Technische Universität München
Lehrstuhl für Kommunikationsnetze
Prof. Dr.-Ing. Wolfgang Kellerer

Research Internship Report

Abstract Road Network Topologies for Network Planning

Author:	Patri, Sai Kireet
Matriculation Number:	03669297
Supervisor:	Elena Grigoreva
Begin:	24. August 2016
End:	20. December 2016

With my signature below, I assert that the work in this research internship has been composed by myself independently and no source materials or aids other than those mentioned in this report have been used.

München, 11.12.2016
------------------------------- -------------------------------
Place, Date Signature

München, 11.12.2016
------------------------------- -------------------------------
Place, Date Signature

Abstract

This document is a report of the research internship that was carried out as a part of meeting the requirements for Masters in Communications Engineering. It describes a model to automatically generate different types of Road Networks by using minimal amount of input parameters. These datasets of road networks can be further used for testing and simulation of communication networks. The model is programmed in MATLAB and the different results and comparisons with real city maps are given.

Contents

Chapter 1

Introduction

Network planning is an iterative process, encompassing topological design, network-synthesis, and network-realization. It is aimed at ensuring that a new communications service meets the needs of the customer and operator. An important part of Network Planning is Geographical Network Planning which takes into account specific local geography of an area, e.g. the road distribution, building distribution, natural obstacles and height. However, these properties in a city vary from area to area and cannot be used to test communication networks and make generic conclusions. Here arises a need for generation of abstract topologies for road networks, which a user can manipulate and use for analysis and testing. In this internship, an abstract two dimensional model is created to generate random road networks based on a tensor-field and tracing hyperstreamlines approach [CEW+08].

The model can generate road networks in the style of Manhattan Grid, Radial Grid or a combination thereof. The road network density can also be varied and a number of parameters such as Map size, generation of seeds for tracing hyperstreamlines, probability of big roads in the map, are user modifiable. The system is presented as a MATLAB program, in which a user gives the required input and the road network output is generated.

The organization of this report is as follows. Chapter 2 deals with the main research question and the details on how the problem is formulated. Subsequently, all the theoretical aspects related to the internship, such as Tensor Fields and Hyperstreamlines are discussed in detail. Chapter 3 deals with the details of the implementation of the various modules and then moves on to show the various results which are generated. The results also contain the comparison of the designed system with maps of real cities. Finally, Chapter 4 states

Chapter 2

Background

The chapter discusses in detail the mathematical approach used in this internship, followed by the research question of this internship, the related work and how the various research papers were found to be suitable or unsuitable to achieve the goals.

2.1. Tensors and Hyperstreamlines

In this work, a tensor, *t*, refers to a 2x2 symmetric matrix which is of the form, as defined in [CEW+08]:

$$R\begin{pmatrix} cos2\theta & sin2\theta \\ sin2\theta & -cos2\theta \end{pmatrix} \text{where } R\geq 0 \text{ and } \theta \in [0, 2\pi). \quad (1)$$

The major eigenvectors of *t* are $\left\{\lambda\begin{pmatrix} cos\theta \\ sin\theta \end{pmatrix} | \ \lambda \neq 0\right\}$ and the minor eigenvectors are $\left\{\lambda \begin{pmatrix} cos\theta+\pi/2 \\ sin\theta+\pi/2 \end{pmatrix} | \ \lambda \neq 0\right\}$.

This implies that the major and minor eigenvectors are perpendicular to each other. A tensor field, T is a continous collection of tensors, that associates every point **p = (x,y)** $\in \mathbb{R}^2$ in the Point Space with a tensor given by T(**p**). Different tensor fields, each having its own basis are generated and then they are combined to form a combination of tensor fields. In this implementation, we use two major basis fields as inputs, namely, Grid Tensor basis field and Radial Tensor basis field. [CEW+08] also use Heightfield information and Boundary Field information, in the form of input height maps and natural boundaries map, however due to lack of input maps, this work does not include them.

Hyperstreamlines are a set of vectors which define a curve that is a tangent to an eigenvector field everywhere along their path. Depending on the type of the eigenvector field, a hyperstreamline can be termed as major or minor. However, it must be noted that each type of road, whether it be major or minor, has its own major and minor hyperstreamline.

2.1.1 Grid Tensor Fields

The grid tensor field is used to model Manhattan Grids, which means that the roads are straight lines and intersect each other orthogonally. Given a direction, (u_x, u_y) for a point **p** in the Point space, we define the grid tensor to be:

$$T(\mathbf{p}) = l\begin{pmatrix} cos2\theta & sin2\theta \\ sin2\theta & -cos2\theta \end{pmatrix} \text{ where } l = \sqrt{u_x^2 + u_y^2} \text{ and } \theta = \arctan\left(\frac{ux}{uy}\right). \quad (2)$$

This basis field helps us generate major and minor hyperstreamlines with constant direction and angle throughout the network. The eigenvector direction does not depend on the location of the point **p** in the Point Space. For a purely Manhattan Grid network, we need only the Grid Tensor Fields in our model.

