. However, with suitable clustering algorithms, excelling results can be achieved.

While studying level of detail, we conclude that providing unambiguous images is a fatal concern when dealing with global context. Avoiding visual clutter by the removal of trails with high density traffic helped flow visualization technique to outperform other visualization techniques in the identification of global trends. In high level of detail, the visual parameters that are used to encode traffic density in node-link visualization are analyzed. We obtained that users found width coding method unsuccessful in interpretation of traffic density. On the other hand, branching method gave promising results in terms of representing traffic density and comparability in high level of detail.

REFERENCES

[1] D. A. Keim, C. Panse, and M. Sips, “Information Visualization: Scope, Techniques and Opportunities for Geovisualization”, Exploring Geovisualization, J. Dykes, A. M. MacEachren, and M.-J. Kraak, eds., Elsevier, Amsterdam, 23-52, 2005.

[2] J. Dykes, A. M. MacEachren, and M. J. Kraak, “Exploring Geovisualization”, Elsevier, Amsterdam, 2005.

[3] U. M. Fayyad, G. G. Grinstein, and A. Wierse, “Information Visualization in Data Mining and Knowledge Discovery”. Morgan Kaufmann, 2002.

[4] B. Shneiderman, “The eyes have it: A Task by Data Type Taxonomy for Information Visualizations”, Proceedings of 1996 IEEE Visual Languages, 1996.

[5] B. Shneiderman, “Network Visualization by Semantic Substrates”, IEEE Transactions on Visualization and Computer Graphics, 2006.

[6] C. Ware, “Information Visualization: Perception for Design”, Elsevier, 2004.

[7] A. Kershenbaum, K. Murray, “Visualization of Network Structures”, CCSC, 2005.

[8] G. Kumar, M. Garland, “Visual Exploration of Time-Varying Graphs”, IEEE Transactions on Visualization and Computer Graphics, 2006.

[9] N. Henry, J. D. Fekete, and M. J. McGuffin, “NodeTrix: A Hybrid Visualization of Social Networks”, IEEE Transactions on Visualization and Computer Graphics, 2007.

[10] P. Eades, “A heuristic for graph drawing”, *Congresses Numerantium,* 42**,** 149–160, 1984.

[11] T.M.G. Fruchterman, E. Reingold, “Graph Drawing by Force-Directed Placement”, Software-Practice and Experience, 1129-1164, 1991.

[12] T. Kamada, S. Kawai, “An Algorithm for Drawing General Undirected Graphs”, Information Processing Letters 31, 7-15, 1989.

[13] T. Kamps, J. Kleinz and J. Read, “Constraint-Based Spring-Model Algorithm for Graph Layout”, Proceedings of Graph Drawing”, 349-360, 1995.

[14] R.A. Becker, S.G. Eick, and A.R. Wilks, "Visualizing Network Data", IEEE Transactions on Visualization and Computer Graphics, 16-28, 1995.

[15] B. Huffaker, E. Nemeth, and K. Claffy, "Otter: A General Purpose Network Visualization Tool", In Proc. INET, 1999.

[16] B. J. Breitkreutz, C. Stark, and M. Tyers, "Osprey: a network visualization system", Genome Biol. 4(3): R22, 2003.

[17] M. Ghoniem, J. D. Fekete, and P. Castagliola, "A Comparison of the Readability of Graphs using Node-Link and Matrix-Based Representations", Proceedings of IEEE Symposium of Information Visualization, 17-24, 2004.

[18] R. S. Laramee, H. Hauser, H. Doleisch, B. Vrolijk, F. H. Post, and D. Weiskopf. "The State of the Art in Flow Visualization: Dense and Texture-Based Techniques", Computer Graphics Forum Volume 23 Issue 2, 203-221, 2004.

[19] F. H. Post and J. van Wijk. "Visual Representation of Vector Fields: Recent Developments and Research Directions", chapter 23, pages 367–390. Springer, 1994. in: L. Rosenblum et al. (eds.), Scientific Visualization: Advances and Challenges.

[20] M.A.Z. Dippe´ and E.H. Wold, "Antialiasing through Stochastic Sampling," Computer Graphics (SIGGRAPH '85 Proc.), B.A. Barsky, ed., vol. 19, pp. 69-78, July 1985.

[21] D. H. Laidlaw, R. M. Kirby, C. D. Jackson, J. S. Davidson, T. S. Miller, M. Silva, W. H. Warren, and M. J. Tarr, "Comparing 2D Vector Field Visualization Methods: A User Study", IEEE Transactions on Visualization and Computer Graphics, Vol. 11, No.1, 2005.

[22] R.M. Kirby, H. Marmanis, and D.H. Laidlaw, "Visualizing Multivalued Data from 2D Incompressible Flows Using Concepts from Painting", Proceedings in Visualization '99, 1999.

