Technical University of Munich
Faculty of Civil Engineering and Geodesy
Department of Cartography
Prof. Dr.-Ing. Liqiu Meng

Web based visual analysis of 3D lightning data

Mohammad Abusohyon

Master Thesis

Bearbeitung: 15. 04. 2014

Studiengang: Cartography (Master)

Supervisor: **Dipl.-Ing. Stefan Peters**

2014

Declaration of Authorship

Last name: First name:

I declare that the work presented here is, to the best of my knowledge and belief, original and the result of my own investigations, except as acknowledged, and has not been submitted, either in part or whole, for a degree at this or any other University.

Formulations and ideas taken from other sources are cited as such. This work has not been published.

_____ _____

Location, Date Signature

"Anybody who has been seriously engaged in scientific work of any kind realizes that over the entrance to the gates of the temple of science are written the words: 'You must have faith".

Max Planck

Abstract

Addressing natural phenomenon impacts on urban areas and local environments have been the major workload filling the researchers and analysts communities' schedules.

With the advanced technologies available today, for collecting data during long periods of time (as in satellites technologies), the researchers and analysts become forced to deal with large amount of data related to natural phenomena. This new demand for dealing with huge data sources has urged the need for an appropriate visualization and analysis package, combined of innovative methods and powerful tools to ease the researchers and analysts tasks.

Lately many tools have been developed to provide analysts with comprehensive solutions for performing various analysis and visualization tasks, especially the tasks for detecting phenomenon changes in terms of directions, volumes and coverage.

However, few of these current tools have provided their solutions through the web. The main goal of this thesis is to design and develop a web application that enables the users to perform online visualization and analysis processes for addressing a certain weather phenomenon (lightning storms).

The main design of the web application was focused on providing the user with an interactive visual and explorative tool, for analysing multidimensional lightning cells tracks, in 2D and 3D figures .This visualization capability, will help the user in gaining new insights of the phenomenon dynamics and also in understanding the characteristics of its nature. In addition to that, the design will focus on targeting a broad sector of users, by designing the application to be more feasible and usable for users with different interests and backgrounds e.g. weather analysts, decision makers, students...Etc.

Contents

List of Figures

Abbreviations

LINET	**Li**ghtning detection **Net**work
VHF	**V**ery **H**igh **F**requencies
RDBMS	**R**elational **D**ata **B**ase **M**anagement **S**ystem
OGC	**O**pen **G**eospatial **C**onsortium
GML	**G**eography **M**arkup **L**anguage
GIS	**G**eographic **I**nformation **S**ystem
ArcIMS	**A**rc **I**nternet **M**ap **S**erver
LAN	**L**ocal **A**rea **N**etwork
MCR	**M**ATALB **C**ompiler **R**untime
JVM	**J**ava **V**irtual **M**achine
UI	**U**ser **I**nterface
GUI	**G**raphical **U**ser **I**nterface
HTTP	**H**yper **T**ext **T**ransfer **P**rotocol
XML	**E**xtensible **M**arkup **L**anguage
JS	**J**ava**S**cript
AJAX	**A**synchronous **J**avaScript and **X**ML
JSP	**J**ava **S**erver **P**ages
SQL	**S**tructured **Q**uery **L**anguage
JDBC	**J**ava **D**ata **B**ase **C**onnectivity
APIs	**A**pplication **P**rogramming **I**nterfaces

To my Parents:

Without your love and care, I never succeeded

Chapter 1

Introduction

With the modern technical achievements in producing and manufacturing advance sensing unites for measuring natural phenomena, scientist and research can obtain limitless amount of data representing any phenomenon of interest. For addressing a certain phenomenon, the scientist and research usually examine the data representing this phenomenon under different analysis and evaluation processes. Lately visualization methods and visual analytics techniques become an essential element for examining and evaluating any data set.[1]

The usage of visualization methods and visual analytic techniques has combined two important elements. The first element is to exploit the advances in computer graphics for visualizing patterns or distinguished geometries in the data sets. The second element is to exploit human ability of constructing mental images from visualized patterns and distinguished geometries.

However, even with these advantages of visualization and visual analytics in examining and analyzing data sets, the user still has to deal with many restrictions before begging his process of analysis. One of the main restrictions for the users to deal with is the suitability of the visualization methods for analyzing their data sets. Even with the various available visualizing tools, provide by many platforms, the user still cannot exploit these tools beneficially in the process of visualizing and analyzing his data sets.

In addition to the restriction in beneficially exploiting available visualizing tools, many of these tools are provided in commercial forms and will not be available for usage without prelicensing and authorization from the original developing firm.

In conclusion, for addressing a specific phenomenon represented by large amount of data, the visualization tool should be more centered in addressing this specific phenomenon. The tool should also be provided in a platform independent from any ownership restrictions.

The proper approach for addressing a specific phenomenon is to build and develop an application that's oriented from the begging to address and analyze this specific phenomenon. The application design and implementation tool should serve the visualization and analysis of the phenomenon in the most optimum way. The application should cover with its functionalities every possible aspect for visualizing and analyzing the phenomenon. In result the

user will be provided with a tool for performing special information extraction and knowledge revealing for the phenomenon hidden characteristics, which leads to better realization of the phenomenon potential impacts and helps in improving decision making process.

To insure the user accessibility to the application functionalities, the application design also should focus on providing the application free from any software or hardware restriction. This requirement can be achieved by finding an independent medium for disseminating data and communicating with users. The best medium for providing limitless access to the application functionalities is the web.

Later on, in the following chapters, a detail explanation will be provided to discuss the issues of designing and implementing visual analysis applications on the web.

1.1 Problem Definition

The lightning storms phenomenon can be considered as one of the frequent weather conditions affecting the upper region of Bavaria (Germany).

Detecting this dynamic weather phenomenon can be done using various methods and measurements techniques such as weather satellites, special radars or ground detecting networks.

The European lightning detection network (LINET), is considered to be one of the permanent data sources for detecting and positioning lightning storms around Europe. Capable of providing 3D coordinates measurements for lightning strikes points with acceptable accuracies.

With the importance of addressing the lightning storms phenomenon as a dynamic weather condition affecting the upper region of Bavaria and with the available and continuous positioning data for lightning strikes points provided by LINET, it becomes a necessity to design and implement an interactive tool that can foster the process of visualizing, analyzing and understanding this weather phenomenon.

A first try for designing and implementing an interactive visual analysis tool was done by Stefan Betters and Liqiu Meng[1]. The interactive visual analysis tool was build using MATLAB software platform and provided multiple interactive representations of the lightning phe-

[1] Technical University Munich, Department of Cartography, Munich, Germany

nomenon. The interactive tool showed high performance for loading and visualizing lightning data according to different user demands.

However, this interactive tool can only be run and executed on a MATALB software platform, which added a level of restriction on the uses of the interactive visual analysis tool. MATALB is not just a developing software platform; MATALB is a commercial software supplier that applies many ownership and authorization policies on the usage of MATALB software products. These MATALB policies prevent any use or access to any of the MATLAB products without a previous MATALB authorization.

This means that for any user interested in implementing the interactive tool, MATALB software installation and authorization should be first done so later the user can run and execute the interactive visual analysis tool.

In conclusion, for implementing an efficient visual analysis tool, for the purpose of addressing lightning storms data, new efforts should be provided to design and develop a visual analysis tool that is free from any restriction. For filling this task, a scientific research topic was presented for addressing the possibility of extending the existing visual analysis tool to the web.

The research should cover both the theoretical and practical parts for addressing and implementing the suitable solution.

1.2 Objectives

As motioned in the previous section, the lightning storms phenomenon has a major effect on the weather condition in the upper region of Bavaria. An advance visualization and analysis tool was implemented on the available lightning data with a level of restriction on freedom of usage. To overcome this restriction, a suggestion was made to address the extension of this visual analysis tool to the web.

In this context, the objectives of the thesis research are as follows:

- Design and develop a new interactive visual analysis tool. The design should implement a tool free form any restrictions, especially the type of restrictions presented by the developing software platform.
- Provide unlimited access to the new interactive visual analysis tool functionalities by using web capabilities with no interference from any third party.
- Benefiting from the existing interactive visual analysis tool design and techniques.

- Providing multiple representations of the lightning storms data in the extended web visual analysis tool, to reveal the hidden characteristic and analysis the lightning storms data from every possible aspect.
- Embedding a fusible interface in the new web visual analysis tool design, to ease the interaction between the user and the visual and analysis functionalities.

1.3 Research solution

To overcome the previous mentioned restriction in the problem definition section, when using the visual analysis tool and also to achieve the objectives mentioned in the previous sections, the appropriate solution is to redesign and develop a new visual analysis tool that promotes the free use of all existing visual analysis functionalities on the web.

To achieve this task, a research should be done to find the best software platform for the web application development. The software platform should provide an independent developing environment from any ownership restriction. The best solution for this requirement is to find an open source for providing developing software platforms.

The fourth chapter of this thesis will provide a full explanation of the research process in finding the suitable web developing software platform. As an introduction for the upcoming solution, the research focused in using Java technologies as an open source for building and developing the web applications.

1.4 Thesis Structure

The thesis general structure revolves around presenting the idea of extending visual analysis techniques to the web. The preliminary chapters will present the current challenges with the general proposal for providing the proper solution, later a research of the related scientific matters and disciplines that could be beneficial for the implementation process will be provided. The fallowing chapters will give an overview of the data used in the web application. Later a methodology for building the web application will be presented. The implementation process will be presented in details in a separate chapter and the last chapters will presents the conclusions and the outlooks.

In the following, a detailed explanation for the content of each chapter:

Chapter 1: provides an introduction to the importance of extending visual analysis techniques to the web. The chapter discussion will define the case of interest under process and summarize the objectives of addressing this case.

Chapter 2: presents a scientific review of the related disciplines fostering the design and implementation process. The chapter discussion will focus on defining some basic terms used in addressing the lightning storms case. Details concerning related scientific fields will be introduced with an overview of some similar approaches that already implemented visual analysis solutions on the web.

Chapter 3: contains detailed description of the data that will be visualized and analyzed under the web application.

Chapter 4: presents the proposed methodology for building and developing the web application with details concerning the used technologies and techniques.

Chapter 5: presents detailed explanation on how the methodology techniques were implemented.

Chapter 6: presents the work discussion and conclusion.

Chapter 7: presents the recommendations and outlooks for future updates and enhancement of the implemented web application.

Chapter 2

Scientific review

Before discussing the techniques for extending visualization and analysis methods to the web, a review for some of the fundamental concepts related to dynamic phenomena analysis, should be done. Knowing how to define a dynamic phenomenon and understanding what is it refereeing to, eases the process of designing and implementing a visual analysis tool. The solid understanding of the phenomenon nature helps the developer in designing a tool that suits the phenomenon examination and analysis process. In other words, understanding the geographical phenomenon nature is an essential element for designing an efficient visualization and analysis tool.

The fallowing section will provide a comprehensive definition of the dynamic phenomenon, which is taken from various disciplines of science. Also the fallowing section will present the geographical methods for modeling a dynamic phenomenon, in seek of better understanding and realization of the dynamic phenomenon.

2.1 Definitions

2.1.1 Dynamic phenomenon

What is a dynamic phenomenon? What does it refer to of events occurring in the nature? And how can we describe the dynamic phenomenon?

A dynamic phenomenon description varies in different ways, according to which discipline of science is addressing the dynamic phenomenon. If the dynamic phenomenon has been addressed under the field of Geophysical Fluid Dynamics[2], the definition will be comprised of equations and mathematical formulas explaining the dynamic processes and potential patterns of the phenomenon's movements [2]. If the concepts of the Geophysical Fluid Dynamics discipline are used to address the lightning storms phenomenon, the developer will be using the atmospherical model to define and model the lightning storms phenomenon.

[2] The main subject addressed by the discipline Geophysical Fluid Dynamics, is the natural phenomenon motion occurring in earth oceans and atmosphere over a wide range of time and space. the discipline uses nonlinear dynamics, mathematical analysis ,computational modeling, theoretical approach lab simulations and filed measurements to address and explain the nature of dynamic phenomenon.

To preserve the context of this thesis, the dynamic phenomenon definition will be explained by a cognitive approach. The cognitive definition can be derived from the cognitive linguistic disciplines.[3]

The cognitive linguistic disciplines don't refer explicitly to the natural dynamic phenomena. However they still provide an obvious cognitive archetype for defining and modeling any dynamic phenomenon.

The proposed cognitive archetypes, suggests that any dynamic situation is originally composed of certain basic elements, these elements are: the *state*, *event* and the *process*.

The cognitive archetype definition depends on cognitive perception of topological relations between the objects in reality and depends also on cognitive realization of changes in objects characteristics during time.[4]

For defining a dynamic process, the cognitive archetype definition revolves around the idea of considering a dynamic process as a hierarchical structure of *states or events and processes*:

Initial Situation (state/event):	represents an object in the initial phase.
Transition (process):	represents changes and actions on the object that transforms the object status from the Initial phase to the final phase (with a temporal sense).
Final situation (state/event):	defines the final status of the object.

The cognitive archetype definition also provides taxonomy for the transitions (processes).

According to the definition, the transitions can be split into different types; depending on the different status the object can take during a transition. Going more in to details, if a transition between two situations doesn't indicate a main change in the object properties, the transition will be referred to as static situation transition. [5] On the other hand, if the transition between two situations phases indicates a main change, the taxonomy will turn the focus on the object's spatial movement and the forces steering this movement. If the spatial movement was under constrain, the transition will be mark with a different taxonomy than if the spatial movement wasn't under constrain.

As an example for implementing this cognitive definition, Shipley, Fabrikant and Lautenschütz have explicitly mentioned the conceptual term *event* to refer to the stages of a Dynamic geographic phenomenon [6].

However, even with this cognitive archetype definition, that's facilitating the realization of

a dynamic phenomenon, it's better to rely on a definition that has been extracted from the specialist domains for addressing geographical dynamic phenomena.
Goodchild and Glennon [7], have provided a specific definition for the term geographical dynamic phenomenon. In their definition, the terminology "geographical" referrers to "the surface and near-surface of the Earth" and the terminology "dynamic" referrers to "changes through time, and the characterization, understanding, and prediction of such changes".
In other words, *a geographical dynamic phenomenon is any phenomenon occurs at the surface or near the surface of the Earth with a predicted change through time*.

2.1.2 Discrete data and Continuous fields

Extracting spatial information from humans' descriptions is considered to be a poor and inefficient process. When people try to map their surroundings, depending on their own understandings of the geographical elements, they always mix their descriptions with their own definitions of topological and topographical relation. Normally these definitions changes from an individual to another and they never correspond. The resultant of human descriptions for earth surface is unfortunately complex expressions and poor descriptions that cannot depict well the earth surface.
To solve this inconsistency between the humans when they are trying to depict the earth surface and also to solve the non uniformity between humans when they define the topological and topographical relations between the earth surface features, formalized definitions and comprehensive models of the earth surface should be proposed.
In these definitions and models, the earth surface features are defined by unified representations that distinguish each earth surface feature from another. Also the definitions and models will present a unified explanation of the topological and topographical relations between the earth surface features.
Finally these definitions and models will be used by every observer to describe the earth surface features and shear this description with the others .[8]

One of the most used definitions for modeling the earth surface features is the ***Discrete data*** definition.

The ***Discrete data***, referrers to the entities and objects separated in space of interest with known geographical location, with also determined boundaries and distinguished attributes. The discrete data is usually used in representing man mad features (houses, blocks, Streets...Etc) or separated natural features (forests, rivers, lakes...Etc.)

Another widely used definition for modeling the earth surface features is the ***Continuous fields.***

The ***Continuous fields***: represents a smooth and continuous variation of known variable over predetermined space, the variable variation can be represented by a continuous surface or can be defined by a mathematical equation. The continuous fields can be implemented for geographical features as terrains elevations or for field properties such as air temperature, soil minerals concentration...Etc.

It's important to mention here, that there are many definitions defining discrete data and continuance fields terms on bases taken from databases management and structuring design. The goal of this particular definition of discrete data and continuance fields' terms is to make the representative elements of earth surface features in the data model more fitting for database operations (data querying, data exploring and data retrieving).[9]

2.1.3 Spatially extended objects

After discussing the definitions of a dynamic phenomenon and also after discussing the definitions modeling the natural geographical features, for the goal of providing a unified representation and an efficient sharing process, the next subject that falls in the context of defining important concepts related to dynamic phenomenon analysis, is the definition of spatially extended objects.

The definition of spatially extended objects focus on defining and distinguishing the boundaries and borders of spatially extended object. These boundaries and borders can be distinguished ether physically by the physical appearance of object border (e.g. a lake has a defined physical border) or arbitrary by human demarcations of the object border (e.g. a province has human demarcation border).[10]

Smith and Varzi have discussed the potential hypothesis for physically distinguishing the object borders, according to the object inner elements uniformity. If the object inner components are composite uniformly with distinguishable inner boundaries, the object can be divided according to these boundaries into several components with well defined perimeters.

Smith and Varzi have also discussed the hypothesis of distinguishing the object borders from an arbitrary perspective, where the object components are not the basic concern for the division process, but rather more the cognitive human vision for dividing this object. By this hypothesis, Smith and Varzi were pointing toward the human nature and capabilities of proposing

non realistic dividing borders in the object of interest, to simplify the realization of this object (e.g. dividing the earth in to two hemispheres).

For lightning storms case, the main concern is to distinguish the exterior borders of the lightning storms especially when their volumes (or extensions) are changing during time.

Depending on Brentano's definition [11] for boundaries of features varying in their characteristics with respect to space and time, Smith and Varzi have introduced the term: coincidence of fiat boundaries. Fait boundaries are an imaginary separation lines between dynamic object components. Usually these separation lines are determined by cognitive realizations and not by physical characteristics. The fiat boundary resulting from the separation process is considered as a determination boundary that belongs to both adjacent separated parts.