2.1.2 Radial Tensor Fields

Radial road network pattern is used to model road networks, which have a circular or an oval pattern. These patterns of roads are rare in cities in the United States, but are very common in European and Asian cities. Purely radial road networks can be found in parts of cities like Paris, Berlin or Prague. However, it is understood that radial networks are much noisier since there is always a difference between the tensor fields of two different points. To create a radial structure, we need a radial context, or a point which we take as the center for all the radial activity. This central design element will have major hyperstreamlines which are concentric circles and minor hyperstreamlines which are straight lines emanating from the center. First, we define a center point $\mathbf{p_o} = (x_0, y_0)$ and each tensor in the Point space is then defined by:

$$T(\mathbf{p}) = \begin{pmatrix} y^2 - x^2 & -2xy \\ -2xy & x^2 - y^2 \end{pmatrix} \text{ where } x = x_\mathbf{p} - x_0 \text{ and } y = y_\mathbf{p} - y_0 \qquad (3)$$

2.1.3 Combining Tensor Fields

One of the properties of tensor fields is that two different basis tensors corresponding to the same point **p** can be summed up to obtain a combined tensor field. This combined tensor field is then used to trace major and minor hyperstreamlines. Here, every user specification is used to create a global basis tensor field as described in [CEW+08]. These basis fields are then summed up using radial basis functions such that the resulting tensor field satisfies the requirements of the user. The equation used is as follows:

$$T(\boldsymbol{p}) = \sum_i e^{-d\|\mathbf{p}-\mathbf{p}_i\|^2} T_i(\mathbf{p}). \qquad (4)$$

Where d is the decay constant, **p** is the point in the computational domain; $\mathbf{p}_i$ is the point of the basis element currently under computation in the Point space, where i varies from 1 to the size of the map, inintegral steps of 1; T_i is the basis tensor field corresponding to the design element (grid or radial).

Due to the inclusion of randomness and the need of varying the input parameters to obtain desired road network pattern, the combination equation has been modified to get desirable output. This modification is explained in Section 3.1.1 of Chapter 3.

2.1.4 Seed Generation and Hyperstreamline Tracing

Once the combined tensor field is generated, the road network needs to be built like a graph having nodes and edges. The main steps involve generating seeds all across the network, tracing hyperstreamlines using those seeds as starting point, by using the combined tensor fields to guide the hyperstreamlines and then converting the hyperstreamlines into a graph G = (N,E) where N represents the nodes and E edges. It has to be noted that, in the implementation of road network, a node N may or may not mean an intersection. It can also be a node between two edges which are part of the same major or minor road.

1) Seed Generation

The seeding scheme used by [CEW+08] is a fairly complex one. Since these seeds are used to determine the starting point of the hyperstreamlines, these seeds are assigned to a priority queue and each one is placed at a minimum distance away from each other in all directions. Seeds which are at the minimum allowed distance to each other get a higher priority. Once the tracing of the hyperstreamline starts, new seeds are added to the set of the existing seeds in a procedural fashion. However, in this internship, a simpler and a more random seeding technique is used which gives satisfactory results.

2) Hyperstreamline Tracing

[CEW+08] use an adaptive Runge-Kutta scheme to calculate hyperstreamlines. For any particular point in the Point Space, we first need the eigenvectors of that point and the direction of the last propagation. Once the eigenvectors are found, we use the method to find the direction of the current point and also the next integration point. The propagation direction is calculated from only the magnitude and the sign is usually ignored. Furthermore, whenever a hyperstreamline approaches within the minimum allowable distance of another hyperstreamline or of a seed, it is stopped. The seeding scheme used in the internship is a simplified version of this method.

3) Graph Generation

Once the hyperstreamlines are traced, the intersections of all the major and minor hyperstreamlines and the lines themselves are used to form a graph with vertices and edges. However, there occurs sometimes an issue where two vertices of the graph are too close to each other. To avoid this, a few functions which try to remove separate intersections being generated in proximity to each other are used. In [CEW$^+$08], create a user design element which allows users to introduce some noise or to relax a few constraints and edit the generated network.

2.2. Research Question

The main research question of this internship revolves around building a road network with the most minimal inputs. The methods to use a geographic approach often provide accurate maps, but involve extensively manual work. Moreover, current open-source solutions like Open Street Map (https://openstreetmap.org/) do not contain accurate building and natural obstacle information, since anyone can edit the information. For example, in OSM, a small road maybe marked as a large road or a patio in a park maybe marked as a building. Such anomalies generate inconsistent input assumptions which lead to unforeseeable errors while testing networks. Proprietary solutions do exist in the market, but they are not developed for network planning, but for applications like computer games development. Communication network researchers need models which are quick and efficient, while preserving the main geographical characteristics. To tackle these issues, we consider finding a more generic solution which would be independent of a concrete topology as defined by [CEW+08]. A need arises for development of different types of road network topologies with different input parameters, which need to be defined. Once the parameters are optimized and the map is generated, they can be validated against real maps to test their accuracy.

2.3. Related Work

In the Multimedia and Computer Vision research field, procedural generation is used to randomly create large amounts of data. A model which inculcates procedural generation into its system is known as a Procedural Model. Several papers like [PM01], [Bor03] and [TT08] use procedural techniques for creating urban maps for various applications.

[PM01] were the first to point out that road network modeling lays at the core of any urban network generation. Their procedural algorithm starts with one major road and then proceeds to add more road segments, resulting in a tree like structure. The authors used an L-system method of procedural generation. However, L-Systems do not give any control to the user over their generation. And although they achieve a high level of detail, they often have less control over the output. Hence, for a certain input parameters, similar results cannot be guaranteed. Since we look for parameters to control the output, L-systems cannot be used as a direction to proceed in.

To gain more control over the resulting topology, we use the approach of building a road network as described in [CEW+08]. The authors use generation of tensor fields to guide the generation of street networks. The reason for using tensors is that a tensor field generally gives rise to two sets of hyperstreamlines, one following the major eigenvector field and the other following the minor eigenvector field. These major and minor hyperstreamlines can be utilized to correspond to the two dominant directions in the Point space which are needed to draw road networks.

However, while using [CEW+08], it is evident that the hyperstreamline tracing is complex and uses a lot of computing power, hence a simpler solution for hyperstreamline tracing by the method of bilinear interpolation is used, as described in [Cha09].

Chapter 3

Implementation/Results

In this chapter, we discuss the implementation of the road network modelling in Section 3.1, along with the changes made to the equations in [CEW+08] so that the output better suited our requirements. In Section 3.2, the various outputs along with the corresponding input parameters are discussed in detail.