[23] G. Turk, D. Banks." Image-Guided Streamline Placement", Computer Graphics Annual Conference Series (Proceedings of SIGGRAPH'96, August 4-9, 1996), pages 453- 460, ACM Press.

[24] B. Jobard, W. Lefer, “Unsteady Flow Visualization by Animating Evenly-Spaced Streamlines”, Computer Graphics Forum, 2000.

[25] A. L. Fuhrmann and E. Gröller, Real-time techniques for 3D flow visualization. In Proc. Visualization ’98, pages 305–312, 1998.

[26] G. Volpe, “Streamlines and streamribbons in aerodynamics”. In 27th Aerospace Sciences Meeting, Reno, NV, January 1989.

[27] J.P.M. Hultquist. “Constructing Stream Surfaces in Steady 3-D Vector Fields”, In Proceedings Visualization ’92, pages 171–178. IEEE Computer Society Press, Los Alamitos, CA, 1992.

[28] J. van Wijk. “Spot noise-Texture Synthesis for Data Visualization”, In T. W. Sederberg, editor, Computer Graphics (Proceedings of ACM SIGGRAPH 91), volume 25, pages 309–318. ACM, 1991

[29] W. C. de Leeuw and R. van Liere. “Spotting Structure in Complex Time Dependent Flow”, In H.Hagen, G. M. Nielson, and F. H. Post, editors, Scientific Visualization, pages 9–13. IEEE, June 1997.

[30] B. Cabral and L. C. Leedom. ”Imaging Vector Fields Using Line Integral Convolution”, Proceedings of ACM SIGGRAPH 1993, Annual Conference Series, pages 263–272. ACM Press / ACM SIGGRAPH, 1993.

[31] S. Card, J. Mackinlay, and B. Shneiderman, “Readings in Information Visualization”, Morgan Kaufmann, 1999.

[32] D. P. Luebke, “Level of Detail for 3D Graphics”, Morgan Kaufmann, San Francisco, 2003.

[33] B. Engin, S. Balcisoy, B. Bozkaya, “Introducing Level of Detail to 3D Thematic Maps”, Geoviz 2009, Hamburg, 2009.

[34] H. Hoppe, “Smooth View-Dependent Level-of-Detail Control and its Application to Terrain Rendering”, In IEEE Visualization 98 Conference Proceedings, 1998.

[35] A. M. MacEachren, F. P. Boscoe, D. Haug and L. W. Pickle. “Geographic visualization: Designing manipulable maps for exploring temporally varying georeferenced statistics”. In Proceedings, Information Visualization '98, 1998.

[36] ESRI. Arc view. http://www.esri.com/software/arcgis/arcview/index.html.

[37] ADVIZOR. http://www.advizorsolutions.com/products.htm.

[38] T. Munzner, E. Hofffman, K. Claffy, B. Fenner, “Visualization the Global Topology of the Mbone”, Proceedings, IEEE Symposium on Information Visualization ‘96, pp. 85-92, Oct. 28-29 1996, San Francisco, CA, 1996.

[39] B. McManus, “Thematic Mapping”, GIS Resources Document 05-70, 2005.

[40] D. H. House, C. J. Kocmoud, “Continuous Cartogram Construction”, Proceedings of the conference on Visualization'98, 1998.

[41] D. Dorling, A. Barford, M. Newman. “Worldmapper: The World as You’ve Never Seen it Before”, IEEE Transactions on Visualization and Computer Graphics, 2006.

[42] B. Cabral and C. Leedom, “Imaging Vector Fields Using Line Integral Convolution”, Proc. SIGGRAPH ’93, pp. 263-270, 1993.

[43] H.W. Shen, D. L. Kao, “A New Line Integral Convolution Algorithm for Visualizing Time-Varying Flow Fields”, IEEE Transactions on Visualization and Computer Graphics, 1998.

[44] D. Stalling and H.-C. Hege. “Fast and Resolution Independent Line Integral Convolution”, ACM SIGGRAPH Computer Graphics Proceedings. Annual Conference Series, 249-256, 1995.

[45] G. Scheuermann, H. Burbach, and H. Hagen. “Visualizing Planar Vector Fields with Normal Component Using Line Integral Convolution”, In Proceedings IEEE Visualization ’99, pages 255–262. IEEE Computer Society, 1999.

[46] F. S. Hill, "Computer Graphics using OpenGL", pp. 292, Prentice Hall, 2001.

[47] J. Rubin, D. Chisnell, “Handbook of usability testing: How to plan, design, and conduct effective tests”, New York, Wiley, 1994.

[48] J. S. Dumas, J. C. Redish, “A Practical Guide to Usability Testing”, England: Intellect, 1999.

[49] M. B. Rosson, J. M. Carroll, “Usability engineering”, Morgan Kaufmann