Smith and Varzi have also included Brentano's proposal for the process of determining fait boundaries according to a temporal aspect. Brentano's suggested that fait boundaries can also be defined in a temporary matter, in which the fait boundaries are located in space, at a certain time. Later these fait boundaries will be relocated during change of time.

Taking under consideration the coincidence fiat boundaries for separating a dynamic object body, with the proposal for temporary determination of boundaries, the result will be an applicable technique for dynamically visualizing and molding the lightning storms. Where the lightning storm is divide in to various components (e.g. convex halls in 2D or 3D) with fiat boundaries relocated according to time change and regardless of any physical boundary of the whole lightning storm.

2.2 Related scientific disciplines

After discussing the meaning of the term dynamic phenomenon and also after discussing the suggested representation forms for a dynamic phenomenon, time to discuss the disciplines involved in the process of visualizing and analyzing a dynamic phenomenon. The discussion will also include the disciplines used to extend the dynamic phenomenon visualization and analysis methods into the web.

2.2.1 Visualization

To understand the importance of visualization in the processes of gaining new insights and information, the human visual system should be taken under focus[12]. The human eye contains 125 Million receptors capable of receiving an electromagnetic spectrum with wavelengths between 390 to 700 nanometers, which known as the visible spectrum. During this process of receiving the visible spectrum, approximately third of the human brain is involved in analyzing the received data from the visible spectrum. The brain eventually will build a mental image from the received visual spectrum and proceed in further realization process.[13]

Knowing these facts about the human visual system and the human reaction toward visible spectrum, leads to the importance of embedding the visualization element in any process related to data maiming or data analysis.

Also from the previous explanation of human visual system and human reaction process toward visible spectrum, a formal definition for the visualization process can be derived as the following:

"Visualization is a human cognitive process for forming mental image of a domain in space".[14]

However, for the context of visual analysis, the visualization definition was interpreted by many scientists as the use of computer graphics and representations for handling large amount of data to extract new knowledge and gaining insights.[15]

WILLIAMS, SOCHATS and MORSE have introduced more specifications into the visualization definition, to include computer technologies in the visualization process. WILLIAMS, SOCHATS and MORSE have defined the visualization term as: "*Using graphics, Images and animated sequences to represent complex data structure and complex data dynamic behavior to reveal the events, processes and concepts*". The following two sections, will present more principles and techniques in the same context with WILLIAMS, SOCHATS and MORSE definition, to reveal and extract hidden information from datasets that represents dynamic process.

2.2.2 Visual analysis and temporal analysis

When testing a data set under conventional data analysis processes, the data set values will be processed and manipulated according to a certain hypothesis for revealing the hidden information. The analysis process will also measure the conformity of the data analysis results to the hypotheses assessment for the outcomes. In exploratory data analysis, the approach for processing the data set focuses more in revealing the hidden information with an empirical process. In this process the analyst will explore the data regardless of any hypothesis, searching for unusual trends, patterns, relationships or inconsistencies appearing visually in the mapped data. later the analyst will present his conclusions as an extracted information from the data set visual testing.[16]

During this process of ***visual analysis***, the analyst main strategy will be to monitor and measure the variations in the phenomenon properties in every possible form. The monitoring and measuring process could focus on the phenomenon spatial distribution in space, or on the variations of the phenomenon coverage area and occupied volume in space. Also the monitoring and measuring process could include the focuses on the phenomenon movement's direction and speed during time. Another possible form of observing phenomenon variations is to observe the variation in the phenomenon unity. In other words, is the phenomenon consists from one main object during the phenomenon movement? Or possibly can be split into multiple objects during this movement?

Temporal analysis

As simple as the monitoring and measuring process seems to be, still the analyst cannot perform the monitoring and measuring process immediately, some important data organizational aspects should be determined at the beginning.

To enable the analyst from measuring the variations in a certain phenomenon, the phenomenon should be divided into small portions. Dividing the phenomenon into small portions eases the tasks of observing and analyzing the phenomenon variations more than if the phenomenon been observed and analyzed as a one complete unite. However, even with this advantage driven from portioning the phenomenon, it's important to point out that the portioning process has a significant effect on the observation and analysis process outcomes. According to which portioning technique the phenomenon is divided to, the observation and analysis values will change considerably. This portioning effect has gathered the attention to the importance of defining a basic technique for dividing the phenomenon body in to small portions without af-

fecting the observation and analysis outcomes. In other words, finding a portioning technique for extracting realistic changes in the phenomenon characteristics.

DONNA J. PEUQUET and NIU DUAN [17] have proposed ***time periods***(in their model for spatio-temporal data analysis) as the organizational technique for observing and detecting realistic changes in the phenomenon properties, phenomenon characteristics or phenomenon relationships when using visualizing analysis process.

In this time based model they suggested "time to be the primary organizational basis for recording changes". The sequenced changes in phenomenon components and phenomenon properties will be detected according to sequenced time periods in the timeline of the phenomenon.

The detection will begin from the starting point in the phenomenon timeline and stops at the end of the first period. The changes in the phenomenon components and phenomenon properties during this period (e.g. location in space) will be recorded and measured. The detection process continues for each phase of the timeline and at each phase the changes in the phenomenon components and phenomenon properties will be recorded and measured until this process finished by reaching the last phase in the timeline.

DONNA J. PEUQUET and NIU DUAN have also proposed multiple procedures for portioning a dynamic phenomenon into different time periods. For example, they proposed that the portioning technique could depend on distinguishing major changes occurring during the phenomenon life time (an event) and later use these distinguish changes to portion the phenomenon into time periods. The distinguished changes will be recorded individually in the phenomenon timeline and their time of occurrence will define the time periods quantities and durations.

This technique for recording the distinguished rates of changes inside a phenomenon and later portioning the phenomenon timeline to periods according to the occurrence time of each one of the distinguished changes has been known inside the analysis society as the *event-based process*.

For portioning the lightning storms phenomenon, the desktop application for visualizing and analyzing lightning storms has implemented a time base approach.

The lightning point data were separated into time intervals with duration of 10 minutes. The resultant portions have contained at least 10 lightning strikes points. Later the lightning portions were clustered to bigger cells for better representation and visualization. The clustering was distance based with distance-threshold of 6 km.[18]

2.2.3 Visual Analytics:

The primary idea behind any visual analytic process is to combine humans' cognitive capabilities when forming mental images with computer advance graphics representations of natural phenomena, for providing better realizations and understandings of the natural phenomenon characteristics. In this process the user will implement the available visualization and analysis methods to reveal the hidden information with high quality. The higher quality in revealing hidden information will enable the analyst of reaching the correct conclusions and performing the right decisions.[19]

To transform this theoretical idea to an applicable procedure the user should be provided with an interactive visualization and analysis tool. The visualization and analysis tool will provide the necessary platform for combining user cognitive capabilities with computer graphics capabilities[3].

2.2.4 Methods and tools of visual analytics

A visual analytical tool varies in functionalities and techniques according to the cases binning under analysis. Also the visual analytical tool will vary in functionalities and techniques according to the type of data representing these cases binning under analysis. In the following a list of the Visual analytical tools used in addressing different natural phenomena with different data types:

- **Map animations**: used for presenting the phenomenon movements or the phenomenon changes during the phenomenon occurrence time.
- **Map iterations**: ordering a group of maps representing one particular phenomenon in a special sequence to present the phenomenon changes and variations.
- **Data Querying**: In a querying process, the phenomenon data are usually structured in a relational database management system (RDBMS), to enable users of sending data queries requests and receiving data queries results (movement type, direction, speeds, time ...etc). [20]
- **Focused view**: represents a partial view taken from a bigger scene with higher zooming levels. The focused view presents the data with a significant enlargement (especially in 3D presentation) with also the possibility of panning and rotating the view. Usually this technique is used to visually examine the presented phenomenon with high level of de-

[3] More details for designing and implanting an interactive visualization and analysis tool will be provided in chapter 4, section 3

tails obtained from the enlargement process. However this technique is only displaying a part of the data and a *link* to the whole data display is needed. The linking process is performed by providing both displays of the data next each other, the focused view and the original display and then shading the area enlarged by the focused view inside the original display.[21]

- **Space-time cube**: used for presenting discrete data with continuance changes during time of occurrence. In this technique, the discrete data are aggregated to clusters according to a temporal factor [22]. For positioning the clusters inside the cube, the external sides of the cube are used as scaling axes for positioning the data clusters.
  Usually the edges of the rectangular base inside the cube are used to represents X and Y axes. For representing the altitudes of clusters, the cube height is used as an altitudes scaling axis. [23]

2.2.5 Four dimensions representation:

As mentioned in the previous section, the space time cube technique is used to position clustered data in a 3D display. These clusters are viewed during the whole time of the phenomenon occurrence. However, if these clusters can be visualized or hidden according to changes in time, this will simply transform the space time cube functionality from 3D representation to 4D representation, where the fourth dimension refers to time shifting. For applying this transform of functionality in the space time cube, an additional element should be included to add the fourth dimension representation. The additional element is known as the “interactive display” which is basically a dynamic visualization that displays the visualized objects according to shifts in time; usually the time shifting is implemented by user requests. [24]

For adding the four dimensional representation to the application that visualize the lightning storms data, a time slider (that can be controlled by the user) was proposed. By using this time slider, the user can shift between different time intervals and the application will apply changes (interactively) to the main display according to the shifting process. The user can visually analyze the changes in the phenomenon characteristics according to which part of the timeline the phenomenon has been visualized to.

2.2.6 Web Mapping:

Using the World Wide Web as disseminating tool has become the main theme of our modern digital era. When following the aspects of our daily life and daily activities, as working or studying or even searching for a near grocery shop, we notes that accomplishing these activates always involve the process of gathering information from the internet. The gathered information can be received in different forms as in a visual form or in some services in an acoustical form.
These capabilities of publishing information in different forms and the feasible of using the internet as a tool for publishing data has influenced the scientist and researchers in the cartography discipline to include the internet capabilities in the processes of designing and producing modern maps.
The important use of maps as symbolic graphs for depicting our surroundings (or any place of interest), brings the needs of an efficient media for publishing these important tools for depicting our surroundings. The internet with its independency of any hardware or software restriction and with its wide capacity of reaching numerous users, qualifies it in being the best medium for publishing maps. Furthermore the internet can add a significant value to maps usability by adding dynamic and interactive capabilities into maps functionalities.[25]
Because of the convenience of embedding maps in the internet the concept of Web mapping has been produced as a separated discipline for addressing the complete integration between the maps functionalities and the web environment to form a new map service. In other words, embedding maps in the web is no longer considered as an extension of paper maps into the web as an image file but rather more as a complete service. This service include functionalities never been available before as maps modification, maps integration, spatial data querying and other new services that exceed the traditional use of maps on the web.

2.2.7 Web-Cartography

Moving from the traditional way of presenting maps in the form of paper sheets to the modern form of on-screen displays that are reachable by web users, have triggered the process of developing new principles for map design and map layout to fit this new functioning inside the web environment. This developing process includes modifying the map basic elements such as graphs, colors, symbols, fonts, legends and other map basic elements.
In addition to modifying maps basic elements, the developing process includes the embedment of various web technologies such as the advanced user interfaces. The modern web user interfaces provides the user with an interactive environment that receives the user's requests

and returns the desired responds. The responds can be sent in forms of hypertexts, graphical objects or maps. Going further more into embedding other web technologies, the latest web applications has included new communication tools that exceeded the usual readable textual form or visual graph form to include communication tools such as sounds files, animations and videos. Embedding these new communication tools in the map design foster and facilitate the user's perception process of the maps data.

The process of embedding web technologies in maps design is not limited only for exploiting web tools and techniques, the process also embeds other technologies related or embedded by the web technology. As an example, lots of modern computer graphics techniques already exist as a main element in modern web maps design. Computer graphics techniques such as transparent display become one of the fundamental tools used for displaying multi layers maps. Also the transparent display has been used frequently for adding a sense of depth for maps containing graphs of overlying objects. Similar to transparency, shading is also a fundamental tool used for designing 3D geographical displays.

However, even with this promising evolve in embedding web technologies and computer graphics in maps production, the embedding process can still be hinder by few drawbacks related to hardware issues as screen resolution, downloading speed or browsing capacity which forces the map design to contain less information and to present less contents for the purpose of lowering the map size and speeding the map downloading process. These drawbacks results in producing less aesthetic maps with low content of information.

For addressing the lightning phenomenon case, no need for providing details explaining how to compensate web mapping drawbacks using techniques such as generalization. The main focus will be on using web techniques related to interactive capabilities to provide an interactive web analysis tool. Also the focus will be on using related computer graphics techniques related to transparency effect for visualizing the phenomenon in 3D displays.

The fallowing paragraph provides details concerning the most used techniques in designing web maps:

1-Priorty: since web maps have to be limited to relatively small sizes, priorities have to be assigned to the basic objects presented by the map to determine their level of appearance. The map object which presents the main theme of the map has usually the priority for permanent appearance. Otherwise, if the object represents a minor detail in the map, it will be assigned with low priority level and will be displayed temporary according to specific event (e.g. when the mouse crosser passes on the hidden object node) or by a user request. Usually the low

priority level objects are structured inside a drop list or a popup window, where the user can select the option to view these hidden objects.

2- Maps scale: usually web maps have no fixed scale; the computer adjusts the map scale automatically according to the zooming level. However, a certain scale could be assigned to the map during the process of map printing.

3-Zooming: zooming is a process used to gather more visual details of the map objects by applying sequent enlargements on the map objects size, this process usually know as zoom in. Zooming also includes the process of obtaining general view of the map objects, by visualizing the entire map objects regardless of their appearance sizes inside the map frame. In web maps, the zooming process depends basically on the map format. For example, if the web map was presented originally in an image format, the zooming process will depend on the resolution the map was produced in.

Web maps can be presented in various formats, and every format affects the zooming process in a different way. One of the most addressed topic in cartography these days, is the principles of generalization, which revolve around addressing the zooming process of maps in vector format.

4- Map Colors: the most important element when addressing map colors is the determination of the scaling concepts for measuring these map colors. These concepts are necessary for selecting the right color combination for represent the different objects that appear in a map; in the following are the basic concepts for scaling the map colors:

- Hue: represents the comparison between the map color and one of the primary colors: blue, green, red or yellow. The goal of this comparison is to define the degree of similarity or difference between the two compared colors.
- Saturation: represents the amount of color intensity.
- Brightness: represents the degree of brightness (from fully brightness to complete dimness).

Most of the new visualization techniques such as: transparency and shading are implemented by modifying colors values according to the previous color scales. For example, when addressing the transparency technique used for visualizing hidden objects in a particular display, which is very similar to the foggy view technique (the objects appears faded as if there is a gray filter abstracting the direct vision of these objects), the technique involves changing the saturation value of the objects colors in the display front scene and background. The objects in the

display front scene will be given low intensity values and the objects in background of the display will be given high intensity values, leading to the visualization of the hidden objects behind the front scene objects.

As in the transparency technique example, the shading technique is also done by using color scaling concepts. The shading technique which is used for adding an extra sense of depth to the display depends on raising the contrast between the presented objects in the display and their back ground. This process of raising contrast can be done by applying three steps: first, considering the color that appears in the object facade as the initial value for the brightness scale. Second, decreasing the color brightness that appears in object sides to a faded value, if the object doesn't contain any sides, the fade color will be directed toward a certain angle for providing a sense of 3D displaying. In most cases the brightness decreasing angle is directed toward the south east of the display, since the illumination in most cases is directed from the North West.[26]

Later in the discussions of the methodology for visualizing lightning storms phenomenon, the transparency technique will be embedded and implemented in the visualization process. Most of the visualization techniques provided by the application, depends on the transparency effect to clarify the view between the overlapping objects in 2D and 3D displays.

2.3 Similar web approaches

After discussing the importance of exploiting the advanced web technologies in presenting cartographic information, this section will emphasize on some of the attempted approaches for implementing these web technologies in a form of an interactive web application. The approaches were oriented to exploit web technologies in producing an efficient web visual analysis tool that can be use in addressing a certain geographical phenomenon and analyze its main characteristics.

The following discussion will present four approaches for addressing four different natural phenomena. The discussion will focus on the approaches creative process for comprehending the challenges of each case of interest, implementing the proper solution and applying the feasible techniques.

The ***first study case***, presents a design for a web application to provide an interactive tool ,the study case where done by Tom Auer, Alan M. MacEachren, Craig McCabe , Scott Pezanowski, and Michael Stryker. The researchers were addressing the plant diversity patterns in the state of California by designing A *web-based client–server interface* that can handle large amount of spatiotemporal point data *for mapping and exploring flora observation data(HerbariaViz)*.[27]

The researchers' vision were originated from the new perspective toward Web-based map applications, as tools for facilitating the integration between spatial dataset and spatial data analysis, that can provide capabilities in querying, displaying, and interpreting point data in space and time. In addition to these capabilities, the researchers fined in the web mapping technologies an appropriate domain for adopting the OGC open standards. This adopting capability will insure the existence of interoperability between the web mapping services gathered and presented inside the researchers' web application.

For managing the explorative analysis process the researchers implemented a *temporal and spatial aggregation approach.* The approach aggregates the presented data during the exploration activity. The aggregation is done ether with respect to the time of data capturing or with respect to the points' locations. The result of the aggregation process was the prevention of any visual overloading in the main display when exploring data with long time periods or with vast content of spatial information.

Finally for improving the functionality of the web application the researchers have succeeded in utilizing various open sources tools such as Flex applications, GML ,GeoServer and PostgreSQL.

The ***second approach*** has a unique advantage of allowing users to interactively perform on-line data analyze, with the possibility of processing the data according to predefined statistical and geo-statistical methods. Furthermore the approach has provided the capability of producing graphical documents such as semi variograms, contour maps and cross-sections.

The researchers Descamps, Therrien and Therrien have pointed in their research (interactive and open approach for the analysis and diffusion of Geoscientific data)[28], to the limitation of the static documents for answering potential users' queries, or providing new detail about the raw data more than what already have been processed and mapped.

These limitations have directed the researchers to the necessity of avoiding static documents in earth sciences data diffusion and replacing them with interactive documents provided by web applications.