3.1. Implementation

As discussed in Section 2, the inspiration to model road networks based on Tensor fields comes from [CEW+08], however due to certain aspects like seed generation and hyperstreamline tracing being extremely complex, we employ easier approach given in other papers which deal with procedural street network generation. Before we start generating our tensor fields, we need to define the first parameter, which is the space in which our map is generated. For that we need to take a point space in x and y direction. It is suggested to create a vector of points [1:1:50] both in the x and y directions, however, this algorithm works well for point space up to 200. Since with the increase in the number of points, the number of calculation increases exponentially, we need to keep in mind the runtime of the program alongwith the computational capabilities of the processor. Once the point space has been defined, further steps viz. generation of tensor fields, tracing hyperstreamlines, generating edges, clustering vertices and generating graph can be executed.

3.1.1 Generation of Tensor Fields

The direction (u_x,u_y) of each point in the x and y space is calculated as a random function of the difference between the maximum value and the minimum value of the point space. Hence, the direction of each point will remain almost constant throughout network. Using Equation (1) we calculate the grid tensor at each point of the point space.

In Section 2.3.2, we generate Radial Tensor Field as shown in Equation (2). Here we use a central design element from where the minor and major hyperstreamlines appear to start. This central design element is fixed at (x_0, y_0) at (-50, -50). This is done since we observe that if the central design element is in the visible point space map (i.e. between (0, 0) and (50, 50)), we see a lot of uncontrollable and random roads which cannot be smoothened by current techniques.

Once the tensor fields are generated, they are combined using the following equation

$$T(\boldsymbol{x},\boldsymbol{y}) = e^{-\mathrm{d_{grid}}(\|\mathbf{x}-\mathbf{x_0}\|^2+\|\mathbf{y}-\mathbf{y_0}\|^2)}\mathrm{T_{grid}}(\boldsymbol{x},\boldsymbol{y}) + e^{-\mathrm{d_{rad}}(\|\mathbf{x}-\mathbf{x_0}\|^2+\|\mathbf{y}-\mathbf{y_0}\|^2)}\mathrm{T_{rad}}(\boldsymbol{x},\boldsymbol{y}). \qquad (5)$$

Where d_{grid} and d_{rad} are the decay constants for grid and radial tensor fields respectively, x and y are the points in x point space and y point space for which we need to find the combined tensor field; x_0 and y_0 are the co-ordinates of the central design element we assign while calculating radial tensor fields. T_{grid} and T_{rad} are the grid and radial tensor field at the particular point.

We can use the parameters d_{grid} and d_{rad} to vary the grid and radial tensor influence. With a greater value of d_{grid} and a small value of d_{rad} we get a purely Manhattan Grid. For a purely radial grid, we need to have a low value of d_{grid} and a high value of d_{rad} and for a mixed grid, which has almost equal influences from both the tensor fields; we need both the parameters to be at comparable values. The user of the program is given a choice to choose the desired type of road network. Based on the choice, preset values are assigned to these parameters to generate the road network as per user specification. Table 1 gives the typical values of d_{grid} and d_{rad} for different road network structures.

	Manhattan	Radial	Mixed
Grid decay constant	-5	10	-0.2
Radial decay constant	10	-6	-0.95

Table 1: Typical values of grid and radial decay constant to obtain different grids, as used in Equation (5)

3.1.2 Seeding Scheme

Once the tensors are created, seeds for the starting point of hyperstreamlines need to be generated. Here a simple method to generate seeds is used. Since the density of seeds generated is directly proportional to road network density, we can define a parameter here which can be altered to get rural, urban or dense urban road networks. The seeding scheme then uses this parameter to generate all the seeds at the beginning and does not use any priority queue. The parameter seed density is used as an exponent to generate the number of seeds and the density of the seeds also reduces as we move away from the central design element of the radial tensor field. Equation (6) defines the method used to calculate seeds.

$$[x_1] = \left(\min(x):3:\sqrt[\gamma]{\max(x)}\right)^{\gamma}$$
$$[y_1] = \left(\min(y):3:\sqrt[\gamma]{\max(y)}\right)^{\gamma}$$

$$[Xseed \quad Yseed] = meshgrid(x_1, y_1) \tag{6}$$

Where x_1 is the set of points as an input for x co-ordinates of seeds, y_1 is the set of input points for y co-ordinates of seeds and γ is the seed density exponent, whose typical values are shown in the table below.

However, we face a disadvantage with this method of seed generation that while tracing the hyperstreamlines, we find that the junctions of the road network are sometimes placed too close to each other. To avoid this, we use Node reduction methods as described in Section 3.1.4. Table 1 gives the parameter values of seed density for different types of road networks.

Road Network Density	Seed Density exponent
Rural	1.95
Urban	1.6
Dense Urban	1.4

Table 2: Typical values of seed density exponent to get varied road network density used in Equation (6)

3.1.3 Hyperstreamline Tracing and Edge Generation

Hyperstreamline tracing in the model given by [CEW+08] requires an adaptive Runge-Kutta scheme to find the direction of the propagation of hyperstreamlines. Instead, a simpler method of bilinear interpolation is used which is explained in detail in [Cha09]. Interpolation can be used to determine the point nearest to a given point assuming that the two-dimensional point space is continuous and there is a high correlation between the neighbouring points.

