TUGAS AKHIR – KI141502

Visualisasi *Coverage* Sekolah Menggunakan Diagram Voronoi dan HTML5

Andrys Daniel Silalahi
NRP. 5112 100 207

Dosen Pembimbing
Ridho Rahman Hariadi, S.Kom., M.Sc.
Anny Yuniarti, S.Kom., M.Comp.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUGAS AKHIR – KI141502

Visualisasi *Coverage* Sekolah Menggunakan Diagram Voronoi dan HTML5

Andrys Daniel Silalahi
NRP. 5112 100 207

Dosen Pembimbing
Ridho Rahman Hariadi, S.Kom., M.Sc.
Anny Yuniarti, S.Kom., M.Comp.Sc.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

(Halaman ini sengaja dikosongkan)

FINAL PROJECT – KI141502

Visualization of Schools Coverage Using Voronoi Diagram and HTML5

Andrys Daniel Silalahi
NRP. 5112 100 092

Advisor
Ridho Rahman Hariadi, S.Kom., M.Sc.
Anny Yuniarti, S.Kom., M.Comp.Sc.

DEPARTMENT OF INFORMATICS
Faculty of Information Technology
Sepuluh Nopember Institute of Technology
Surabaya 2016

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

Visualisasi Coverage Sekolah Menggunakan Diagram Voronoi dan HTML5

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Interaksi Grafika dan Seni
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
Andrys Daniel Silalahi
NRP. 5112100207

Disetujui oleh Pembimbing Tugas Akhir:

1. Ridho Rahman Hariadi, S.Kom., M.Sc. .....
(NIP. *198702132014041001*) (Pembimbing 1)

2. Anny Yuniarti, S.Kom., M.Comp.Sc. .....
(NIP. *198106222005012002*) (Pembimbing 2)

SURABAYA
DESEMBER 2016

(Halaman ini sengaja dikosongkan)

Visualisasi *Coverage* Sekolah Menggunakan Diagram Voronoi dan HTML5

Nama	: Andrys Daniel Silalahi
NRP	: 5112100207
Jurusan	: Teknik Informatika FTIf-ITS
Dosen Pembimbing I	: Ridho Rahman Hariadi, S.Kom., M.Sc.
Dosen Pembimbing II	: Anny Yuniarti, S.Kom., M.Comp.Sc.

ABSTRAK

Terkadang letak dari sekolah-sekolah yang ada dalam suatu wilayah dirasakan tidak merata. Beberapa sekolah didirikan di suatu sisi wilayah yang sama, namun di sisi yang lainnya hanya ada sedikit sekolah bahkan tidak ada. Oleh karena itu, ada baiknya sekolah-sekolah yang ada dalam suatu wilayah dipetakan untuk mengetahui wilayah cakupan masing-masing sekolah agar dapat dilakukan pemeratan letak sekolah dalam wilayah tersebut. Untuk memetakan sekolah-sekolah pada suatu wilayah, dapat digunakan diagram Voronoi. Dengan diagram Voronoi, suatu wilayah akan dipecah menjadi beberapa wilayah bagian berdasarkan titik-titik koordinat letak dari masing-masing sekolah. Dari diagram Voronoi yang dibentuk tersebut, akan diketahui besar wilayah cakupan dari masing-masing sekolah yang ada pada suatu wilayah.

Aplikasi yang dibangun dapat memetakan sekolah-sekolah pada suatu wilayah dalam bentuk diagram Voronoi. Dengan diagram Voronoi yang dihasilkan aplikasi tersebut, dapat diketahui cakupan masing-masing sekolah pada suatu wilayah sehingga mungkin dapat dilakukan pemerataan letak sekolah-sekolah.

Kata kunci: Diagram Voronoi, Pemetaan sekolah, HTML5, Aplikasi berbasis web.

Visualization of Schools Coverage Using Voronoi Diagram and HTML5

Name	: Andrys Daniel Silalahi
NRP	: 5112100207
Department	: Informatics Engineering FTIf-ITS
Advisor I	: Ridho Rahman Hariadi, S.Kom., M.Sc.
Advisor II	: Anny Yuniarti, S.Kom., M.Comp.Sc.

ABSTRACT

Sometimes the location of schools in a region perceived uneven. Some schools set up in a region of the same side, but on the other side there are only a few schools. Therefore, it is better if the schools located within an area mapped to a diagram so that we can determine the coverage area of each school. To map the schools in a region, we can use Voronoi diagram. With Voronoi diagram, a region will be split into several areas based on the location of each school. From the Voronoi diagram, we will know the coverage area of each school in the region. The application built to map the schools in a region to Voronoi diagram. With Voronoi diagram generated by the application, we can see each school coverage in an area.

Key words: Voronoi diagram, Schools mapping, HTML5, Web-based application

(Halaman ini sengaja dikosongkan).