However, to avoid using static document when diffusing Geo data the web, the researcher had to deal with many challenges and difficulties. These changes and difficulties originate from the need of implementing many geo-statistical interpolations processes on the spatial data and later fallowed by complex representations processes. This difficult implementation process hinders the efforts for developing a user-friendly application that can be used by a wide sector of users. The user will be forced to deal with complex formulas and technical difficulties when trying to extract information from the spatial data. To overcome these challenges, the researchers developed a Web-based computer application (APERTA) with functionalities for allowing user to extract spatial data from several databases, analyzing it with statistical and geo-statistical methods and later crating professional maps and plots without the need for dealing with complex formulas or knowing unnecessary presentation technicalities.

As an example for preventing the user from dealing with unnecessary operations when running the application, the researchers have provided the user of their application with an adequate controls (checkboxes, textboxes, radio button and drop-down lists), that can perform and apply the requested formulas and representations automatically without the need of any interaction from the user with complex query languages or software commands.

The researchers also have also avoided the complexity during the development of their application. The researchers focused on using open sources software and tool kits in developing the application such as Apache Web server. This usage of open sources for developing their application provided the users with a web tool that is independent from any software or hardware restrictions.

The ***third case*** addressed by Jaakko Kahkonen , Lassi Lehto , TiinaKilpelainen & Tapani Sarjakoski [29], presents an example for using object-oriented techniques for developing an Interactive visualization application. The application was developed using Java programming language for building the main functionalities. The researchers have pointed in their research to some important matters related to the modern use of web technologies. One of these important matters is the new trend in using the Internet as a geographical information distributer especially with the modern concepts of digital maps and GIS technologies. Another important matter is the problem of visualizing geographical phenomena on the web when using static documents. The researchers pointed out to the important of utilizing the Web as a dynamic tool for presenting and distributing geographical data.
As a result of the previous research notes provided by the researchers around the internet and the importance of using it in distributing geographical data, the researcher have implemented their interactive client application *NetGIS* (Java-based spatial Web browser) using the web technology.

The ***forth case*** presents an interactive web application with capabilities of providing near real-time information for motoring and forecasting marine pollution. The study case addressed by the researchers Kulawiak, Prospathopoulos, Perivoliotis, Łuba, Kioroglou, and Stepnowski [30], has pointed to the critical situation threatening the environment in the Mediterranean Sea. According to the results driven from space borne radar detection and remote sensing data, a dramatic shipping pollution caused by unauthorized discharges was detected. Adding the possibility of a major accident or a natural hazard event to the current pollution situation, the pollution levels will reach a crucial amount and will danger in the environment in the Mediterranean. With this threatening situation, the researcher focus their efforts in developing a web service for monitoring and forecasting the pollution levels in the Mediterranean environment, hoping that this service could help the authorities in performing an effective responses during major pollution events or accidents .
The researchers' approach provided a Web-based Geographic Information Systems (Web GIS), with analysis and disseminating functionalities. The web GIS approach was developed to be operated independently from any traditional GIS desktop application.
The implementation has included the usage of many open sources for software services such as Apache tomcat (including JRE libraries, java servlet engine and JavaScript coding for browser), Map server, Geo server, XML and HTML, with the implantation in one part of the research of ESRI technologies (ArcIMS).

Chapter 3

Data description

The lightning data were originally investigated under a desktop application [31]. In this desktop application the processed raw data were provided by the European lightning detection network (LINET). LINET project was initiated by the Department of Physics at the Technical University of Munich. Later on the project was handed to nowcast GmbH in 2006 for continuous operating.[4]

With 130 sensors distributed in 30 European countries, the lightning detection network provides 3D spatial data for locating the emission altitude and the X, Y position of lightning strikes. The estimated accuracy for this 3D spatial data was about 150 m for the X, Y positioning and1 km for lightning emission altitude.

The accuracy of LINET 3D spatial data has been examined and compared to the 3D special weather radar data and to the NASA special 3D-lightning detection network with very high frequency (VHF). The result of the comparison between LINET data and the data from the 3D weather radar and NASA VHF results in a good match and insures the reliability of LINET data for detecting and evaluating lightning storms.

The lightning data were collected on the twentieth second of July from the year 2010. The range of the collected data extended between Latitudes = 47° – 50° North and Longitudes = 10° – 13° East, covering the Upper region of Bavaria (Germany). The collected data consisted of 34809 lightning points; each point represents a lightning discharge.

Figure1 provides a representation of the lightning point distribution over the upper region of Bavaria. Originally, this representation was build using the desktop application to illustrate the limitation of extracting data from such a simple representation.

The following chapter will provide the methodology for building a web application that can provide more efficient data representations. The provided representations will help in extracting valuable information from the lightning point data.

[4] https://nowcast.de

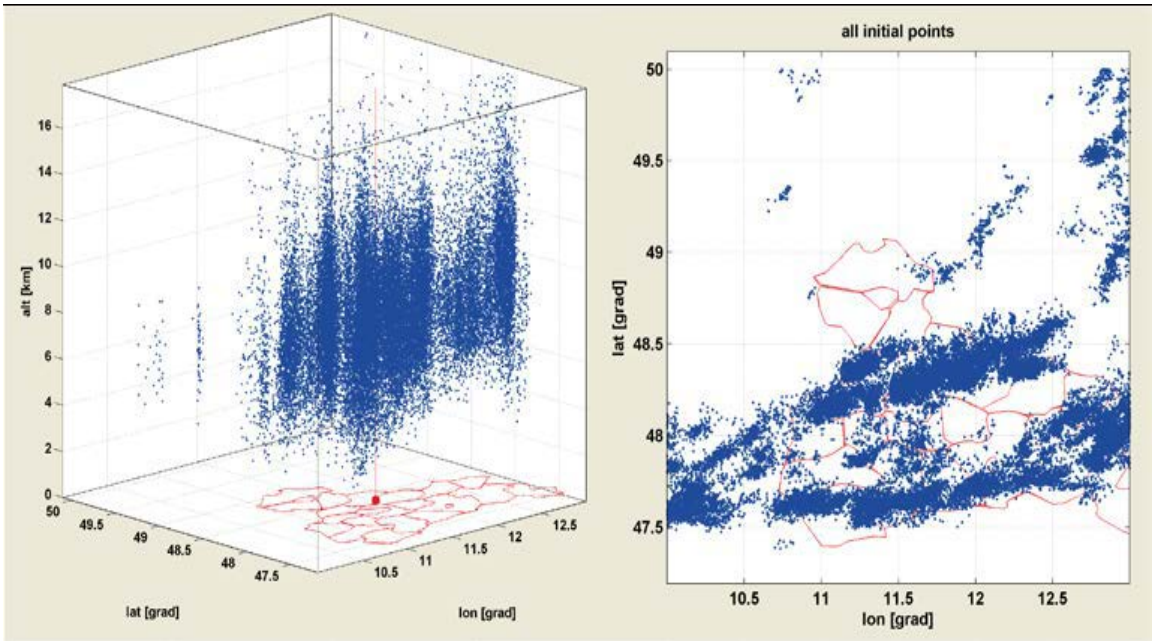


Figure 1: Lightning data distribution over the Upper region of Bavaria[5]

[5] Stefan Peters, H.-D.B., Liqiu Meng, *Visual Analysis of lightning data using Space-Time-Cube*. 2013.

Chapter 4

Methodology

4.1 Primary design

The primary design of the web application functionalities, general frame and user interface (UI), was envisage depending on a previous research [31]. The research proposed a desktop application for visualizing and analyzing lightning storms data using Space-Time-Cube technique. The desktop application was developed using MATLAB software platform for developing a graphical user interface (GUI).

The main goal of the thesis research (as mentioned before) is to extend and implement the current visualization and analysis desktop application techniques into the web, independently from any software or hardware restrictions. Therefore the web application implementation will depend only on open sources (for providing software technologies), for designing and developing web applications.

4.2 User interface

As mentioned previously, the web application design will depend on a former desktop application design. The main theme of the desktop application design (Figure 2) is to provide the user with an interactive visual analysis tool. The tool design included the implementation of controls for selecting and manipulating different visualization methods such as zooming, rotating or adding multiple displays. The tool design also provides the possibility of visualizing the lightning phenomenon according to a timeline.

4.3 Basic user interface tools

For obtaining an efficient performance from the user interface; a number of basic visual and analysis tools should be considered in the design. The design should include a main display window such as the dynamic figure, with parallel tools for selecting possible visualization

methods. Also the design should include controls for changing the dynamic figure parameters ether by selection or by user inputs.

The design of the application tools shouldn't stop at the general concepts for designing visual analysis tools, but should exceed that to a more specialized and case cantered design.

In the following section, a number of control tools contained in the application design are addressed. The following section will also provide a discussion about designing the web application tool to handle the lightning phenomenon specifically.

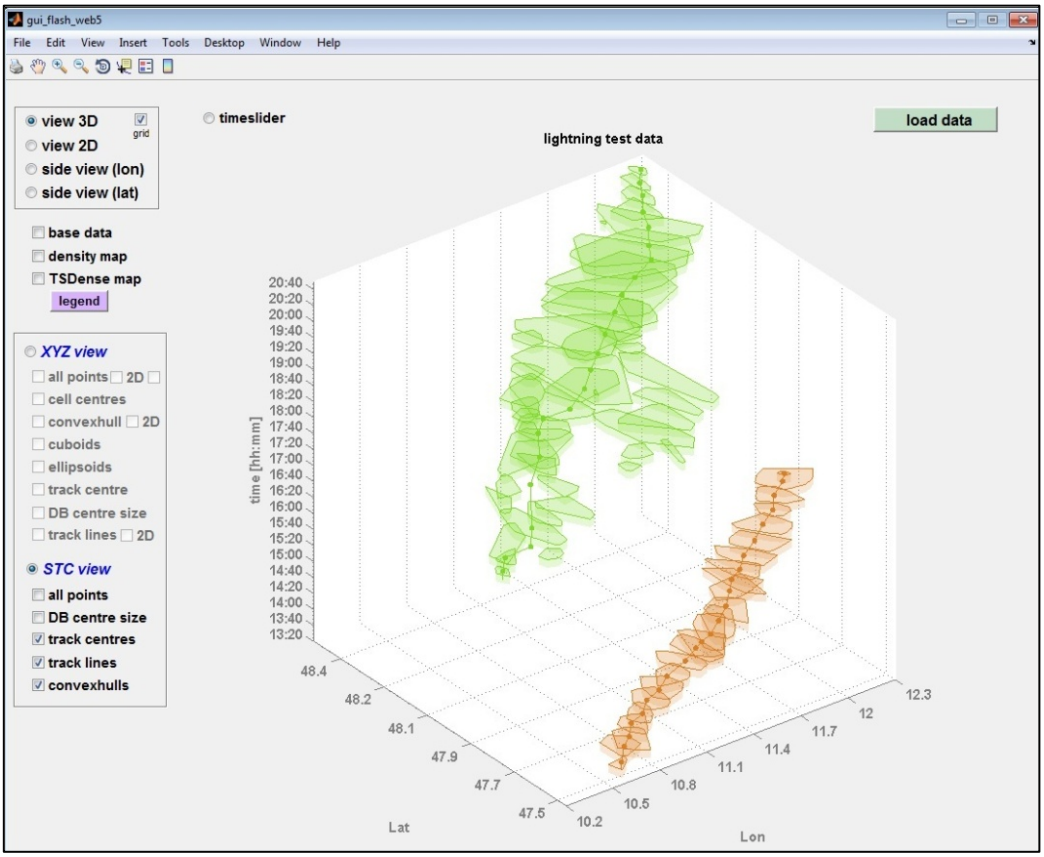


Figure 2: user interface

4.3.1 Dynamic figure

In the process of designing a visual analysis application, the Dynamic figure can be considered as the core of the visual and analysis tool. It is simply a window for the user into the lightning data. With the dynamic figure, the user can visually: test, examine, reveal and extract the hidden patterns and unknown characteristics of the lightning phenomenon. Without the Dynamic figure, the user choice will be limited to test the data in a mathematical model

with results represented as numerical values. The Dynamic figure is the first significant step in the procedure of designing and implementing an efficient visual analysis application.

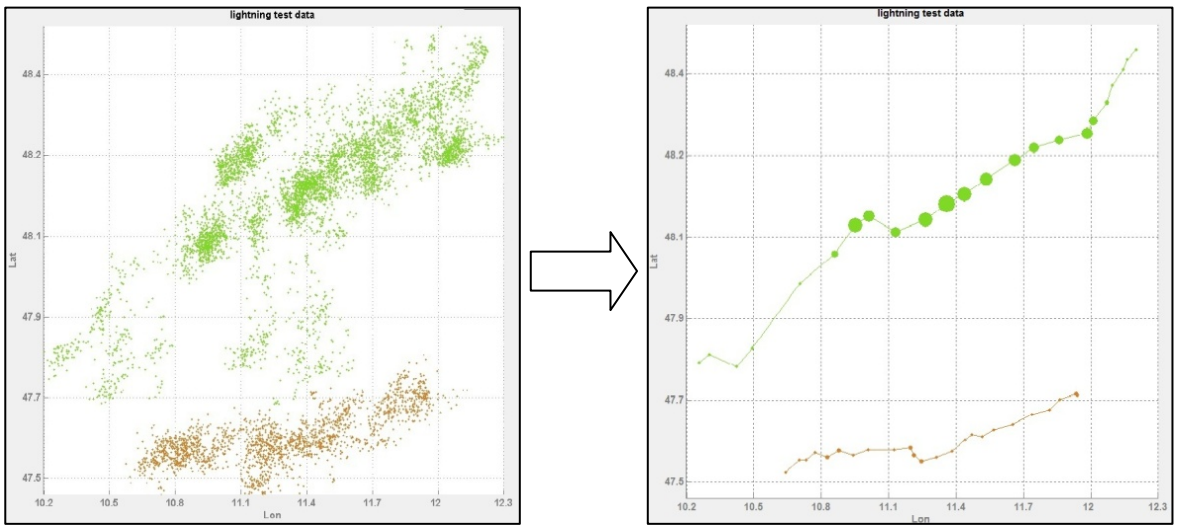


Figure 3: Different data representation and visualization with the dynamic figure box

4.3.2 Radio buttons

As mentioned before, one of the visual analysis application design aspects is to develop and build a tool for handling the lightning storm case specifically. For serving this purpose, the design has included eight different visualizing methods for presenting and mapping the lightning point data. This design of multiple choices for visualizing the lightning point data has provided the user with a comprehensive solution for testing and examining the lightning point data.

To select between these multiple visualization methods, a proper control tool should be embedded in the application design. The tool should serve the purpose of presenting the multiple visualization options with great amount of flexibility when choosing between these visualization options. As a result of this design demand, the radio button tool has been chosen to provide the user with a selection tool for choosing between different visualization options.

4.3.3 Time slider

As mentioned in chapter 2, section 2.2, in discussing the temporal analysis topic, the point data were divided into separated clusters based on time division. The final number of time

periods was found to be 71 periods. According to this division the user may move from one time period to another to observe the changes in the phenomenon characteristics when shifting through time. For providing this capability in real practice, a time slider tool was implemented inside the visual analysis application. Using the time slider tool, the user can shift between different time periods by simply clicking the forward or backward buttons of the timeline scale contained inside the time slider tool. The shifting can be done ether step by step or by big shifts in the timeline scale.

4.3.4 Combined view

The combined view represents a combination of multiple visualization options at the same time. When combining two or three visualization options in the same display, the different techniques provided by each visualization option individually, are grouped into one technique, giving the user a more powerful tool for examining and understanding of the visualized phenomenon.

4.3.5 Space time cube

Combining the time slider tool, with the 3D view, provides the users with a technique known as: space-time-cube. The implementation of space time cube concept has significant importance in gaining better understanding of lightning phenomenon characteristics with the consideration of time changes.

4.4 Selecting web developing software platform

In the process of selecting an appropriate developing software platform for building the web applications, the focus should be on the level of programming capability and flexibility the software platform can provide. However, as mentioned before, the web application design originally depends on a parallel desktop application design, with the intentions of extending some of the desktop application functionalities to the web application. Putting in mind this dependency of the web application design on the desktop application design, the developer will be restricted in his choices for choosing the developing software platform, to whatever developing software platform that can be combined with the original developing software platform in the desktop application.

The desktop application was build and developed using MATLAB developing software platform. One of the advance features in MATALB is the possibility of implementing and embedding any basic functionality or method defined by MATLAB under the Java developing software platform. The implementation and embedding process are done by using MATLAB builder JA tool. Through this tool, the MATLAB basic functionalities and methods can be encapsulated inside a Java class frame, which is readable and executable by the Java software platform.

This specialty of MATALB in extending its basic functionalities to the use of Java platform had restricted the selection process for the web developing platform in (and only in) Java technologies.

The web application will be built using: Java for building the servlets on the server side, JavaScript for organizing of the HTML files and JSP for providing a dynamic implementation.

4.5 Web application design

After discussing the primary design of the visual analysis web application and the basic functionalities the web application should contain. And also after pointing out to the possible technology used for developing the web application, time now to discuss the actual design of the web application, especially from a technical perspective.

The web application design should provide basically, the same visual analysis functionalities provided by the desktop application design, with also the same level of usability when operating these functionalities by the user. The user should simply enter the URL of the web application on the server and receive a web page with the all necessary visual analysis tools. Also the user should be able to communicate with the server side through this web page easily and smoothly, especially when posting his requests and receiving the responses from the server.

The application design should also prevent any form of complexity during the run time, all of the computing and processing operations should be done completely behind the scene without any interference from the user.

4.5.1 Client to server model

For implementing the pervious criteria of functionalities and performances, the Client to Server model represents a suitable technique for matching the implementation demands.

The client to server model provides the functionalities of computing and processing the user requests on a remote computing recourse know as the server. The computing and processing functionalities provided by this server are known as the server services. The server services can be invoked by the user through personal computing unites, normally these unites are PCs or workstations. The user computing unite (which posts the use inputs to the server side) is known as the client side. The connection between the client side and the server side is done usually by global connection network as the internet, or in some cases, by local connection network (LAN).