To use this in hyperstreamline tracing, the propagation direction is determined by taking a weighted average of four samples of the eigenvectors close to the point which is being evaluated. The formula to determine the next point can be given as follows:

$$Z = A_0(1-x_w)(1-y_w) + A_1(x_w)(1-y_w) + A_2(y_w)(1-x_w)+ A_3x_wy_w \quad (7)$$

Where Z is the point (x,y) to be evaluated. The four samples near the point to be evaluated are taken by using floor and ceiling functions. A_0 = [floor(x), floor(y)], A_1 = [ceil(x), ceil(y)], A_2 = [floor(x), ceil(y)] and A_3 = [ceil(x), floor(y)]. The weights for each of these co-efficients is defined as x_w = x-floor(x) and y_w = y-floor(y).

In this method, assuming a seed as a starting point, the distance weighted average of four points in positive x, negative x, positive y and negative y is found and then a new point value is estimated. In the tracing function, the resampling is repeated for 300 times or until the function encounters a point which is a seed. If the interpolated point is the same as the last known point, the direction is reversed. This method helps in creating a vector of major and minor hyperstreamline points throughout the two-dimensional point space.

Once the hyperstreamline is traced, it is evident that every point where a hyperstreamlinestarts, is one of the majot nodes in the graph. However, these are not the only places where a major node may exist. Here, we introduce a new parameter known as the Major Road Probability, which increases the number of major nodes in the system. The major road probability can lie between 0 and 1. For our calculations, we fix it at 0.3. The major road probability factor produces most ideal results when kept below 0.4. Every node is then assigned a number and the node numbers are arranged in an ascending order, also taking into account if the node is a major node or a minor node.

Once the nodes are formed and numbered, a relationship between each node can be derived based on the direction of the hyperstreamlines and edges can be generated. A few things to keep in mind while forming edges is that every edge should flow from a smaller node to a higher node and there can be no edges having zero length. Once the node and edge relationship is formed, it needs to be determined whether the edge is a part of a major road or a minor road. If the edge lies between two major nodes, it is considered to be a major road and for any other case, it is a minor road.

3.1.4 Node Reduction and Finding Junctions

After creating all the nodes, there are still many nodes in the graph network which might be incomplete, which means it may have number of edges coming in and nothing going out, or no edges coming in and an edge going out. This cannot be allowed since the edges are allowed to end only at the seed points. To tackle this issue, all the edges are co-related to their respective nodes and for each node; number of incoming edges and outgoing edges are calculated. Using graph theory, we can quickly eliminate nodes which do not satisfy the rank of the directions (incoming or outgoing) of the edges to be at least 2. This finally leaves us with a more or less complete graph. The anomalies in the graph can be then attributed to random placement of seeds during the seed generation phase.

Using MATLAB’s internal function of histogram with the input parameter as the edges, we can find out the indices of nodes and the number of times each node is accessed. If the number of time a node is passed is greater than 2, the node then becomes a junction or an intersection in the graph. It is then added to the list of junctions and the nodes, edges and junctions can be scattered on the map.

3.2. Results

Since the procedural model gives us flexibility to choose the size of the map, the type of the road network and the density of the road network, several permutations and combinations to generate road networks can be derived. The results for Manhattan grid Urban and Rural, Mixed Grid Urban and Rural and Radial Grid Urban and Rural will be shared. The parameter of the size of map has been fixed at 50x50 m. This means that the point space in the x direction lies between 1 and 50 and point space in the y direction lies between 1 and 50. Once all the cases have been presented, three different grid types are compared with maps of real cities at 50x50 m resolution to check for similarities.

3.2.1 Different Grid Results

Here we define the following input parameters into our model and then the output is generated in the form of a map.

1) Pure Manhattan Urban Grid

a) Input parameters

Parameter	Resolution	Grid Decay Const	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	-5	10	1.6	0.3

Table 3.2.1 Input Parameters for Manhattan Grid Urban Road Network

b) Results

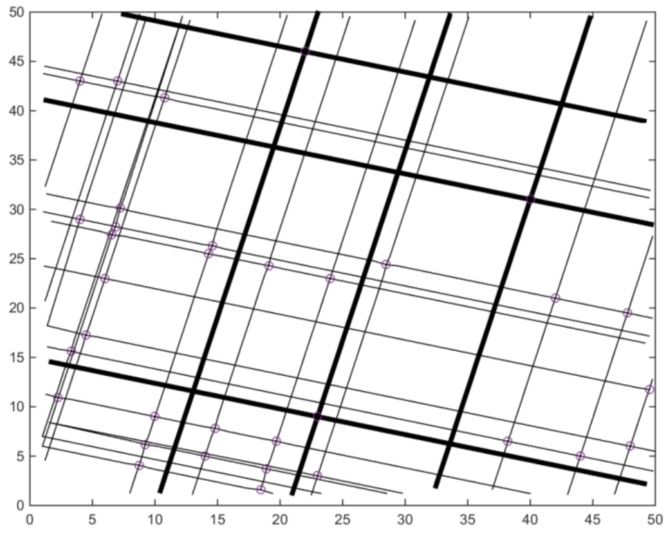

Fig 3.2.1: Manhattan Grid in an Urban Road Network

The dark lines signify major roads and the lighter ones signify minor roads. The blue markers signify the position of the starting seeds generated by the seeding scheme. As an observation, we can note that this network has 31 nodes from which hyperstreamlines start, a total of 164 nodes and 171 edges and the number of junctions or intersections is equal to 34. This proves the statement made in Section 3.1.3 that Hyperstreamline starting points correspond to major nodes of the graph.