KATA PENGANTAR

Puji syukur penulis panjatkan kepada Tuhan yang Maha Esa karena atas segala berkat dan pertolongan-Nya penulis dapat menyelesaikan tugas akhir yang berjudul "Visualisasi coverage sekolah menggunakan diagram Voronoi dan HTML5"

Pengerjaan tugas akhir ini penulis lakukan untuk memenuhi salah satu syarat memperoleh gelar Sarjana Komputer di Program Studi S-1 Jurusan Teknik Informatika Fakultas Teknologi Informasi Institut Teknologi Sepuluh Nopember.

Penulis mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan baik secara langsung maupun tidak langsung selama proses pengerjaan tugas akhir ini hingga selesai, antara lain:

1. Tuhan yang Maha Esa yang masih memberikan kesehatan dan kekuatan bagi penulis.
2. Keluarga penulis, Bapak Antoni Silalahi, Ibu Sonti L. Tobing, adik-adik dan juga keluarga yang tidak dapat penulis sebutkan satu per satu yang telah memberi dukungan moral dan material serta doa untuk penulis.
3. Bapak Ridho Rahman Hariadi, S.Kom., M.Sc. selaku dosen pembimbing I yang telah memberikan bimbingan dan arahan dalam pengerjaan tugas akhir ini.
4. Ibu Anny Yuniarti, S.Kom., M.Comp.Sc. selaku dosen pembimbing II yang telah memberikan bimbingan dan arahan dalam pengerjaan tugas akhir ini.
5. Bapak, Ibu dosen Jurusan Teknik Informatika ITS yang telah banyak memberikan ilmu dan bimbingan yang tak ternilai harganya bagi penulis.

6. Seluruh staf dan karyawan Jurusan Teknik Informatika ITS yang banyak memberikan kelancaran administrasi akademik kepada penulis.
7. Seluruh pihak yang tidak bisa saya sebutkan satu persatu yang telah memberikan dukungan selama saya menyelesaikan Tugas Akhir ini.

Penulis menyadarai bahwa masih terdapat banyak kekurangan dalam tugas akhir ini. Oleh karena itu, penulis menerima dengan rendah hati kritik dan saran untuk pembelajaran dan perbaikan ke depannya. Semoga tugas akhir ini dapat memberikan manfaat yang sebaik-baiknya.

Surabaya, Desember 2016

Penulis

DAFTAR ISI

(Halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

(Halaman ini sengaja dikosongkan)

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

BAB 1
PENDAHULUAN

Bagian ini akan dijelaskan hal-hal yang menjadi latar belakang, permasalahan yang dihadapi, batasan masalah, tujuan dan manfaat, metodologi dan sistematika penulisan yang digunakan dalam pembuatan tugas akhir ini.

1.1 Latar Belakang

Sekolah adalah sarana pendidikan yang paling penting untuk kehidupan masyarakat khususnya anak-anak. Di Indonesia, jumlah sarana pendidikan ini dapat dikatakan telah mencukupi jumlah kebutuhan masyarakat. Namun, terkadang letak dari sekolah-sekolah yang ada dalam suatu wilayah dirasakan tidak merata. Beberapa sekolah didirikan di suatu sisi wilayah yang sama, namun di sisi yang lainnya hanya ada sedikit sekolah bahkan tidak ada. Oleh karena itu, ada baiknya sekolah-sekolah yang ada dalam suatu wilayah dipetakan untuk mengetahui wilayah cakupan masing-masing sekolah agar dapat dilakukan pemeratan letak sekolah dalam wilayah tersebut.

Untuk memetakan sekolah-sekolah pada suatu wilayah, dapat digunakan diagram Voronoi. Dengan diagram Voronoi, suatu wilayah akan dipecah menjadi beberapa wilayah bagian berdasarkan titik-titik koordinat letak dari masing-masing sekolah. Dari diagram Voronoi yang dibentuk tersebut, akan diketahui besar wilayah cakupan dari masing-masing sekolah yang ada pada suatu wilayah.

Dalam tugas akhir ini, dikembangkan sebuah aplikasi yang dapat memetakan sekolah-sekolah pada suatu wilayah dalam bentuk diagram Voronoi. Aplikasi ini menerima inputan berupa titik-titik koordinat dari sekolah-sekolah dan luasan bidang

wilayahnya. Dari inputan-inputan tersebut, aplikasi akan menghitung dan membagi luasan bidang wilayah menjadi beberapa bagian berdasarkan titik-titik sekolah yang ada, sehingga dihasilkannya pemetaan berupa diagram Voronoi. Dengan diagram Voronoi yang dihasilkan aplikasi tersebut, dapat diketahui cakupan masing-masing sekolah pada suatu wilayah sehingga mungkin dapat dilakukan pemerataan letak sekolah-sekolah.