In the practical implementation of the Client to Server model, the developer can chose various implementing paradigms for the Client to Server model, but the ultimate paradigm for performing an efficient application processing, is the peer to peer processing paradigm [32].

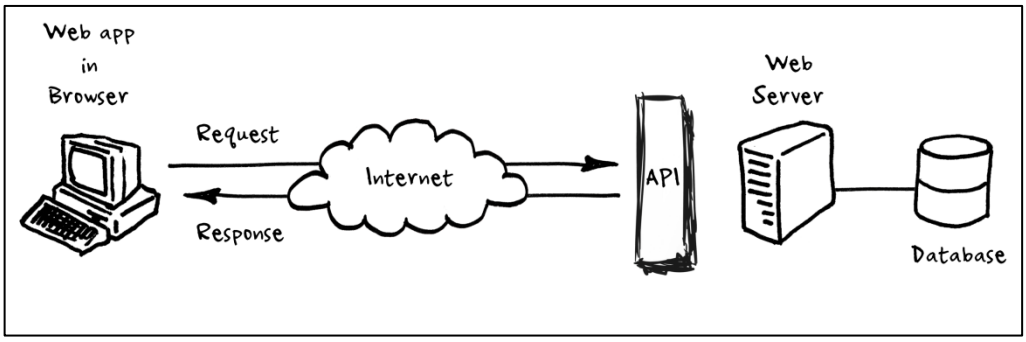


Figure 4: client to server model [6]

4.5.2 Programming techniques

The process of building and developing the web application will move through various stages of programming, testing and evaluating to determined how much of the application development process has been achieved for filling the design requirements.

In the first step toward building the web application, the MATLAB basic functions and previously developed codes (used in building the desktop application) should be modified and edited, to fit the implementation requirements in the web application.

Since the web application is developed trough Java software platform (as mentioned previously in section 4.4), the implementation of MATLAB basic functions and previous devel-

[6] http://thefloppydisk.wordpress.com/2013/05/08/how-to-build-a-restful-web-api-on-a-raspberry-pi-in-javascript/

oped codes inside Java platform, has certain demands and restriction, especially that the developing environment is moving from MATLAB platform, which represent a fourth programming generation, to a less developed programming environment such as Java, which represents a third programming generation[7].

Unfortunately, this non uniformity in programming capabilities, have caused a technical problem in using the original desktop application functionalities. A new task is added to the developer workload, the developer has to reprogram the whole desktop application codes, so the MATLAB codes can fit for the use of the java software platform.

In the following, a detailed example for reprogramming the basic functions of MATLAB and the remarks and notes that needs to be taken in mind when extending these functions to java programming platform.

4.5.2.1 MATLAB functions for Web application

As mentioned before, the desktop application provides the users with eight different options for visualizing the lightning storms data; each option is implemented using MATALAB codes. Executing and Running these MATLAB codes in a MATALB Compiler Runtime (MCR) gives the estimated results properly and efficiently. However when extending these MATLAB codes in to a Java software platform and later trying to run these codes under Java virtual machine, the application simply crashes and an error message will be sent to the user interface.

The differences between the runtime environment in MATALB and Java software platforms, and the differences in their developing capabilities levels, forces the developer (as mentioned previously) to modify and reprogram MATLAB codes to be more fitting for Java developing criteria and Java execution requirement.

For extending MATALB basic functions and codes to Java software platform, the developer has to use (and only use) the similar programming principles used for structuring codes in both developing tools. This process includes using the similar methods between the two software platforms for defining (declaring) variables and also the similar methods for defining arrays.

[7] However, even with this difference in developing capabilities, the Java platform can still provide a powerful developing functionalities (for other developing requirements) more than what most programming platforms can provide. For this reason the Java platform will still consider a suitable tool for developing the web application.

For example, *structure arrays* are a common technique used in MATALB developing environment for structuring different types of data in one file. This is similar to the structuring means used in other developing platforms such as C, but it's not used inside Java developing environment.

For declaring a structure array inside MATALB, the `STRUCT` declaration statement is used to arrange multiple data types in one structure array. The process begins by declaring a substructure form known as field. The field represents a container for containing only one specific type of data. When a group of fields are combined together, a structured array is defined (in the simplest manner of defining a structured array).

When moving from MATALB specifications for declaring different data types to Java specifications, a technical problem occurs, since Java provides a declaration for only one data type per one array. In other words, an individual array declared in Java, cannot hold two different types of data at the same time. Most of Java arrays are declared in a similar manner to variables declarations, ether as `integer`, `double`, `string` or any other data types that can be declared in Java.

The only way to have an array with multiple types in Java is to declare an array of the type `object` and nest each array with different type or different dimension as an object element inside this array of objects. Later by using a `for` loop statement, the nested arrays can be called and casted to their original types e.g. `integer`, `char`, `double`...Etc.

To overcome this non similarities between MATALB and Java, the developer has to implement a matching process that includes (and only includes) the use of common techniques for declaring variables or defining statements by both platforms (statements such as: for loop statement, while loop statement and any other statement that is commonly used by both programming languages).

In the following a MATALB code (scriptlet) that has been edited to fit in the Java declaration criteria.

```
function  plottingAllPoint3D =AllPoint3D(arg)
f = figure;
for k=1:7101
    i=arg(k,5);
    if i ==1
        d(k)= plot3(arg(k,1),arg(k,2),arg(k,3),
'*','markersize',2,'Color',[0.85 0.50 0.15],'visible','on');
        hold on;
        view(3);
    end
    if i  == 2
        d(k)= plot3(arg(k,1),arg(k,2),arg(k,3),
'*','markersize',2,'Color',[0.50 0.85 0.15],'visible','on');
        hold on;
        view(3);
    end
    grid on
end
plottingAllPoint3D=webfigure(f);
close(f);
end
```

Figure 5: Matalb example code for plotting lightning point data

The previous MATALB code declares a function for plotting point data in a three dimension perspective, the function receive the inputs as an array with predefined size (the dimensions should not exceed the declared limits of (7101 * 5)).

After passing the point data array to the plotting function, the `for` loop filters the array columns to separate and classify the points according to their assigned graphical values. For every point data record, if the fifth column value (which represent a classification code) is equal to: 1, the point will be plotted with the color value [0.85 0.50 0.15] and if the fifth column value equals: 2, the point will be plotted with the color value [0.50 0.85 0.15].

It's important to mention here that this plotting function is designed originally for plotting a static array of certain values (lightning storms point data) and not intended to be used for plotting dynamic arrays with non predefined values.

4.5.2.2 JAR files

The JAR term is an acronym for Java Archive, which represents an aggregated package of java classes, metadata and other sources files that can be implemented on a Java virtual machine (e.g. apache Tom cat).

To extend the MATALAB functions to Java software platform, the MATALAB functions should be encapsulated inside the JAR file, which contains the necessary classes and methods for invoking and executing MATALB functions on Java software platform.

Building the JAR file is an automated operation provided by MATALAB developing software platform and can be modified and edited by the user (structuring the classes and methods included in the JAR file).

4.5.2.3 Need for mySQL

For visualizing the lightning storms data using MATALB software platform, a number of pre-processing operations have to be performed on the lightning raw data; otherwise the direct visualization of the raw data will result in a poor presentation of the lightning phenomenon. However, the fallowing section will not focus on discussing the preprocessing operations details, but rather on the structuring of the preprocessed data inside the desktop application, to reveal the need for using mySQL.

In desk top applications, the data are preserved in particular space on the memory, without any logical connection between the data entities. However in web applications, the server side is responsible for preserving the data in a totally different and unique manner. Usually the data should be structured in a logical pattern with defined relations between its entities, before it can be stored by the server. This form of structuring is performed by using the concept of: *relational database*. The relational database represents a logical structuring form for containing data. (This is necessary to perform operations such as: data query and data retrieve).

Due to the differences in structuring and preserving data between desktop application and web application, the uses and implementations of relational database management system for managing the lightning point data and passing it to the Java platform, becomes a necessary approach.

There are several open sources that can provide an accessible relational database management system; one of the most widely used open sources is the mySQL.

The Implementation of mySQL is supported with a powerful user interface that can ease the administration and management process of the relational database and prevent the user from dealing with unnecessary coding commands. This is done by automatically generating the code that represents the database structuring through the user interface tools. The code

generation process can be done completely behind the scene; with the option for the user to generate the code manually.

4.5.3 Dynamic web pages

For designing a dynamic web page, the same HTML markup language used for designing the statics web pages is used. For adding the dynamic performance, a number of web developing software platforms should be included and a certain technique combining these developing software platforms should be implemented.

4.5.3.1 JSP

The JSP term stands for **J**ava **S**erver **P**age, which is a technology used to generate a dynamic HTML content on the server. The core of the JSP technology is the possibility of implementing the JSP files as a servlet object on the server side. Going more into details, when a JSP file (originally stored on the server side) is invoked by the client side, the server runs this JSP file as servlet object and preserves a space for it on the memory. Later the server observes the additional client requests and applies the new changes on the JSP object that has been already loaded on the memory, if no farther requests were sent from the client side, the server will keep loading the JSP object without any changes.[33]

4.5.3.2 Java Script

JavaScript is a high level programming language for developing web pages, capable of adding dynamic performance to a web page. Originally JavaScript was based on Java syntax with some classes and functionalities derived from Self and Scheme programming languages. Recently JavaScript becomes the most commonly used language for interpreting web pages and interacting with users requests on the client side. Due to the object oriented programming style of JavaScript and also due to the capability for adopting the asynchronous schemas, JavaScript becomes widely use for developing web application.[34]

For building the lightning web application, the JavaScript libraries and functionalities will be embedded for organizing the HTML file behavior when receiving user requests. With the unique JavaScript APIs for handling graphical user interface implementation, the JavaScript technology will serve significantly in implementing the lightning storm web application. In addition to that, lots of browsers developing tool kits are designed to function on a JavaS-

cript software platform, which facilities embedding various tools for communicating with the application users, leading to a better design for the application interface.

Dojo

Dojo is a rich open source for providing web functionalities over a JavaScript soft ware platform. Manly the Dojo tool kit is used for developing web pages and web applications. With its capability of providing an end to end solution and with its adoption of java script standard library for implementing different widgets and also with its flexibility for testing the frame work during the development process, it becomes the developers most used tool kit for building browser pages.

The Dojo tool kit is basically portioned to:

Base: for providing foundations for every module in the package.

Core: build on the Base for facilitating processes as parsing widgets, adding animation effects and other advanced features in the goal of optimizing JavaScript functionalities on the browsers.

Dijit: represent a library built directly upon Core, that can be used and implement without the need of writing any JavaScript code.

DojoX: collection of modules that still not officially considered as stable or fitting for Core or Dijit.

Util: represent a JavaScript framework testing unit. Special tools can be built using Util to provide custom versions of Dojo that have different production settings.

The Dojo capability of providing functionalities for fostering web pages developing, keeps on going with the availability of utilizing dojo kit in AJAX scenarios. Later this utilizing availability will becomes a necessity in developing the lightning storms web application and one of the major reason for embedding Dojo techniques in the web application. [35]

4.5.3.3 Java Srevlet

The servlet is the mean engine for implementing and executing any predefined functions and methods on the server side. Without this engine no implementation will be possible on

the server and none of the server functionalities can be reachable by the client side (the browser).

In other words the servlet is an extension of the server functionalities to the client side that can be invoked and manipulate according to user requests. Usually most of servlet engines are developed using Java software platform since Java technology provides many classes and functionalities for organizing the communication between the server side and the client side.

For executing these java classes and functionalities, a java execution platform known as java virtual machine (JVM) has to be installed to contain and run these classes. Fortunately Java technologies are supported by most server operation systems, which make the Java servlet a suitable engine for developing and running web applications.

The advantages of using Java servlets goes on to include many features as the power of exploiting all of Java APIs for improving access and connectivity to networks ,URLs and data bases(which will play a major role in using mySQL database in our application). Another advantage of using Java servlets is their efficient performance when been loaded on the server as a respond to user calls, they remain on the server memory as single object without the need to be interpreted or processed every time the user submit a call for a servlet , which leads to immediate respond to user requests and reflect lightweight method of invocation.[36]

In addition to the previous advantages, Java servelts usage grantees an added value to other levels of efficiency such as: information protection and safty, elegance dealing with APIs and handing the coding scripts (since its object oriented programming platform) and the ability of integrating with server functionalities (logging, authorization…ETC).

As mentioned earlier, for running the servlet java classes, a container has to contain these classes (JVM). There are different types of servlet containers the most important are the *Standalone container*, the *Add-on servlet* container and *Embeddable* container.

The *Embeddable* container is lightweight servlet deployment platform, than can be included inside a web application files and during executing time become the real server. Apache tomcat server is one of the famous open server functionalities providers, and also supports the *Embeddable* container technique. This technique will be the backbone for running and testing the trials versions of the lightning storms web application, since this technique allows the developer to embed the server files in his application and perform real time server services without the need for any form of connections to a remote server to run the applica-

tion services. After the testing and developing phases ends, the whole application files will be including on the servlet container and will be implemented on the server side as JAR file.

4.5.3.4 AJAX

AJAX is technique for combining a group of web developing and programming technologies to work in a manner similar to what the user can experience from a desktop application.

The term *AJAX* stands for Asynchronous JavaScript And XML file, which can also include other technologies related to web development such as: DOM (document object model), HTML and CSS.

In practice the AJAX approach follows certain steps when running the web application, the process starts by making a call to the AJAX engine from the JavaScript control tools in the user interface(HTML file), the AJAX engine (which is a JavaScript object: XMLHTTPrequest) performs asynchronous call to the server side, the server responds to the call and runs the necessary calculations and connections to database, later the AJAX engine receive the server respond and pass it to the user interface without loading the whole HTML page.[37]

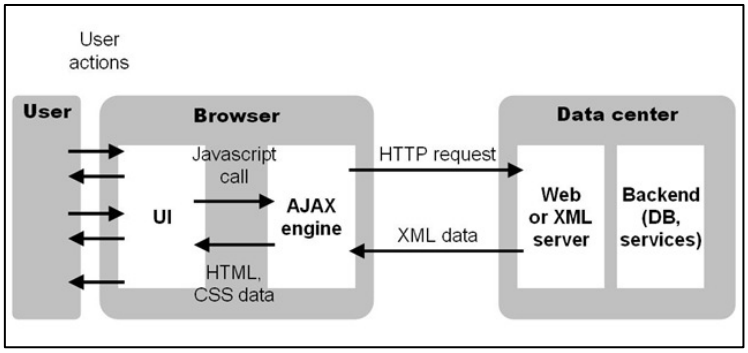


Figure 6: AJAX Diagram

Chapter 5

Implementation

5.1 Building MATALAB Codes

As discussed before, when moving from MATLAB which represents fourth generation of programming and developing software platform, to Java which represents third fourth generation of programming and developing software platform, this process hinders the developer progress and limits the flexibility of his developing environment. The major drawback comes from the non unified standards for syntax structure and programming statement between the two platforms. To overcome this case of non uniformity, the developer should address both software platforms techniques and try to find up the common programming standards that can be interpreted in both platforms.

In the following a MATLAB code example that have been edited to fit inside Java interpreting environment.

Plotting lightning point data in 3D example

In the example code (Figure 7) a declaration of a function for plotting point data in 3 dimensional representation, the point data is passed into the function as an array `(arg),` the function then declares a graphic figure object `figure` with the name `f,` a figure object is a window on the display screen that MATLAB uses for displaying the graphical output. Since the passed data to the function is the lightning data, the function is designed specifically to map this data in the application, the `for` loop is defined to perform 7101 iterations which matches the number of records for the lightning point data.
The function plots the data in 3D by using the default MATLAB plotting function `plot3.` Specific colors are assigned to the mapped points according to their classification records, if the point record equals 1, it will be colored as `[0.85 0.50 0.15],` if the point record equals 2, it will be colored as `[0.50 0.85 0.15].`
the last part of the function declares a `webfigure` for displaying the mapped points, the web figure is used for embedding MATALB default figures on to the web browser, with the possibility for the users to manipulate the figure functionalities as if they are using the MATLAB software.

```
function  plottingAllPoint3D =AllPoint3D(arg)
f = figure;
for k=1:7101
i=arg(k,5);
if i ==1
plot(k)= plot3(arg(k,1),arg(k,2),arg(k,3),
'*','markersize',2,'Color',[0.85 0.50 0.15],'visible','on');
hold on;
view(3);
end
if i  == 2
plot(k)= plot3(arg(k,1),arg(k,2),arg(k,3),
'*','markersize',2,'Color',[0.50 0.85 0.15],'visible','on');
hold on;
view(3);
end
grid on
end
plottingAllPoint3D=webfigure(f);
close(f);
end
```

Figure 7: Plotting lightning point in 3D code

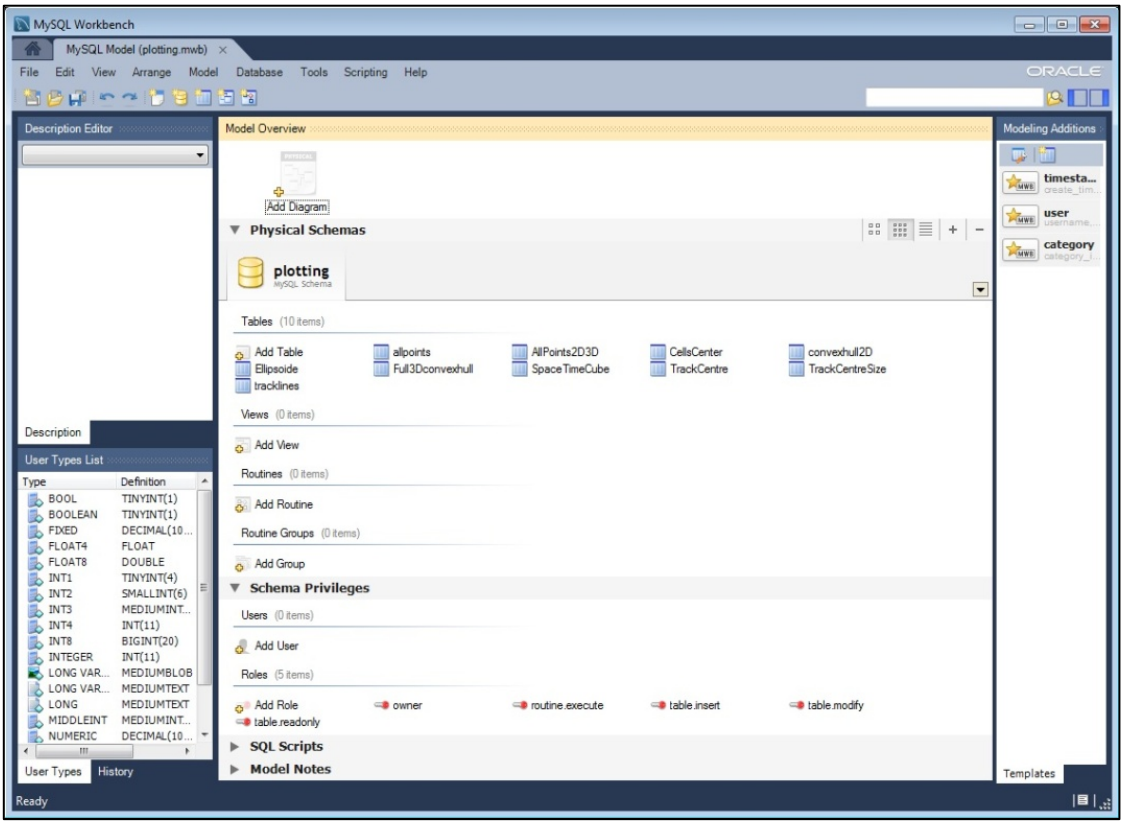
5.2 mySQL implementation

For implementing the relational data base provided by mySQL, two steps are formed:

First: building the database using mySQL software (MySQL workbench 6.0).