2) *Pure Manhattan Rural Grid*

a) Input parameters

Parameter	Resolution	Grid Decay Const	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	-5	10	1.95	0.1

Table 3.2.2 Input Parameters for Manhattan Grid Urban Road Network

b) Results

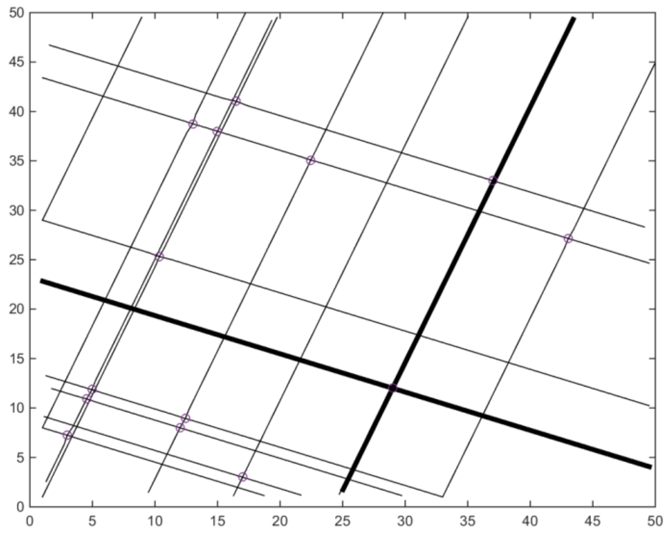

Fig 3.2.2: Manhattan Grid in a Rural Road Network

As we can see in the rural grid, lesser number of roads exist and there are generally more areas between the roads. If compared with the Manhattan Urban Grid Network, we can see that only with the change of the seed density index and the major Roads probability, we can vary our graph.

3) *Pure Radial Urban Grid*

a) Input parameters

Parameter	Resolution	Grid Decay Const	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	10	-6	1.6	0.3

Table 3.2.3 Input Parameters for Radial Grid Urban Road Network

b) Results

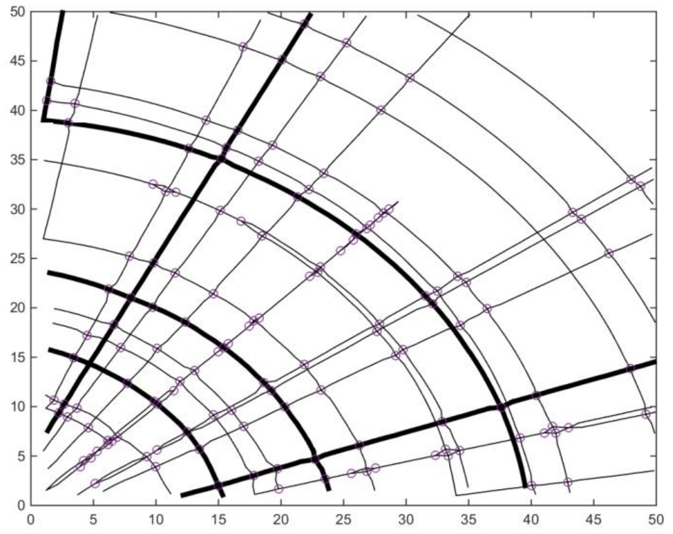

Fig3.2.3: Radial Grid in an Urban Road Network

In the pure radial grid, we keep the central design element out of the purview of our box since it is extremely crowded at the center with hyperstreamlines, nodes and intersections which leads to haphazard tracing of the hyperstreamlines. Generally, a pure radial network is the noisiest of all the configurations since all the parameter keep changing at every point. However, we can see here that some of the minor roads stop abruptly which is also quite evident in real city maps.

4) *Pure Radial Rural Grid*

a) Input parameters

Parameter	Resolution	Grid Decay Const	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	10	-6	1.95	0.1

Table 3.2.4 Input Parameters for Radial Grid Rural Road Network

b) Results

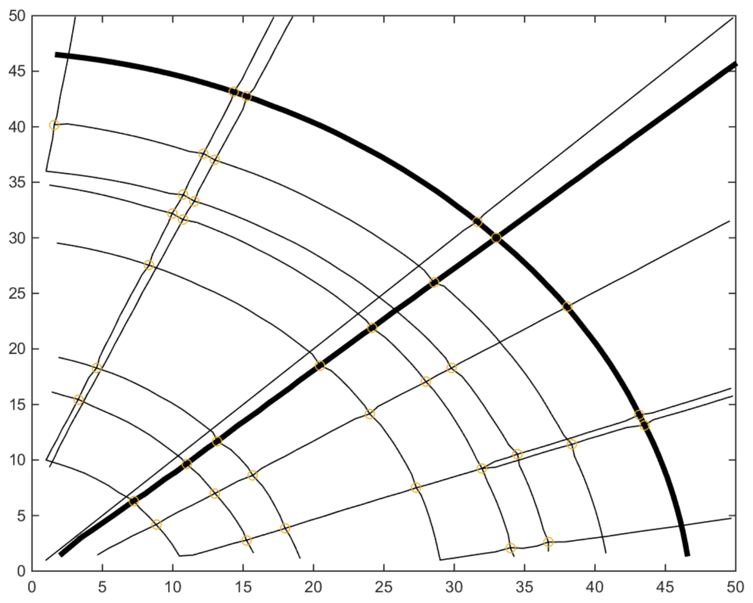

Fig3.2.4: Radial Grid in a Rural Road Network

The rural grids are always less noisy since with the reduction in the seed density, the number of hyperstreamline start nodes do not intersect which leads to lesser interference and smoother road networks.