Second: connecting the database to the servlet engine. (JDBC, Java Database Connectivity).

For the first step, the mySQL workbench provide the developer with a comprehensive tool for building and managing the data base, without the need of dealing with the any manual coding processes. The mySQL workbench enables the user from designing his data structures (tables) and editing his data entities, through a sophisticated interface. The interface contains multiple tools and options for structuring and editing the data easily and efficiently. After finishing the steps of generating tables and editing their contents, the coding process is done automatically behind the scene without any interference from the user. The final database structure will be loaded and saved under a directory chosen previously by the user. It's important to mention that mySQL workbench still enables the user from building and structuring his database manually, if the user preferred to go on with a manual developing method.



mySQL workbench UI

For the second step:

Because the database is built outside of the web application, a certain connectivity procedure should be implemented to establish the connection between the application and the database.

This connectivity procedure is applicable by using the mySQL application programming interfaces (APIs).mySQL *DriverManager* is the main object in mySQL for providing the connectivity interfaces (*Class.forName* and *com.mysql.jdbc.Driver*), which enables the database connection.

In the fallowing example code, an example of how these two programming interfaces are implemented to perform a connection to the database and to load data from specific table:

```
try
{
    Class.forName("com.mysql.jdbc.Driver");
} catch (ClassNotFoundException e) {
    throw new ServletException(e);
}
double[][] coor = new double[71][5];
try
{
    java.sql.Connection connection = DriverManager.getConnection(
        "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
    java.sql.Statement statement = connection.createStatement();
    ResultSet resultSet = statement.executeQuery("Select X, Y, Z,
Class, Size   From trackcentresize");
    int i = 0;
    while (resultSet.next()) {
        coor[i][0] = resultSet.getDouble("X");
        coor[i][1] = resultSet.getDouble("Y");
        coor[i][2] = resultSet.getDouble("Z");
        coor[i][3] = resultSet.getDouble("Class");
        coor[i][4] = resultSet.getDouble("Size");
        i++;
    }
    connection.close();
} catch (SQLException e) {
    throw new ServletException(e);
}
```

Figure 8: Java code for loading mySQL tables

5.3 JavaScript, JSP, XML and HTML

For explaining the implementation process of JavaScript, JSP, XML and HTML, in building and forming the basic user interface (UI) elements, an important aspect should be pointed out; these technologies should be combined together in certain manner for the UI to be operational.

The combination of these technologies is based on the functionalities of each individual technology and their main use in the field of web application development.

In the fallowing, a detail explanation of the role of each one of these technologies in implementing the UI functionalities:

First: the HTML represents the main container for all these technologies, except for XML which is represented in separated file, but still connected to the other elements during execution time.

Second: JSP (as explained before) enables the developer from adding a dynamic behavior to the web page.

Third: XML used for documenting the main application elements (files) and data structure to clarify these elements and structures for the server during the run time. Without the XML defining and pointing out for the server the basic elements in the application, the server cannot locate, realize or execute these elements even if they already existing on the server.

Furth: JavaScript represents the main engine for organizing the processes of:

a) Receiving user's inputs.
b) Passing these inputs to the server
c) Receiving the server execution result in return.

In addition to the previous processes, JavaScripts also provides the platform for implementing technologies (such as DOJO) that focuses on organizing the user inputs.
In the fallowing an example code combining all of the previous discussed technologies. For each part of the code, an explanation will be provided to point out the used technology in this part of the code and the standard syntax format.

In the first part of this example code, it's clear that the whole document is defined inside HTML tag <html>…</html>. The fallowing head tag <head>…</head>, is declared for providing a container for the JavaScript code. The link tag <link> is used to define a relation between the current document and external styling document used for the Dojo slider im-

plementation. The Dojo implementation is defined and processed inside JavaScript tag <script>…</script>. As result of implementing this Dojo script, the user will be able from changing the dynamic web figure parameter (cell number), in an easy and sufficient way. This provided tool comes in a simple and easy to use form of a horizontal slider:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
pageEncoding="ISO-8859-1" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<link rel="stylesheet"href="dojoToolKit/dijit/themes/claro/claro.css">
<script>
dojoConfig ={async: true,
parseOnLoad: false};
</script>
<script src="dojoToolKit/dojo/dojo.js">
</script>
<script>
var widget;
require(["dojo/dom","dojo/parser","dijit/form/HorizontalSlider",
"dijit/registry","dojo/domReady!"],
function go(dom, parser, HorizontalSlider, TextBox, number)
                    {
parser.parse();
slider = new dijit.form.HorizontalSlider({
name: "slider",
value: 35,
minimum: 1,
maximum: 71,
intermediateChanges: false,
showButtons: true,
discreteValues: 71,
style: "width:300px;",
onChange: function doit(value) {
dom.byId("sliderValue").value = value;
widget = dijit.byId("sliderValue");
document.getElementById("mytext").value = widget.get("value");
document.getElementById("object").submit("mytext");
                    }
                    ,
                  }
, "slider");
                 }
                );
```

Figure 9: example code combining JavaScript, JSP, and HTML-part 1

The second part of the code represent a parsing method for the slider inputs, when the user changes the slider value, this value is immediately parsed to an integer variable by the parsing method. This step is necessary since the servlet engine is defined to deal with user inputs as an integer value, which will be used later in executing Java and MATLAB classes .

```
<%
String sizeStr = request.getParameter("mytext");

int mytext = 35;
boolean sizeSet = false;
if (sizeStr != null && sizeStr.length() > 0
     {

mytext = Integer.parseInt(sizeStr);
sizeSet = true;

        }
%>
</script>

</head>
```

Figure 10: example code combining JavaScript, JSP, and HTML-part 2

As mentioned previously, the JavaScript tag `</script>` should enclose this section of code for providing Dojo Slider tool and parsing its value. Finally, the whole script tag is enclosed by the head tag `</head>`.

The third part of the example code represents the main body of the HTML document contained inside the body tag `<body>…</body>`, inside the body tag the developer can form many declarations for adding important elements to the HTML document such as the `div` tag `<div>…</div>` for defining cretin CSS style to group of elements inside the HTML file.

One of the important declinations inside the body tag is the declaration of the form tag `<form>…</form>`, which is used to handle user inputs to the HTML file. The form tags contains various tag elements for dealing with user inputs, one of the most used tags is the input tag `<input>…</input>`, which is used here specifically for handling the user inputs through Dojo slider.

```
  <body class="claro">
  <div id="slider">
  </div>
  <form id=object method="get">
      <p><input id="sliderValue" name="mytext" type="text"
      data-dojo-type="dijit/form/TextBox" value="<%=mytext%>" /></p>
  <p><input type="hidden" name="mytext" id="mytext" /></p>
```

Figure 11: example code combining JavaScript, JSP, and HTML-part 3

The final section of this code example presents the AJAX technique for sending the user inputs to server and processing the results as of the whole operation was implemented in a desktop application (asynchronously implementation).

Inside the `script` tag a XMLHttpRequest( ) object is declared under the name myRequest,

the object specifies the requesting method for calling the server side, (the GET method)and the URL pointing for the servlet name on the server (sliding123?) with the passed slider value which represents the user input ('sliding123?sliderValue=<%=mytext%>').

The object also specifies the type of the data received from the server side (Text) as respond for the GET method request.

```
<script type="text/javascript">
try
{
var myRequest = new XMLHttpRequest();
            }
      catch (e)
      {
      try
      {
      var myRequest = new ActiveXObject('Msxml2.XMLHTTP');
              }
      catch (e)
      {
  try
  {
  var myRequest = new ActiveXObject('Microsoft.XMLHTTP');
              }
  catch (e)
  {
  document.write('XMLHttpRequest not supported');
              }
            }
          }
  myRequest.open('GET','sliding123?sliderValue=<%=mytext%>', false);
  myRequest.send(null);
  document.writeln(myRequest.responseText);
</script>
```

Figure 12: example code combining JavaScript, JSP, and HTML-part 3

5.4 Java for building application servlet

As mentioned in the previous chapter, the servlet engine is considered to be the main core for running the web application functionalities. in the follow section, an example code will be provided to reveal the main structure of the servlet code. The code will contain the main classes and functions defined in any servlet file. The code also will contain the extra methods and statements used for invoking the MATALB functionalities from the JAR file.
The first part of servlet code is the **import** declarations, which is used to refer to a static member in the web application. The **import** declaration calls the necessary methods (under the available classes) by using their canonical names.[8]

```
package timeSlider.moh.test1;
import java.io.IOException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;

import passingTimeSliderNumber.*;

import com.mathworks.toolbox.javabuilder.MWException;
import com.mathworks.toolbox.javabuilder.MWJavaObjectRef;
import com.mathworks.toolbox.javabuilder.MWNumericArray;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigure;

import com.mathworks.toolbox.javabuilder.webfigures.WebFigureHtmlGenerator;
import com.sun.xml.internal.ws.api.config.management.policy.
ManagementAssertion.Setting;
```

Figure 13: Servlet example code-first part (import declarations)

The second part of the code is a declaration for the main **class** sliding, which implements the HttpServlet subclass. The HttpServlet subclass provides the basic functionalities for the main class sliding to operate on web site (doGet method, init and destroy method…ETC).[9]

[8] http://docs.oracle.com/javase/specs/jls/se8/html/jls-7.html#jls-7.5.1

[9] http://docs.oracle.com/javaee/1.4/api/javax/servlet/http/HttpServlet.html

```
public class sliding extends HttpServlet
{
   private static final long serialVersionUID = 1L;
   protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException
{
   String test = request.getParameter("sliderValue");
   WebFigure figure1 = null;
```

Figure 14: Servlet example code-Second part (declaration of main class sliding)

The third part provides a call procedure for the relational database.[10]

```
  try
        {
      Class.forName("com.mysql.jdbc.Driver");
        }
   catch (ClassNotFoundException e)
      {
      throw new ServletException(e);
        }
  double[][] coor = new double[71][5];
  try
        {
      java.sql.Connection connection = DriverManager.getConnection(
      "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
      java.sql.Statement statement = connection.createStatement();
   ResultSet resultSet = statement.executeQuery("Select X, Y, Z, Class,
Size   From trackcentresize");
  int i = 0;
  while (resultSet.next())
            {
  coor[i][0] = resultSet.getDouble("X");
  coor[i][1] = resultSet.getDouble("Y");
  coor[i][2] = resultSet.getDouble("Z");
  coor[i][3] = resultSet.getDouble("Class");
  coor[i][4] = resultSet.getDouble("Size");
  i++;
            }
      connection.close();
        }
   catch (SQLException e)
      {
      throw new ServletException(e);
        }
```

Figure 15: Servlet example code-third part (relational database connection)

[10] More details provided in previous mySQL implementation section.

The forth part of the cods provide the main declaration of the object calc, that implements the methods defined previously in the MATLAB file (the MATLAB file is already encapsulated inside a Java class).

```
  try
        {
      Plotting calc = new Plotting();
   MWNumericArray size = new
MWNumericArray(Integer.parseInt(request.getParameter("sliderValue")));
  Object[] results = calc.TrackCentreSizeBackup(1, coor, size);
  MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
  figure1 = (WebFigure) ref.get();
  getServletContext().setAttribute("UserPlot1", figure1);
        }
   catch (MWException e3)
      {
      e3.printStackTrace();
        }
```

Figure 16: Servlet example code-fourth part (implementing MATALB functions)

The fifth part of the code provides the declaration for returning the MATLAB web figure to the client side.

```
  StringBuffer buffer = new StringBuffer();
  WebFigureHtmlGenerator webFigureHtmlforStefan = new
    WebFigureHtmlGenerator("WebFigures", getServletContext());
  try

        {
   String outputString =
webFigureHtmlforStefan.getFigureEmbedString(figure1, "UserPlot1",
"application", "600", "600", null);

      buffer.append(outputString);

        }
   catch (Exception e)

      {
      e.printStackTrace();

        }
   response.getOutputStream().print(buffer.toString());