5) *Mixed Urban Grid*

a) Input parameters

Parameter	Resolution	Grid Decay constant	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	-0.2	-0.95	1.6	0.3

Table 3.2.5 Input Parameters for Mixed Grid Urban Road Network

b) Results

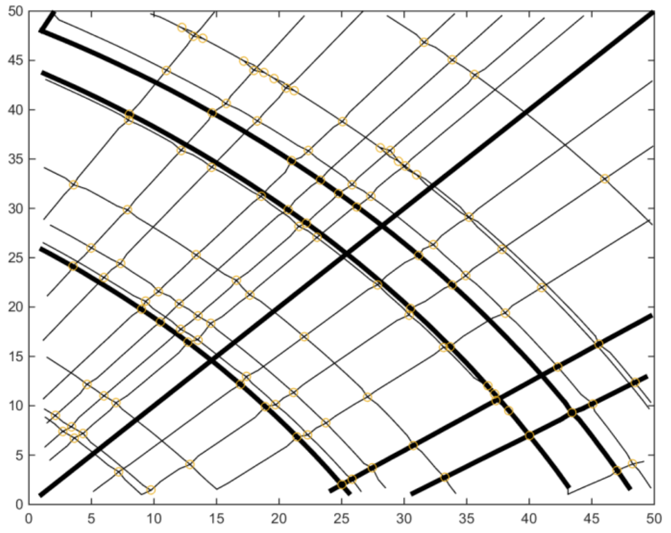

Fig3.2.5: Mixed Grid in an Urban Road Network

As we can see in the mixed urban grid, the major roads follow a radial road network with a larger radius of curvature but the minor roads form a Manhattan Grid pattern. This sort of design is common in most of the cities, where the major roads form rings and link roads around the city whereas the roads connecting business and residential areas are generally in a Manhattan Grid-like arrangement.

6) Mixed Rural Grid

a) Input parameters

Parameter	Resolution	Grid Decay constant	Radial Decay constant	Seed Density Exponent	Major Road Probability
Value	50x50	-0.2	-0.95	1.95	0.1

Table 3.2.6 Input Parameters for Mixed Grid Urban Road Network

b) Results

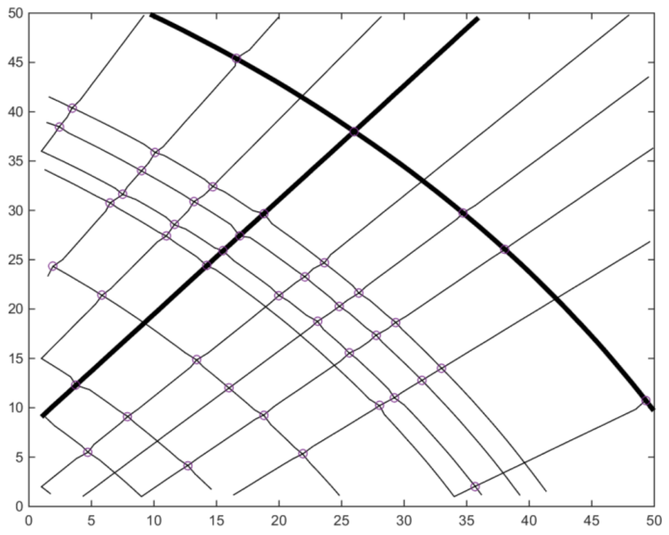

Fig3.2.6: Mixed Grid in a Rural Road Network

3.2.2 Grid Results compared to Real City Maps

In this section we take cross sections of a real city's map from Open Street Maps and try to generate an almost similar map from the designed model. To find the input parameters, a trial and error approach was used. However, most of the input parameters, like seed density, input resolution and Major Road Probability can be approximated. The real challenge occurs in getting the radial and the grid decay constant right. As we can see from the obtained result, we do get similar networks, however some anomalies do exist, which have been duly explained in each of the sections below.

1) Manhattan Grid

Figure 3.2.8 shows a 50x50 m map of the area around Technische Universität München. To simulate the same in our model, we need to observe that this is a pure Manhattan Grid in an Urban scenario. We see 4 major roads and several minor roads, hence this has a higher Major Road Probability. Hence we can input the same parameters into our model and generate a similar map.

Input Parameters

Parameter	Resolution	Grid Decay constant	Radial Decay constant	Seed Density Exponent	Major Road Probability	Central Design element
Value	50x50	-6	10	1.6	0.3	(-5,-5)