    }
}
```

Figure 17: Servlet example code-fifth part (returning MATALB web figure)

Chapter 6

Results and discussion

Addressing the topic of extending visual analysis functionalities on the web, for addressing a certain weather phenomenon, has introduced the problem of interoperability when combining two different developing software platforms to design and implement the visual analysis application.

The results of the developing process has shown the possibility of extending the main functionalities build by MATALB software platform to the web using Java developing software platform. However, the integration between MATALB and Java was an unfixable process that experienced many technical difficulties. The reason for these difficulties during the implementation is the differences between MATALB and Java in their basic functionality. The MATALB is considered to be a fourth generation programming and developing technology, oriented originally for developing numerical computation applications. In the other hand, Java is considered to be a third generation programming and developing technology, oriented for practical applications development.

In addition to the differences in the main functionality between the two technologies, both technologies have different performance level during the runtime. Originally the fourth generation programming and developing technology was produced to improve the performance of the third generation programming and developing technology. The third generation suffered from slow performance during runtime and produces lots of implementation errors.

In conclusion, the attempt for extending MATALAB functionalities on the web by using Java software platform, with all of these major differences between the both software platforms, has revealed the following outcomes:

1- There is no possibility for embedding the original MATALB codes used of building the desktop application inside Java software platform. The Java JVM is simply incapable of interpreting the MATLAB code and the web application will crash during runtime.

2- In result of this limitation, all functionalities build by MATALB software platform should be reprogrammed to fit the Java JVM specifications.
3- To fulfill this requirement, the thesis practical work included reprogramming MATALB code and later testing every visual analysis functionality under Java JVM specifications. All of the visual analysis functionalities provided by the MATALB application were running under the Java JVM successfully with similar results and performance.
4- MATALB contains advance developing tool for building the user interfaces. In the Java case, extra tool kits should be added to the Java developing software platform to provide the same user interface functionalities provided by MATALB.
5- During the design and programming phase for building the web user interface, the necessary software tool Kits needed for adding the user interface functionalities to the Java platform was embedding successfully. The implemented web user interface has provided the basics functionalities available in the MATALB application interface.
6- The thesis methodology and implementation proven in a practical procedure the possibility of extending and integrating two different developing platforms with different capabilities and performance for developing web applications.
7- The thesis methodology and implementation have also provide a primary procedure that can be used later on in building other analysis tools for addressing a different natural phenomenon.
8- The provided procedure can also be used for building an open visual analysis tool for addressing any type of data.

Chapter 7

Conclusion and outlook

After addressing the case of performing visual analysis for a lightning storm phenomenon, and also after handling the task of extending this visual analysis techniques performed on the lightning storm phenomenon to the web, a number of conclusions can be presented in the context of this thesis research:

1- For a better visual analysis performance in addressing a special whether condition, the implemented visual analysis tool should be designed and developed from the begging to handle the requirement for addressing this specific whether condition.
2- Implementing a new and fusible technique for extending the existing desktop application on the web. The implementation has provided a procedure for executing every functionality available in the desktop application successfully.
3- Exploiting every open sources technology and technique to extend the existing visual analysis application to the web. The implementation has provided a technique for designing and developing web applications free from any restrictions.
4- Proving the possibility of extending analysis functionalities between multiple platforms as MATLAB and Java, even if the two platforms handle the developing processes and the functionalities execution differently.

As an outlook for further potential steps in enhancing the visual analysis techniques for addressing the lightning storms phenomenon. And also as an outlook for further usage of the web as a medium for performing visual analysis process, the fallowing outlooks can be addressed:

1- Different clustering techniques can be applied on the lightning data according to the aspect the research focuses on.
2- The methods for visualizing lightning data can include any other possible visualization technique, ether by using different colors or by using different geometrical representations. Adding animations or sequent graphs can also be possible approach for visualizing the lightning data.
3- The implemented procedure for extending the desktop visual analysis functionalities to the web can be enchased by providing additional options for the user, for example:

providing the option of loading the user own lightning data or any other similar point data and processing it under the web application.

4- Further researches can be done to combine other platforms with MATLAB developing software platform. Also further researches should be done to foster the interoperability between Java developing software platform and MATALB developing software platform.

5- More focus can be oriented towered the principles of usability and how to apply these principles in visual analysis process.

References

1. Daniel Keim1, G.A., Jean-Daniel Fekete3, Carsten G¨org4,J¨orn Kohlhammer5, and Guy Melan¸con, (2008), ***Visual Analytics Definition, Process, and Challenges***.

2. McWilliams, J.C., ***Fundamentals of Geophysical Fluid Dynamics***.

3. Moulin, H.H.a.B., (2007), ***Using Cognitive Archetypes and Conceptual Graphs to Model Dynamic Phenomena in Spatial Environments***. p.69-82

4. Desclés, J.-P., (1990), ***Langages applicatifs, langues naturelles et cognition***.

5. Desclès, J.-P., (1994), ***Quelques Concepts Relatifs au Temps et à l'Aspect pour l'Analyse des Textes***. p.57–88

6. Shipley, T.F., S.I. Fabrikant, and A.-K. Lautenschütz, (2013), ***Creating Perceptually Salient Animated Displays of Spatiotemporal Coordination in Events***. p.259-270

7. Michael F. Goodchild, A.G., ***REPRESENTATION AND COMPUTATION OF GEOGRAPHIC DYNAMICS.***

8. Burrough, P.A. and R.A. McDonnell, (1998), ***Principles of Geographical Information Systems***.

9. GOODCHILD, M.F., (1992), ***GEOGRAPHICAL DATA MODELING***. p.401-408

10. Barry Smith, A.C.V., (1997), ***Fiat and Bona Fide Boundaries: Towards an Ontology of Spatially Extended Objects***. p.103-119

11. Brentano,(1976), ***Philosophische Untersuchungen zu Raum, Zeit und Kontinuum***.

12. McDerby, M.T.a.M., ***Experiences of Using State of the Art Immersive Technologies for Geographic Visualization.***

13. Oyster, C.W., (1999), ***The Human Eye – Structure and Function***.

14. E., W.J.G.S.K.M.M., (1995), ***Viualization***. Annual review of information science and technology.

15. Telea, A.C., (2008), ***Data visualization,principles and practice***. 2008.

16. Gennady ANDRIENKO, N.A., Peter GATALSKY, ***Mapping spatio-temporal data for exploratory analysis***.

17. DONNA J. PEUQUET, N.D., (1995), ***An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data***. International journal of geographical information systems, p. 7-24.

18. Peters, S. and L. Meng,(2013), ***Visual Analysis for Nowcasting of Multidimensional Lightning Data.*** ISPRS International Journal of Geo-Information, p. 817-836.

19. Gennady Andrienko, N.A., Urška Demšar, Doris Dransch,Jason Dykes, Sara Fabrikant, Mikael Jern, Menno-Jan Kraak, HeidrunSchumann, Christian Tominski, (2010), ***Space, Time, and Visual Analytics.***

20. Andrienko, N., G. Andrienko, and P. Gatalsky, (2003), ***Exploratory spatio-temporal visualization: an analytical review.*** Journal of Visual Languages & Computing, p. 503-541.

21. A. Buja, D.C., D.F. Swayne, (1996), ***Interactive high-dimensional data visualization.*** Computational and Graphical Statistics, p. 78–99.

22. A. Fredrikson, C.N., C. Plaisant, B. Shneiderman, (1999), ***geographic and categorical aggregations viewed through coordinated displays.*** ACM Press, New York, p. 26–34.

23. MacEachren, A.M., (1995), ***How Maps Work: Representation, Visualization, and Design.*** The Guilford Press.

24. Andrienko, G.A.a.N., ***Dynamic and interactive displays.***

25. Menno-Jan Kraak, A.B., (2001), ***Settings and needs for web cartography.*** Web Cartography, p. 1-7.

26. Worm, J.v.d., (2001), ***Web map design in practice.*** *Web Cartography,* p:91-111.

27. Tom Auer , A.M.M., Craig McCabe , Scott Pezanowski, Michael Stryker, *HerbariaViz:* (2011), ***A web-based client–server interface for mapping and exploring flora observation data.*** Ecological Informatics, p. 93-110.

28. G. Descamps, P.T., R. Therrien,(2006) ***An interactive and open approach for the analysis and diffusion of geoscientific data.*** Computers & Geosciences, p. 643-655.

29. Kahkonen, J., et al., (1999), ***Interactive visualisation of geographical objects on the Internet.*** International Journal of Geographical Information Science, p. 429-438.

30. Kulawiak, M., et al., (2010), ***Interactive visualization of marine pollution monitoring and forecasting data via a Web-based GIS.*** Computers & Geosciences, p. 1069-1080.

31. Stefan Peters, H.-D.B., Liqiu Meng, (2013), ***Visual Analysis of lightning data using Space-Time-Cube.***

32. Berson, A., (1996), ***Client/Server Architecture.*** Second edition.

33. Goodwill, J., (2000), ***Pure JSP - Java Server Pages: A Code-Intensive Premium Reference***. p. 320.

34. Flanagan, D., (2011), ***JavaScript: The Definitive Guide***. Sixth edition.

35. Russell, M.A., (2008), ***Dojo: The Definitive Guide***. First Edition.

36. Jason Hunter, W.C., (April 2001), ***Java Servlet Programming***. 2nd Edition.

37. Allen, R., ***Web development with JavaScript and Ajax illuminated***.

Appendix

A. MATLAB codes for plotting lighting data in different representations forms

A.1 Code for plotting lightning points in 3D (point geometry)

```
function  plottingAllPoint3D =AllPoint3D(arg)
f = figure;
for k=1:7101
i=arg(k,5);
if i ==1
plot(k)= plot3(arg(k,1),arg(k,2),arg(k,3), '*','markersize',2,'Color',[0.85
0.50 0.15],'visible','on');
hold on;
view(3);
end
if i  == 2
plot(k)= plot3(arg(k,1),arg(k,2),arg(k,3), '*','markersize',2,'Color',[0.50
0.85 0.15],'visible','on');
hold on;
view(3);
end
grid on
end
plottingAllPoint3D=webfigure(f);
close(f);
end
```

A.2 Code for plotting lightning clusters centers (point geometry)

```
function  plottingCellsCenter =cellsCenter(arg)
f = figure;
for k=1:71
i=arg(k,4);
if i ==1
d(k)= plot3(arg(k,1),arg(k,2),arg(k,3), ‘x’,’markersize’,12,’Color’,[0.85
0.50 0.15],’linewidth’,2);
hold on;
end
if i  == 2
d(k)= plot3(arg(k,1),arg(k,2),arg(k,3), ‘x’,’markersize’,12,’Color’,[ 0.50
0.85 0.15],’linewidth’,2);
hold on;
view(3);
end
end
plottingCellsCenter=webfigure(f);
close(f);
end
```

A.3 Code for plotting lightning clusters tracks centers (point geometry)

```
function  plottingTrackCentres =TrackCentres(arg)
f = figure;
for k=1:71
i=arg(k,4);
if i ==1
out_TrackCentre2(k) = plot3(arg(k,1),arg(k,2),arg(k,3),'.','color',[0.85
0.50 0.15],'markersize',15);
hold on
end
if i ==2
out_TrackCentre2(k) = plot3(arg(k,1),arg(k,2),arg(k,3),'.','color',[ 0.50
0.85 0.15],'markersize',15);
end
end
plottingTrackCentres=webfigure(f);
close(f);
end
```

A.4 Code for plotting lightning clusters tracks (line geometry)

```
function  TrackLine3D =tracklines3D(arg)
f = figure;
for k=1:71
i=arg(k,8);
if i ==1
d(k)=line(arg(k,1:2),arg(k,3:4),arg(k,5:6) ,'Color',[0.85 0.50 0.15]);
end
if i ==2
d(k)=line(arg(k,1:2),arg(k,3:4),arg(k,5:6) ,'Color',[ 0.50 0.85 0.15]);
end
end
view(3);
TrackLine3D=webfigure(f);
close(f);
end
```

A.5 Code for plotting lightning clusters as spherical objects (volume geometry)

```
function  plotting cellsCentreSize =TrackCentreSize(arg)
f = figure;
for k=1:71
i=arg(k,4);
h=arg(k,5);
if i ==1
out_TrackCentre2(k) = plot3(arg(k,1),arg(k,2),arg(k,3),'.','color',[0.85
0.50 0.15],
'markersize',h);
hold on,grid on
end
```

```
if i ==2
out_TrackCentre2(k)  =  plot3(arg(k,1),arg(k,2),arg(k,3),'.','color',[  0.50
0.85 0.15],
'markersize',h);
hold on, grid on
end
end
plotting cellsCentreSize=webfigure(f);
close(f);
end
```

The spherical objects are varying in their sizes to depict the changes in the lightning storm volume

A.6 Code for plotting lightning clusters as convex hull in 2D (Polygon geometry)

```
function  plottingconvexhull2D =convexhull2D(arg)
f = figure;
eee1=arg(arg(:,4)==1,:);
for k=1:24
ppp1=eee1(eee1(:,3)==k,:);
ddd1=convhull(ppp1(:,1:2));
b(k)=fill3(ppp1(ddd1,1),ppp1(ddd1,2),zeros(1,length(ddd1 )),
[0.85, 0.50, 0.15],'facealpha',.1);
hold on;
end
eee2=arg(arg(:,4)==2,:);
for k=25:71
ppp2=eee2(eee2(:,3)==k,:);
ddd2=convhull(ppp2(:,1:2));
b(k)=fill3(ppp2(ddd2,1),ppp2(ddd2,2),zeros(1,length(ddd2)),
[0.50, 0.85, 0.15],'facealpha',.1);
hold on;
end
plottingconvexhull2D =webfigure(f);
close(f)
end
```

In the previous code, `convhull` is a function provided by Matlab for building convex hull objects out of a set of points in 2D or 3D space. `fill3` is also a Matlab function that can be used to create flat-shaded polygons.

These two default MATALB functions are used to visualize the lightning data as 3D convex hull with distinguish surfaces.

A.7 Code for plotting lightning clusters as convex hull in 3D (volume geometry)

```
function  plottingconvexhull3D =Conv3D(arg)
f = figure;
eee1=arg(arg(:,5)==1,:);
for k=1:24
    ppp1=eee1(eee1(:,4)==k,:);
    ppp2=ppp1(:,1:3);
    ppp3=arg(arg(:,9)==k,:);
    ppp4=ppp3(:,6:8);
    b(k)=trisurf(ppp4,ppp2(:,1),ppp2(:,2),ppp2(:,3),'facecolor',
[0.50, 0.85, 0.15],'facealpha',.1,'edgecolor','none');
    hold on;
end
eee2=arg(arg(:,5)==2,:);
for k=25:71
    ppp1=eee2(eee2(:,4)==k,:);
    ppp2=ppp1(:,1:3);
    ppp3=arg(arg(:,9)==k,:);
    ppp4=ppp3(:,6:8);
    b(k)=trisurf(ppp4, ppp2(:,1),ppp2(:,2),ppp2(:,3),'facecolor',
[0.85, 0.50, 0.15],'facealpha',.1,'edgecolor','none');
    hold on;
end
plottingconvexhull3D =webfigure(f);
close(f)
end
```

In the previous code, the MATLAB function (`trisurf`)is used to create a triangular network from the lightning point data, later the combined triangles will form a surface for representing the volume the lighting strike can occupy from space.

A.8 Code for plotting lightning clusters as ellipsoids

```
function  plottingEllipsoide=ellipsoide(arg)
f = figure;
eee1=arg(arg(:,19)==1,:);
for k=1:24;
ttt1=eee1(eee1(:,18)==k,:);
x1=ttt1(ttt1(:,17)==1,:);
x=x1(:,1:16);
y1=ttt1(ttt1(:,17)==2,:);
y=y1(:,1:16);
z1=ttt1(ttt1(:,17)==3,:);
z=z1(:,1:16);
    b(k)=surf(x,y,z,'FaceColor',[0.85,0.50,0.15],'LineWidth',0.1,
    'EdgeColor',[0.85, 0.50, 0.15],'FaceAlpha',.6,'EdgeAlpha',.6);
    hold on;
end
eee2=arg(arg(:,19)==2,:);
for k=25:71;
ttt2=eee2(eee2(:,18)==k,:);
x1=ttt2(ttt2(:,17)==1,:);
x=x1(:,1:16);
```

```
y1=ttt2(ttt2(:,17)==2,:);
y=y1(:,1:16);
z1=ttt2(ttt2(:,17)==3,:);
z=z1(:,1:16);
    b(k)=surf(x,y,z,'FaceColor',[ 0.50 0.85 0.15],
'LineWidth',0.1,'EdgeColor',[ 0.50 0.85 0.15],
'FaceAlpha',.6,'EdgeAlpha',.6);
    hold on;
end
view(3);
plottingEllipsoide=webfigure(f);
close(f);
end
```

Notes: the MATLAB function (`surf`) creates a surface using X and Y (as vector components) and Z value for representing the height.

A. HTML Code for web page (representing user interface)

The HTML code also includes the codes of JSP, JavaScript and Dojo, which is used for organizing and manipulating the functionality and behavior of the HTML file.

B.1 slider functionality JSP file

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1" %>
    <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
    <html>

    <head>
        <link rel="stylesheet" href="dojoToolKit/dijit/themes/claro/claro.css">
        <script>
            dojoConfig = {
                async: true,
                parseOnLoad: false
            };
        </script>
        <script src="dojoToolKit/dojo/dojo.js"></script>
        <script>
            var widget;
            require(["dojo/dom", "dojo/parser"
                    "dijit/form/HorizontalSlider", "dijit/registry", "dojo/domReady!"],

                function go(dom, parser, HorizontalSlider, TextBox, number) {
                    parser.parse();
                    slider = new dijit.form.HorizontalSlider({
                        name: "slider",
                        value: 35,
                        minimum: 1,
                        maximum: 71,
                        intermediateChanges: false,
                        showButtons: true,
                        discreteValues: 71,
                        style: "width:300px;",

                        onChange: function doit(value) {
                            dom.byId("sliderValue")
                                .value = value;
                            widget = dijit.byId("sliderValue");
                            document.getElementById("mytext")
                                .value = widget.get("value");
                            document.getElementById("object")
                                .submit("mytext");
                        },

                    }, "slider");
                });

             <%
                String sizeStr = request.getParameter("mytext");
            int mytext = 35;
```

```
            boolean sizeSet = false;
            if (sizeStr != null && sizeStr.length() > 0) {
                mytext = Integer.parseInt(sizeStr);
                sizeSet = true;
            } %>
        </script>
    </head>

    <body class="claro">
        <div id="slider"></div>
        <form id=object method="get">
            <p>
                <input id="sliderValue" name="mytext" type="text" data-dojo-
type="dijit/form/TextBox" value="<%=mytext%>" />
            </p>
            <p>
                <input type="hidden" name="mytext" id="mytext" />
            </p>
            <script type="text/javascript">
                try {
                    var my = new XMLHttpRequest();
                } catch (e) {
                    try {
                        var my = new ActiveXObject('Msxml2.XMLHTTP');
                    } catch (e) {
                        try {
                            var my = new ActiveXObject('Microsoft.XMLHTTP');
                        } catch (e) {
                            document.write('XMLHttpRequest not supported');
                        }
                    }
                }
                my.open('GET', 'sliding123?sliderValue=<%=mytext%>', false);
                my.send(null);
                document.writeln(my.responseText);
            </script>
        </form>
    </body>

    </html>
```

B.2 multiple representation functionality JSP file

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
    <html>

    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
        <title>First Demo</title>
        <link rel="Stylesheet" type="text/css" media=all href="./StyleSheet.css" />
        <link href="StyleSheet.css" rel="stylesheet" type="text/css" />
    </head>

    <body>
        <form method="get" action="Gui1">
            <h1> Visualizing Services</h1>
            <h3> Please select the desired view</h3>
            <div style="text-align: left">
                <input type="radio" name="stefan" value="AP3D">All Points3D
                <br>
                <input type="radio" name="stefan" value="CC2D">Cells Centers
                <br>
                <input type="radio" name="stefan" value="3DTC">Tracks Centers
                <br>
                <input type="radio" name="stefan" value="3DTCS">Track Centers Sizes
                <br>
                <input type="radio" name="stefan" value="3DTL">Tracks lines
                <br>
                <input type="radio" name="stefan" value="2DCH">convex Hall2D
                <br>
                <input type="radio" name="stefan" value="3DCH">convex Hall3D
                <br>
                <input type="radio" name="stefan" value="ELL">Ellipsoid
                <br>
                <input type="submit" value="submit" />
            </div>
        </form>
    </body>

    </html>
```

b.3 combined representation functionality JSP file

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
    <html>

    <head>
        <link rel="Stylesheet" type="text/css" media=all href="./StyleSheet.css" />
        <link href="StyleSheet.css" rel="stylesheet" type="text/css" />
    </head>

    <body>
        <form method="get" action="Space123">
            <h1> Visualizing Services</h1>
            <h3> Please select the desired view</h3>
            <div style="text-align: left">
                <p>
                    <input type="checkbox" name="box" value="2DCH" />convex Hall2D</p>
                <br>
                <p>
                    <input type="checkbox" name="box" value="3DTL" />Tracks lines</p>
                <br>
                <p>
                    <input type="checkbox" name="box" value="3DTC" />Track Centers Sizes</p>
                <br>
                <input type="submit" value="Submit">
            </div>
        </form>
    </body>

    </html>
```