Table 3.2.7 Input Parameters to simulate Urban Munich road network

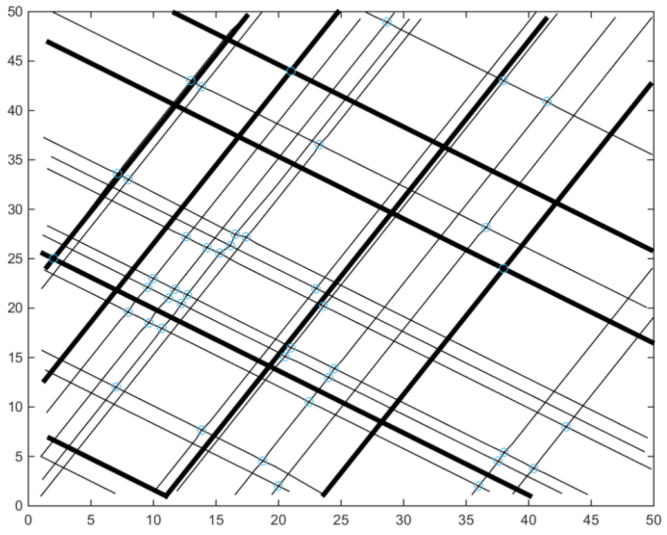

Fig 3.2.7: Pure Manhattan Grid simulating 50x50 area Munich, Germany

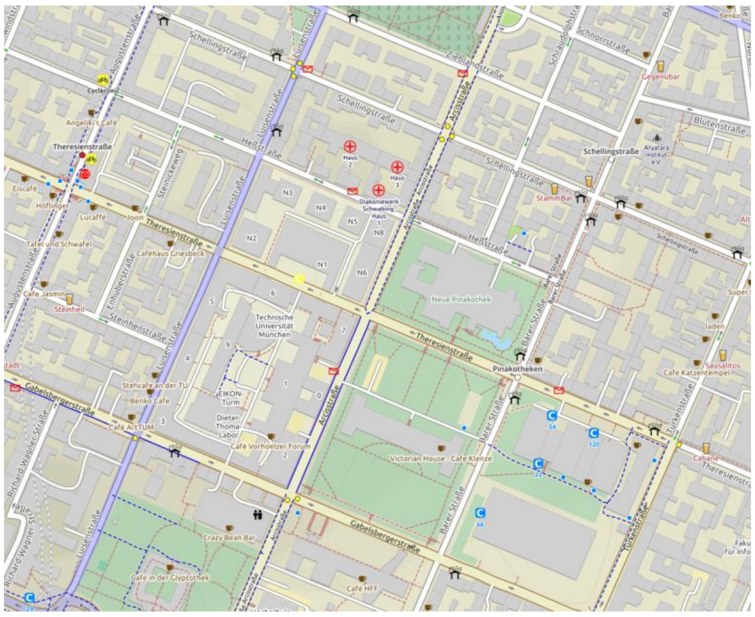

Fig 3.2.8: Map of Urban Munich (Image Courtesy: Open Street Maps)

2) *Radial Grid*

Although it is difficult to find many cities with a complete radial structure, there are parts of cities which do have a radial structure. This maybe either to divert traffic efficiently or for beautification purposes. One such example is Connaught Place in New Delhi, India, which has almost an entirely radial structure. As is visible, purely radial road networks are prone to noise and are generally not used. For this example, the central design element is shifted inside the graph. Due to this, we see that the hyperstreamlines around the center are more concentrated and more haphazard. This happens due to the usage of bilinear interpolation for Hyperstreamline tracing. The input parameters are defined as follows.

Input Parameters

Parameter	Resolution	Grid Decay constant	Radial Decay constant	Seed Density Exponent	Major Road Probability	Central design element
Value	50x50	10	-6	1.2	0.1	(20,20)

Table 3.2.8 Input Parameters to simulate Central New Delhi road network

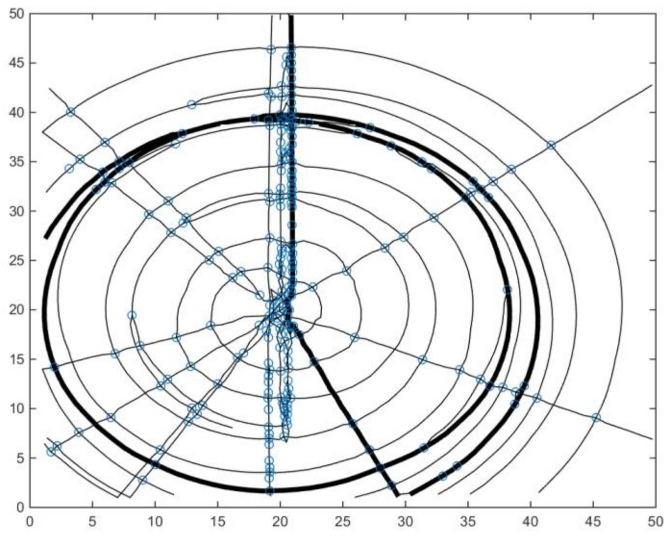

Fig 3.2.9: Pure Radial Grid simulating 50x50m area of Connaught Place, New Delhi, India

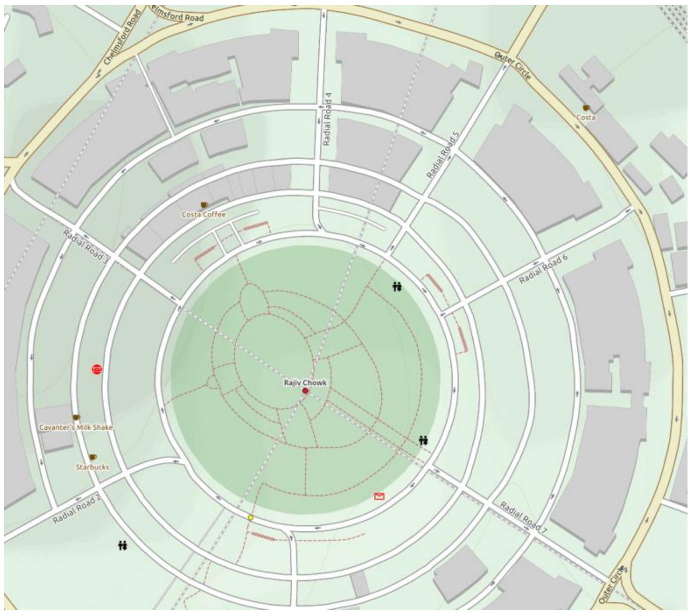

Fig 3.2.10: Map of Connaught Place, New Delhi, India (Image Courtesy: Open Street Maps)

3) Mixed Grid

For a scenario of Mixed Grid in a dense urban scenario, we take a 50x50 m cross section of Berlin's Pankow district. This area of Berlin has major roads with radial curves but has Manhattan Grid in between the major roads. To get such a structure, we need to consider the seed density for dense urban networks and the central design element is shifted to (-100,-100) to get a smoother result in the 50x50 cross section. The model is able to generate a similar structure, although, since more seeds are involved, the minor roads far outweigh the number of major roads giving the road network a denser look. We also see that the roads emanating from the centre do not have any curvature in them. While a curvature on the roads is possible to be obtained using only the radial decay constant, it is advisable to look for more viable solutions. One such approach maybe, to include some heightfield or boundary field influence. Another solution as suggested by [CEW+08] is addition of Perlin Noise to include rotation in the system.