B. Java code for building the servlet file

C.1 Servelt file for running slider functionality

```
package timeSlider.moh.test1;
import java.io.IOException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import passingTimeSliderNumber.*;
import com.mathworks.toolbox.javabuilder.MWException;
import com.mathworks.toolbox.javabuilder.MWJavaObjectRef;
import com.mathworks.toolbox.javabuilder.MWNumericArray;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigure;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigureHtmlGenerator;
import com.sun.xml.internal.ws.api.config.management.policy.ManagementAssertion.Setting;
public class sliding extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        String test = request.getParameter("sliderValue");
        WebFigure figure1 = null;
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException e) {
            throw new ServletException(e);
        }
        double[][] coor = new double[71][5];
        try {
            java.sql.Connection connection = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
            java.sql.Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("Select X, Y, Z, Class, Size
From trackcentresize");
            int i = 0;
            while (resultSet.next()) {
                coor[i][0] = resultSet.getDouble("X");
                coor[i][1] = resultSet.getDouble("Y");
                coor[i][2] = resultSet.getDouble("Z");
                coor[i][3] = resultSet.getDouble("Class");
                coor[i][4] = resultSet.getDouble("Size");
                i++;
            }
            connection.close();
        } catch (SQLException e) {
            throw new ServletException(e);
        }
        try {
            Plotting calc = new Plotting();
            MWNumericArray size = new
MWNumericArray(Integer.parseInt(request.getParameter("sliderValue")));
                 Object[] results = calc.TrackCentreSizeBackup(1, coor, size);
```

```
            MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
            figure1 = (WebFigure) ref.get();
            getServletContext()
                .setAttribute("UserPlot1", figure1);
        } catch (MWException e3) {
            e3.printStackTrace();
        }
        StringBuffer buffer = new StringBuffer();
        WebFigureHtmlGenerator webFigureHtmlforStefan = new
WebFigureHtmlGenerator("WebFigures", getServletContext());
        try {
            String outputString =
webFigureHtmlforStefan.getFigureEmbedString(figure1, "UserPlot1", "application",
"600", "600", null);
            buffer.append(outputString);
        } catch (Exception e) {
            e.printStackTrace();
        }
        response.getOutputStream()
            .print(buffer.toString());
    }
}
```

C.2 Servelt file for running multiple representation functionality

```
package Moh.All.Codes;
import java.io.IOException;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.ServletConfig;
import com.mathworks.toolbox.javabuilder.MWException;
import com.mathworks.toolbox.javabuilder.MWJavaObjectRef;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigure;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigureHtmlGenerator;
import test1.*;
public class Gui1 extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        WebFigure figure1 = null;
        WebFigure figure2 = null;
        String f = request.getParameter("stefan");
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException e1) {
            throw new ServletException(e1);
        }
        if (f.endsWith("2DPC")); {
            double[][] coor = new double[7150][6];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://localhost:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select xcoor, ycoor, color1,
color2, color3, clusters From allpoints");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("xcoor");
                    coor[i][1] = resultSet.getDouble("ycoor");
                    coor[i][2] = resultSet.getDouble("color1");
                    coor[i][3] = resultSet.getDouble("color2");
                    coor[i][4] = resultSet.getDouble("color3");
                    coor[i][5] = resultSet.getDouble("clusters");
                    i++;
                }
                connection.close();
            } catch (SQLException e1) {
                throw new ServletException(e1);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.AllPoints(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
```

```
            } catch (MWException e2) {
                e2.printStackTrace();
            }
        }
        if (f.endsWith("2DAP")) {
            double[][] coor = new double[7101][5];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, cellID, class
From AllPoints2D3D");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
                    coor[i][3] = resultSet.getDouble("cellID");
                    coor[i][4] = resultSet.getDouble("class");
                    i++;
                }
                connection.close();
            } catch (SQLException e3) {
                throw new ServletException(e3);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.Allpoints2D(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e2) {
                e2.printStackTrace();
            }
        }
        if (f.endsWith("2DCH")) {
            double[][] coor = new double[7099][4];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, cellID, class From
convexhull2D");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("cellID");
                    coor[i][3] = resultSet.getDouble("class");
                    i++;
                }
                connection.close();
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
```

```
                Object[] results = calc.convexhull2D(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        if (f.endsWith("AP3D")) {
            double[][] coor = new double[7101][5];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, cellID, class
From AllPoints2D3D");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
                    coor[i][3] = resultSet.getDouble("cellID");
                    coor[i][4] = resultSet.getDouble("class");
                    i++;
                }
                connection.close();
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.AllPoints3D(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        if (f.endsWith("CC2D")) {
            double[][] coor = new double[71][4];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, class   From
CellsCenter");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
                    coor[i][3] = resultSet.getDouble("class");
                    i++;
                }
                connection.close();
```

```
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.cellsCenter(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        if (f.endsWith("3DTC")) {
            double[][] coor = new double[71][4];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, Class   From
trackcentre");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
                    coor[i][3] = resultSet.getDouble("Class");
                    i++;
                }
                connection.close();
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.TrackCentres(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        if (f.endsWith("3DTCS")) {
            double[][] coor = new double[71][5];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, Class, Size
From trackcentresize");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
```

```
                coor[i][3] = resultSet.getDouble("Class");
                coor[i][4] = resultSet.getDouble("Size");
                i++;
            }
            connection.close();
        } catch (SQLException e2) {
            throw new ServletException(e2);
        }
        try {
            Plotting calc = new Plotting();
            Object[] results = calc.TrackCentreSize(1, coor);
            MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
            figure1 = (WebFigure) ref.get();
            getServletContext()
                .setAttribute("UserPlot1", figure1);
        } catch (MWException e3) {
            e3.printStackTrace();
        }
    }
    if (f.endsWith("3DTL")) {
        double[][] coor = new double[71][8];
        try {
            java.sql.Connection connection = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
            java.sql.Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("Select X1,X2, Y1,Y2, Z1,Z2,
cellID, class   From tracklines");
            int i = 0;
            while (resultSet.next()) {
                coor[i][0] = resultSet.getDouble("X1");
                coor[i][1] = resultSet.getDouble("X2");
                coor[i][2] = resultSet.getDouble("Y1");
                coor[i][3] = resultSet.getDouble("Y2");
                coor[i][4] = resultSet.getDouble("Z1");
                coor[i][5] = resultSet.getDouble("Z2");
                coor[i][6] = resultSet.getDouble("cellID");
                coor[i][7] = resultSet.getDouble("class");
                i++;
            }
            connection.close();
        } catch (SQLException e2) {
            throw new ServletException(e2);
        }
        try {
            Plotting calc = new Plotting();
            Object[] results = calc.tracklines3D(1, coor);
            MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
            figure2 = (WebFigure) ref.get();
            getServletContext()
                .setAttribute("UserPlot1", figure2);
        } catch (MWException e3) {
            e3.printStackTrace();
        }
    }
    if (f.endsWith("3DCH")) {
        double[][] coor = new double[7100][9];
        try {
            java.sql.Connection connection = DriverManager.getConnection(
```

```
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select X, Y, Z, cellID,class,
ch1, ch2, ch3, chcellID  From Full3Dconvexhull");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("X");
                    coor[i][1] = resultSet.getDouble("Y");
                    coor[i][2] = resultSet.getDouble("Z");
                    coor[i][3] = resultSet.getDouble("cellID");
                    coor[i][4] = resultSet.getDouble("class");
                    coor[i][5] = resultSet.getDouble("ch1");
                    coor[i][6] = resultSet.getDouble("ch2");
                    coor[i][7] = resultSet.getDouble("ch3");
                    coor[i][8] = resultSet.getDouble("chcellID");
                    i++;
                }
                connection.close();
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.Conv3D(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        if (f.endsWith("ELL")) {
            double[][] coor = new double[3408][19];
            try {
                java.sql.Connection connection = DriverManager.getConnection(
                    "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
                java.sql.Statement statement = connection.createStatement();
                ResultSet resultSet = statement.executeQuery("Select
a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q,r,s From ellipsoide");
                int i = 0;
                while (resultSet.next()) {
                    coor[i][0] = resultSet.getDouble("a");
                    coor[i][1] = resultSet.getDouble("b");
                    coor[i][2] = resultSet.getDouble("c");
                    coor[i][3] = resultSet.getDouble("d");
                    coor[i][4] = resultSet.getDouble("e");
                    coor[i][5] = resultSet.getDouble("f");
                    coor[i][6] = resultSet.getDouble("g");
                    coor[i][7] = resultSet.getDouble("h");
                    coor[i][8] = resultSet.getDouble("i");
                    coor[i][9] = resultSet.getDouble("j");
                    coor[i][10] = resultSet.getDouble("k");
                    coor[i][11] = resultSet.getDouble("l");
                    coor[i][12] = resultSet.getDouble("m");
                    coor[i][13] = resultSet.getDouble("n");
                    coor[i][14] = resultSet.getDouble("o");
                    coor[i][15] = resultSet.getDouble("p");
```

```
                    coor[i][16] = resultSet.getDouble("q");
                    coor[i][17] = resultSet.getDouble("r");
                    coor[i][18] = resultSet.getDouble("s");
                    i++;
                }
                connection.close();
            } catch (SQLException e2) {
                throw new ServletException(e2);
            }
            try {
                Plotting calc = new Plotting();
                Object[] results = calc.ellipsoide(1, coor);
                MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
                figure1 = (WebFigure) ref.get();
                getServletContext()
                    .setAttribute("UserPlot1", figure1);
            } catch (MWException e3) {
                e3.printStackTrace();
            }
        }
        StringBuffer buffer = new StringBuffer();
        WebFigureHtmlGenerator webFigureHtmlforStefan = new
WebFigureHtmlGenerator("WebFigures", getServletContext());
        try {
            String outputString = webFigureHtmlforStefan.getFigureEmbedString(figure1,
"UserPlot1", "application", "600", "600", null);
            buffer.append(outputString);
        } catch (Exception e) {
            e.printStackTrace();
        }
        response.getOutputStream()
            .print(buffer.toString());
    }
}
```

C.3 Servelt file for running combined representation functionality

```
package Space.SideView.Moh;
import java.io.IOException;
import java.io.PrintWriter;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import javax.servlet.ServletConfig;
import javax.servlet.ServletException;
import javax.servlet.ServletResponse;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import com.mathworks.toolbox.javabuilder.MWException;
import com.mathworks.toolbox.javabuilder.MWJavaObjectRef;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigure;
import com.mathworks.toolbox.javabuilder.webfigures.WebFigureHtmlGenerator;
import SpaceTimeCube.*;
public class SpaceTimeCube extends HttpServlet {
    private static final long serialVersionUID = 1L;
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        try {
            Class.forName("com.mysql.jdbc.Driver");
        } catch (ClassNotFoundException e1) {
            throw new ServletException(e1);
        }
        double[][] coor = new double[29386][22];
        try {
            java.sql.Connection connection = DriverManager.getConnection(
                "jdbc:mysql://127.0.0.1:3306/plotting", "root", "123456");
            java.sql.Statement statement = connection.createStatement();
            ResultSet resultSet = statement.executeQuery("Select A1, A2, A3, A4, A5, A6, A7,
A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18, A19, A20, A21, A22  from
SpaceTimeCube");
            int i = 0;
            while (resultSet.next()) {
                coor[i][0] = resultSet.getDouble("A1");
                coor[i][1] = resultSet.getDouble("A2");
                coor[i][2] = resultSet.getDouble("A3");
                coor[i][3] = resultSet.getDouble("A4");
                coor[i][4] = resultSet.getDouble("A5");
                coor[i][5] = resultSet.getDouble("A6");
                coor[i][6] = resultSet.getDouble("A7");
                coor[i][7] = resultSet.getDouble("A8");
                coor[i][8] = resultSet.getDouble("A9");
                coor[i][9] = resultSet.getDouble("A10");
                coor[i][10] = resultSet.getDouble("A11");
                coor[i][11] = resultSet.getDouble("A12");
                coor[i][12] = resultSet.getDouble("A13");
                coor[i][13] = resultSet.getDouble("A14");
                coor[i][14] = resultSet.getDouble("A15");
                coor[i][15] = resultSet.getDouble("A16");
                coor[i][16] = resultSet.getDouble("A17");
                coor[i][17] = resultSet.getDouble("A18");
                coor[i][18] = resultSet.getDouble("A19");
                coor[i][19] = resultSet.getDouble("A20");
```

```
                coor[i][20] = resultSet.getDouble("A21");
                coor[i][21] = resultSet.getDouble("A22");
                i++;
            }
            connection.close();
        } catch (SQLException e2) {
            throw new ServletException(e2);
        }
        WebFigure figure1 = null;
        String[] values = request.getParameterValues("box");
        int h = 0;
        int n = 0;
        int o = 0;
        for (int i = 0; i < values.length; i++) {
            if (values[i].matches("2DCH")) {
                h = 1;
            }
            if (values[i].matches("3DTL")) {
                n = 1;
            }
            if (values[i].matches("3DTC")) {
                o = 1;
            }
        }
        try {
            Plotting calc = new Plotting();
            Object[] results = calc.SpaceTimeCube(1, coor, h, n, o);
            MWJavaObjectRef ref = (MWJavaObjectRef) results[0];
            figure1 = (WebFigure) ref.get();
            getServletContext()
                .setAttribute("UserPlot1", figure1);
        } catch (MWException e3) {
            e3.printStackTrace();
        }
        StringBuffer buffer = new StringBuffer();
        WebFigureHtmlGenerator webFigureHtmlforStefan = new
WebFigureHtmlGenerator("WebFigures", getServletContext());
        try {
            String outputString = webFigureHtmlforStefan.getFigureEmbedString(figure1,
"UserPlot1", "application", "600", "600", null);
            buffer.append(outputString);
        } catch (Exception e) {
            e.printStackTrace();
        }
        response.getOutputStream()
            .print(buffer.toString());
    }
}
```

C. Web application tools and functionalities interface

D.1 Radio buttons

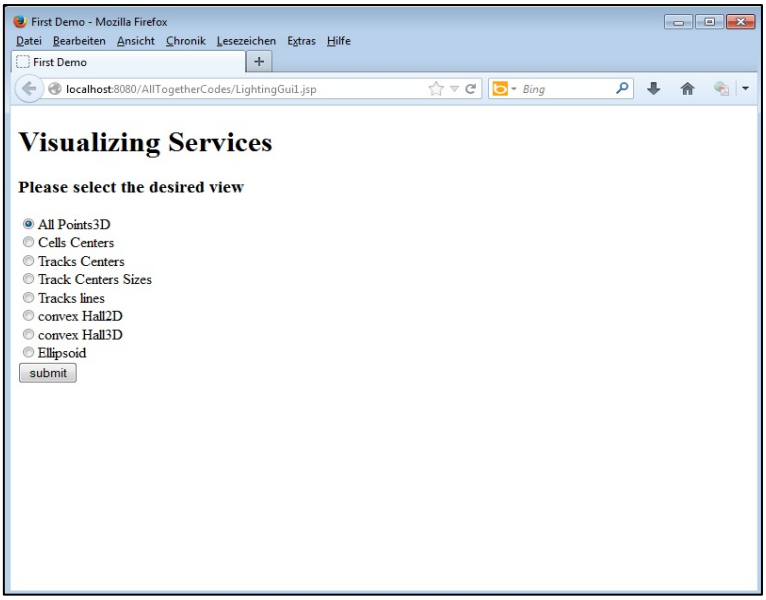


Figure 18: Radio buttons for selecting different visualizing options

D.2 Time slider

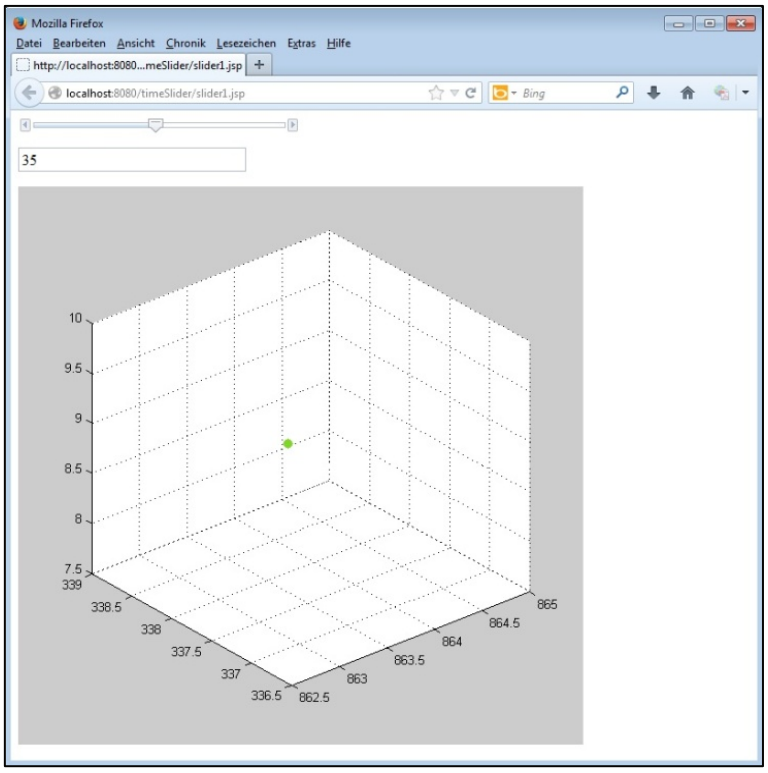


Figure 19: time slider Implantation

D.3 Combined view

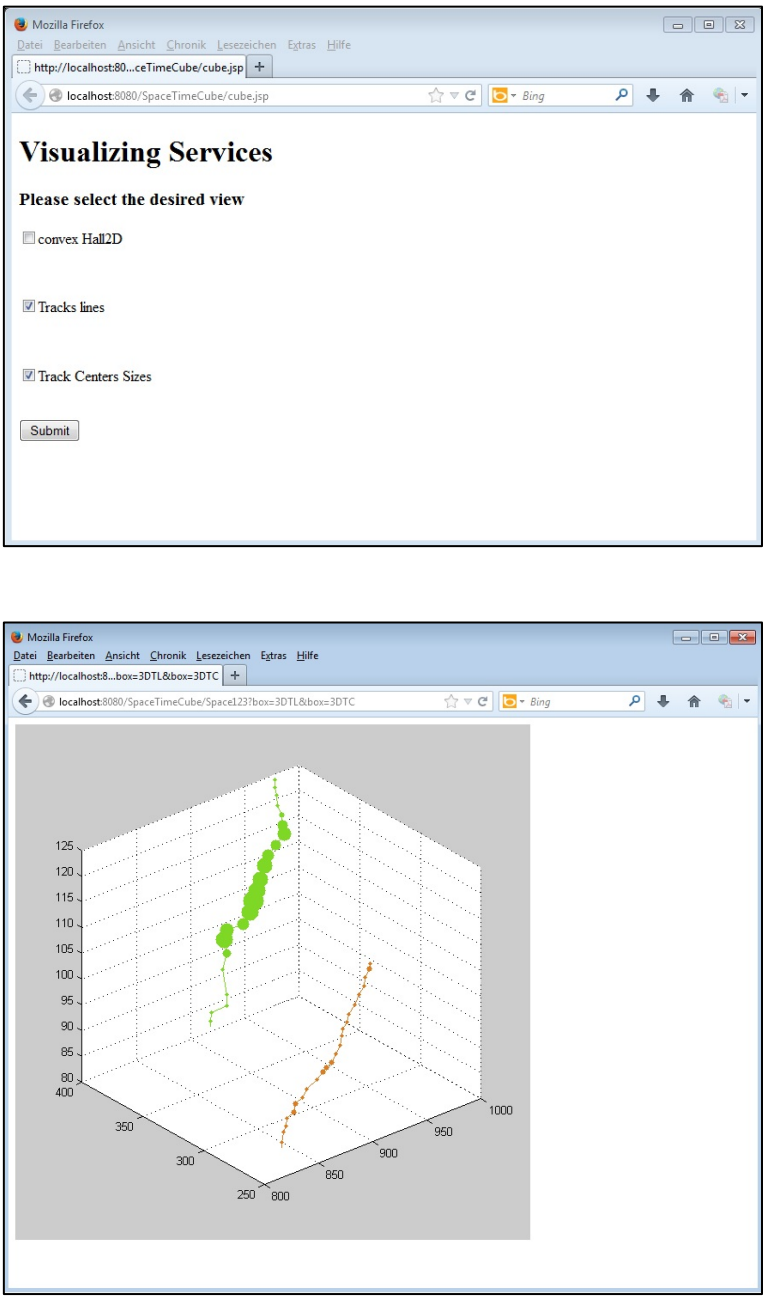


Figure 20 : combined visualizing methods

The previous figure illustrates the combined view functionality. When using the combined view tool, the user can visualize the patterns of the lightning storms, combined with the changes in the lightning storm volume through time. The 3D lines presents the pattern of lightning storms and the spherical objects presents the variation of lightning cloud size during the cloud traveling time.