Input Parameters

Parameter	Resolution	Grid Decay Const	Radial Decay constant	Seed Density Exponent	Major Road Probability	Central design element
Value	50x50	-0.4	-1.1	1.8	0.1	(-100,-100)

Table 3.2.9 Input Parameters to simulate Berlin road network

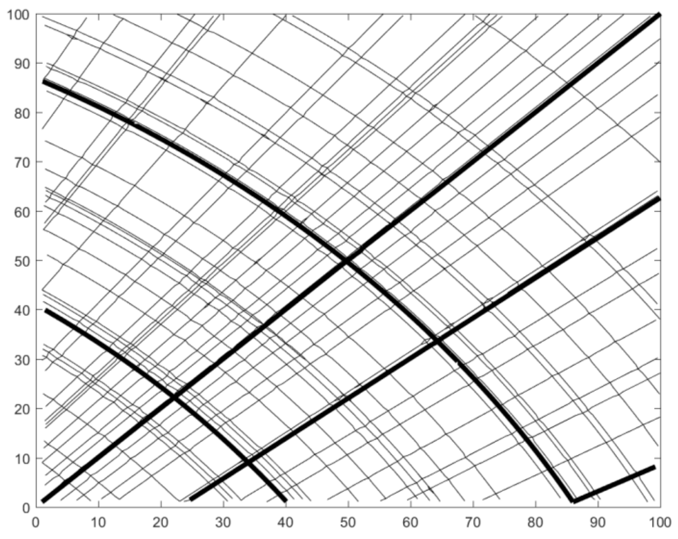

Fig 3.2.11: Mixed Grid simulating 50x50m area of Berlin, Germany

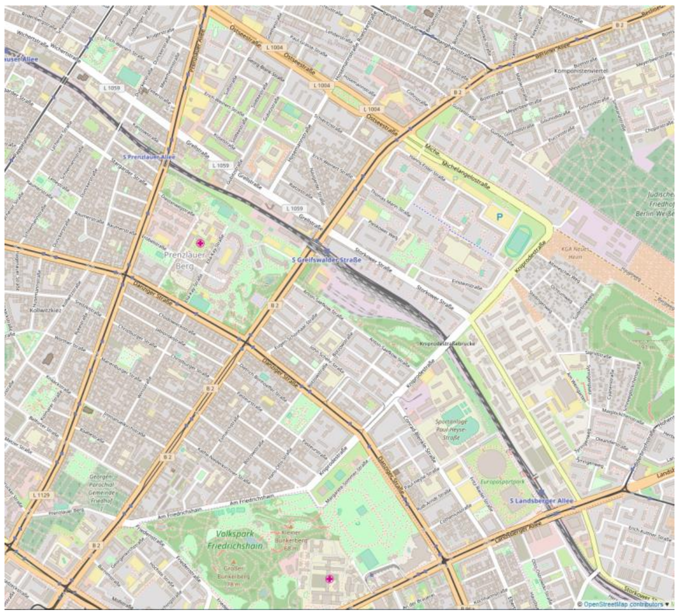

Fig 3.2.12: Map of Pankow Borough, Berlin, Germany (Image Courtesy: Open Street Maps)

Chapter 4

Conclusions and Outlook

The research question for this internship was to create an abstract model to generate a road network which could be used for testing of wireless or optical networks for a wide range of application.
As is seen from the results, using a procedural algorithm as designed in this internship, it is possible to model road networks similar to those existing in various cities around the world. This functionality saves network planners and researchers a lot of time and effort to create road networks for their various tests. The model was programmed in MATLAB and mainly follows a Tensor Field Design approach as described in [CEW+08]. At the moment the model is able to generate a basic road structure, but based on availability of data, it is easy to add further inputs like Height field or Boundary field to make the maps look more realistic. Tracing of Hyperstreamlines is another place where improvement is possible, since we use a simple method of bilinear interpolation as described in [Cha09], instead of the more complex Adaptive Runge-Kutta Method. Further, rotation of the combined field can be added using Perlin Noise and Laplacian smoothing can be done to remove subsequent irregularities.

After optimizing rad networks, the next step is to use a similar approach to procedurally place buildings of different heights in the city. These can be helpful in answering research questions of where to place wireless receivers and base stations in order to ensure maximum coverage in a city. Or it can be helpful in planning the costs required for laying Fiber to the Building in Optical Networks.

Furthermore, the current MATLAB program can be presented as a toolkit or a MATLAB Graphical User Interface where the input parameters are entered and a map is returned as an output.

Bibliography

[CEW$^+$08] G. Chen, G. Esch, P. Wonka, P. Müller, and E. Zhang, "Interactive Procedural Street Modeling," *ACM Trans. Graph. Artic.*, vol. 27, no. 10, 2008.

[Del93] T. Delmarcelle, "Visualizing Second-Order Tensor Fields with Hyperstreaml ines," vol. 7, no. July, pp. 25–33, 1993.

[Bor03] G. Borruso, "Network Density and the Delimitation of Urban Areas 1 Network Geography and the Geography of Networks," *Trans. GIS*, vol. 7, no. 2, pp. 177–191, 2003.

[TT08] S. T. Teoh and T. Soon Tee, "Algorithms for the automatic generation of urban streets and buildings," *Proc. 2008 Int. Conf. Comput. Graph. Virtual Real. (CGVR 2008)*, no. 1, pp. 122–128, 2008.

[LQC$^+$14] Y. Li, G. Qiao, A. Cai, L. Shi, H. Zhao, and G. Shen, "Microwave backhaul topology planning for wireless access networks," *Int. Conf. Transparent Opt. Networks*, vol. 1, pp. 14–17, 2014.

[PM01] Y. I. H. Parish and P. Müller, "Procedural Modeling of Cities," *28th Annu. Conf. Comput. Graph. Interact. Tech.*, no. August, pp. 301–308, 2001.

[Cha09] Chang, Kang-Tsung. "Computation for bilinear interpolation." *Introduction to Geographic Information Systems (5th ed.), New York, McGraw-Hill* (2009).