D.4 Visualization options

D.4.1 visualizing lightning point data in 3D

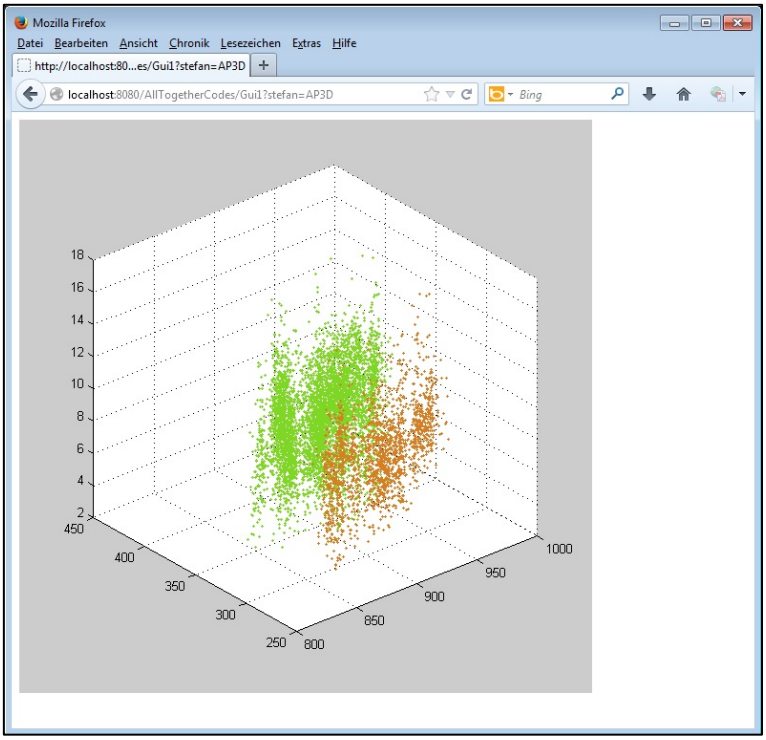


Figure 21: lightning points distribution

D.4.2 visualizing lightning clusters centers

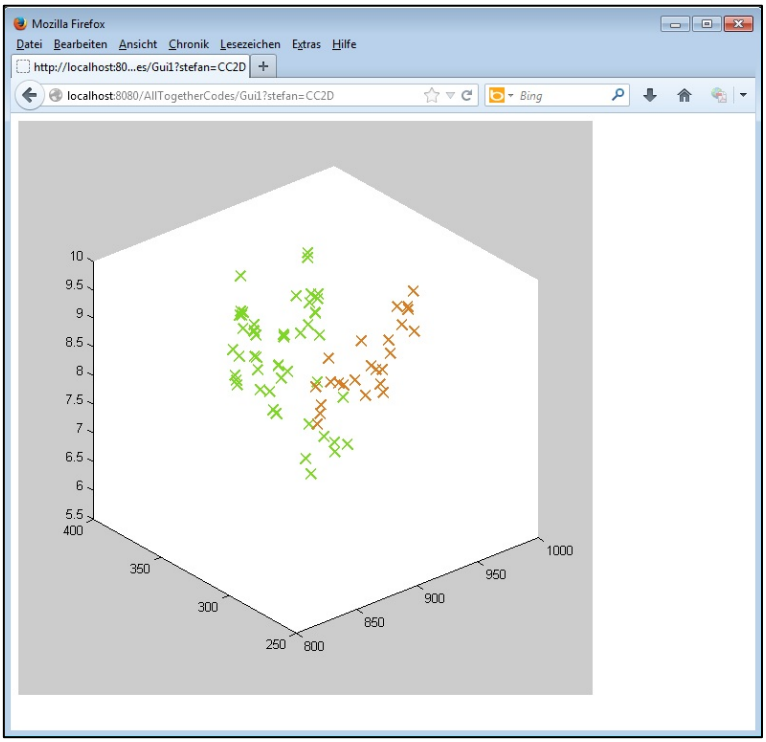


Figure 22: cross representation

D.4.3 visualizing lightning clusters tracks centers

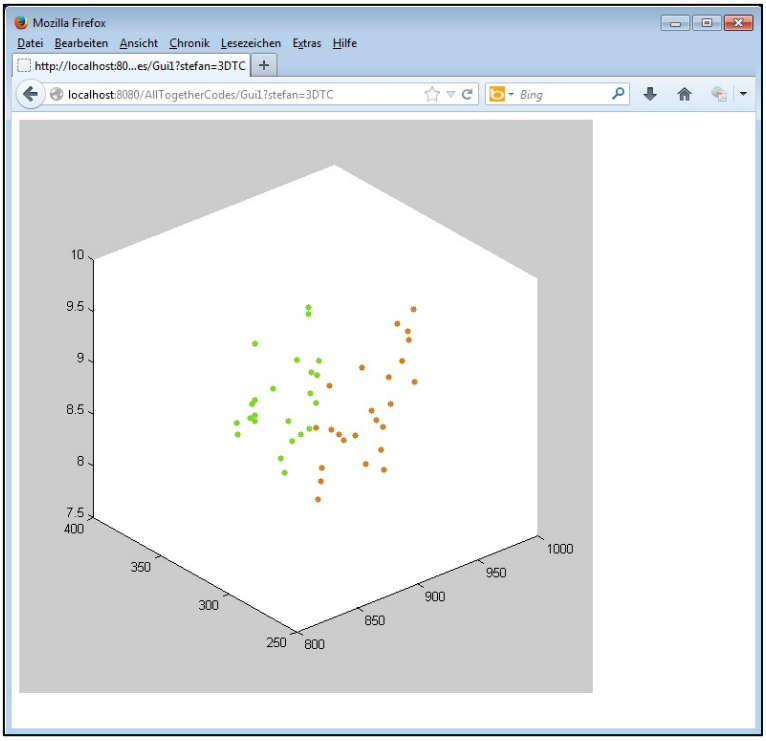


Figure 23: point representation

D.4.4 visualizing lightning clusters tracks

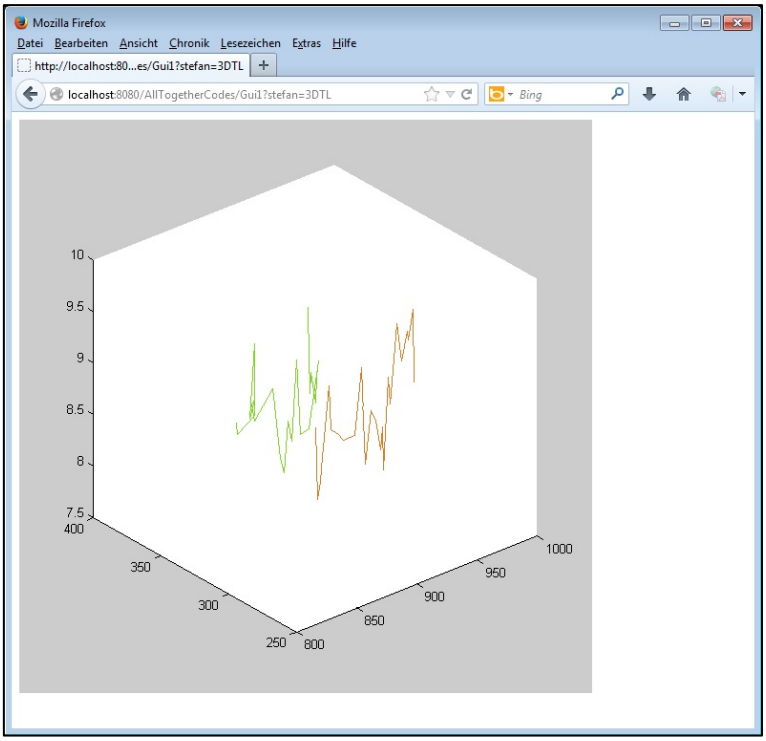


Figure 24: line representation

D.4.5 visualizing lightning clusters as spherical objects

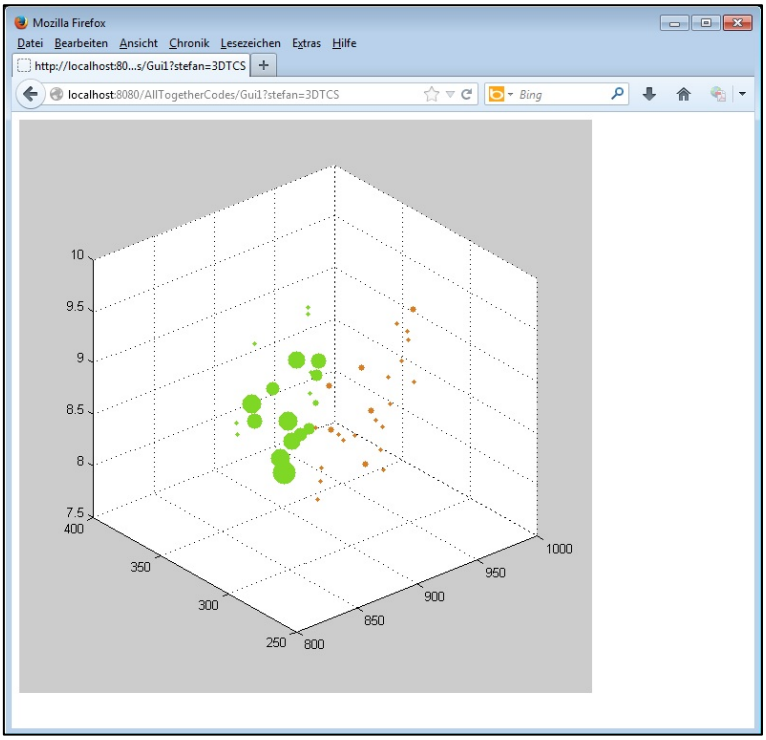


Figure 25: spherical objects representation

D.4.6 visualizing lightning clusters as 2D convex hulls

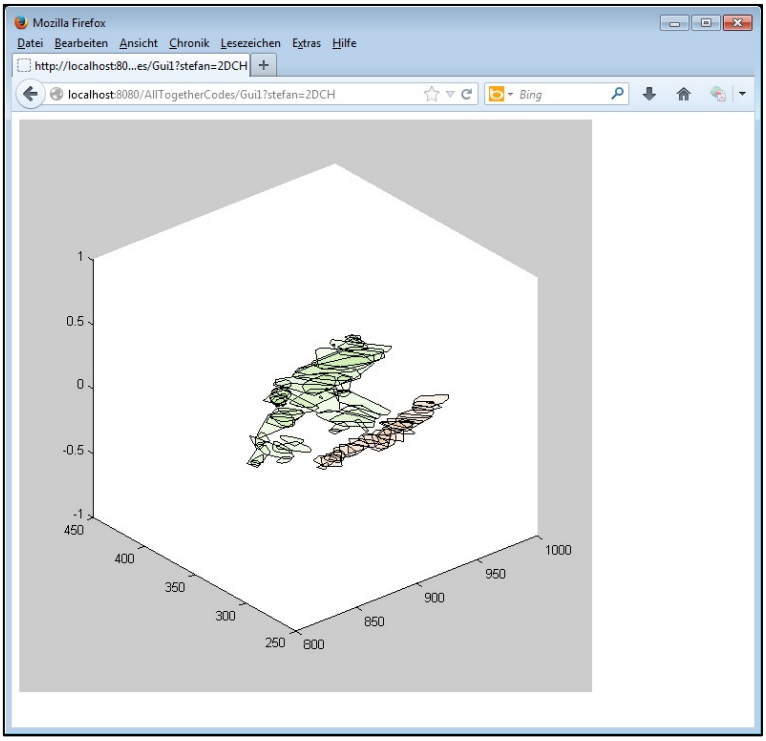


Figure 26: 2D convex hull representation

D.4.7 visualizing lightning clusters tracks as convex hull in 3D

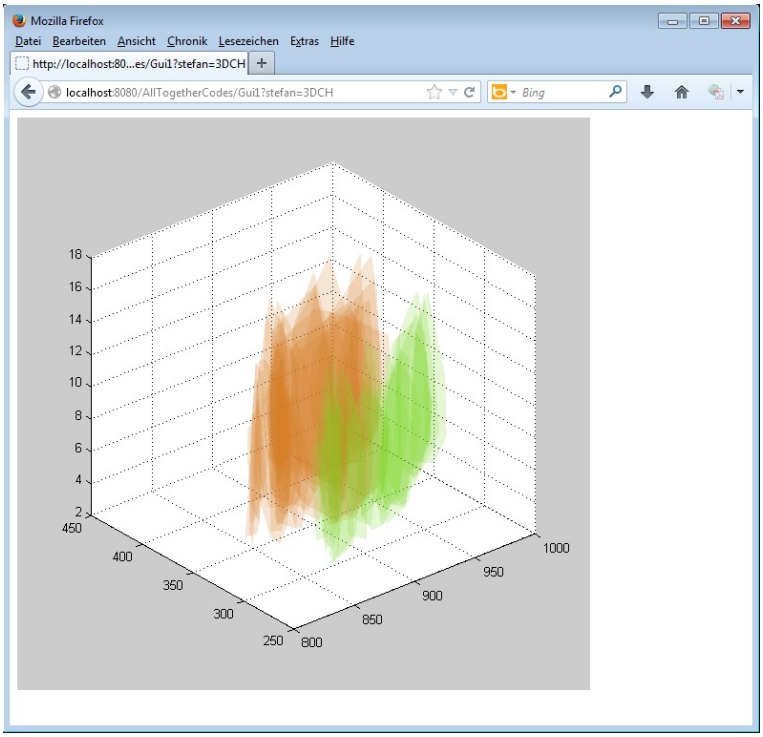


Figure 27: 3D convex hull representation

D.4.8 visualizing lightning clusters as ellipsoids

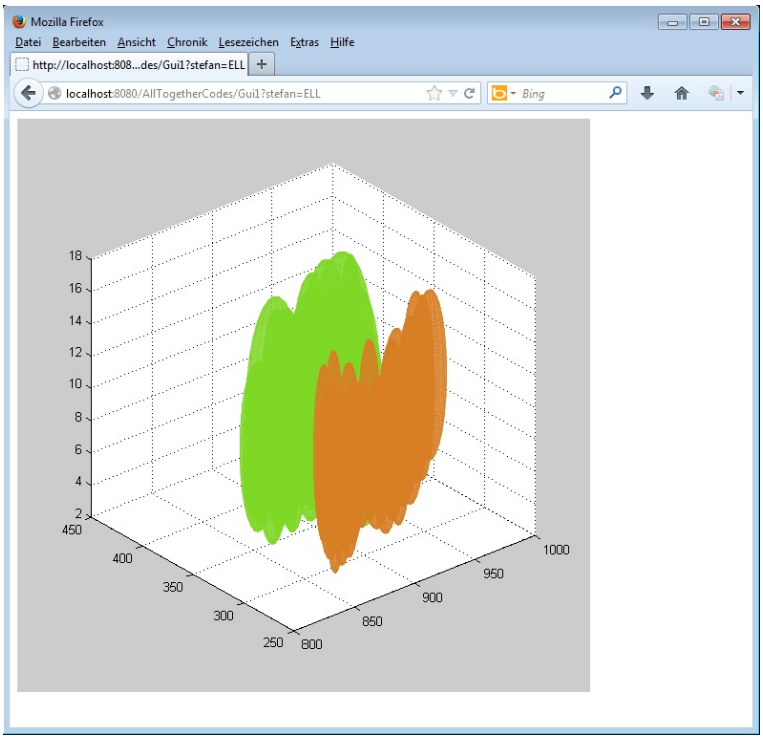


Figure 28: ellipsoidal representation