1.2 Rumusan Masalah

Perumusan masalah yang terdapat pada tugas akhir ini, antara lain adalah:

1. Bagaimana membuat aplikasi yang dapat menghitung sebuah diagram Voronoi dari beberapa titik dan sebuah luasan bidang?
2. Bagaiman membuat aplikasi yang dapat menggambarkan diagram Voronoi?

1.3 Batasan Masalah

Batasan masalah yang terdapat pada tugas akhir ini, aplikasi ini berbasis website dengan bahasa pemrograman HTML5 dan Javascript.

1.4 Tujuan

Tujuan dari pembuatan tugas akhir ini adalah membuat aplikasi yang dapat memetakan wilayah cakupan sekolah-sekolah pada suatu wilayah dalam bentuk diagram Voronoi

1.5 Manfaat

Manfaat dari pembuatan tugas akhir ini, antara lain:

1. Mempermudah pemetaan wilayah cakupan sekolah-sekolah pada suatu wilayah.

2. Dapat membantu analisa pemerataan sekolah-sekolah pada suatu wilayah.

1.6 Metodologi

a. Penyusunan proposal tugas akhir
   Proposal tugas akhir ini berisi tentang deskripsi pendahuluan dari tugas akhir yang akan dibuat. Pendahuluan ini terdiri atas hal yang menjadi latar belakang diajukannya usulan tugas akhir, rumusan masalah yang diangkat, batasan masalah untuk tugas akhir, tujuan dari pembuatan tugas akhir, dan manfaat darihasil pembuatan tugas akhir. Selain itu dijabarkan pula tinjauan pustaka yang digunakan sebagai referensi pendukung pembuatan tugas akhir. Sub bab metodologi berisi penjelasan mengenai tahapan penyusunan tugas akhir mulai dari penyusunan proposal hingga penyusunan buku tugas akhir. Terdapat pula sub bab jadwal kegiatan yang menjelaskan jadwal pengerjaan tugas akhir.

b. Studi literatur
   Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu mengenai diagram Voronoi, HTML5, dan Javascript.

c. Analisis dan desain perangkat lunak
   Fitur yang terdapat pada aplikasi ini adalah:
   1. Menerima input titik koordinat letak sekolah-sekolah dan luasan bidang wilayah.
   2. Menghitung diagram Voronoi berdasarkan inputan.
   3. Menggambarkan diagram Voronoi berdasarkan hasil perhitungan.

d. Pengembangan perangkat lunak
   Pembangunan aplikasi akan dilakukan dengan menggunakan bahasa pemrograman HTML5 dan Javascript.

e. Pengujian dan evaluasi
   Proses pengujian akan dilakukan dengan memberikan inputan berupa koordinat sekolah-sekolah yang ada di kecamatan Sukolilo Surabaya-Jawa Timur untuk dihitung dan dipetakan dalam diagram Voronoi oleh aplikasi. Pengujian dikatakan berhasil apabila aplikasi dapat memetakan sekolah-sekolah inputan ke dalam sebuah diagram Voronoi dan diagram Voronoi yang dihasilkan aplikasi sesuai dengan hasil perhitungan yang seharusnya.

f. Penyusunan buku tugas akhir
   Pada tahap ini dilakukan penyusunan laporan yang menjelaskan dasar teori dan metode yang digunakan dalam tugas akhir ini. Pada tahap ini juga disertakan hasil dari implementasi metode dan algoritma yang telah dibuat. Sistematika penulisan buku tugas akhir ini secara garis besar antara lain:
   1. Pendahuluan
      a. Latar Belakang
      b. Rumusan Masalah
      c. Batasan Tugas Akhir
      d. Tujuan
      e. Metodologi
      f. Sistematika Penulisan
   2. Tinjauan Pustaka
   3. Analisis dan Perancangan
   4. Implementasi

5. Pengujian dan Evaluasi
6. Kesimpulan dan Saran
7. Daftar Pustaka

1.7 Sistematika Penulisan

Buku tugas akhir ini disusun dengan sistematika penulisan sebagai berikut:

BAB I. PENDAHULUAN

Bab ini berisi mengenai latar belakang, tujuan, dan manfaat dari pembuatan tugas akhir. Selain itu, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penulisan juga merupakan bagian dari bab ini.

BAB II. TINJAUAN PUSTAKA

Bab ini berisi penjelasan secara detail mengenai dasar-dasar penunjang dan teori-teori yang digunakan untuk mendukung pembuatan tugas akhir ini.

BAB III. ANALISIS DAN PERANCANGAN

Bab ini membahas tahap analisis dan perancangan dari perangkat lunak yang akan dibangun. Analisis dan perancangan perangkat lunak meliputi perancangan data, arsitektur, proses, dan perancangan antarmuka pada aplikasi.

BAB IV. IMPLEMENTASI

Bab ini membahas implementasi dari rancangan yang telah dibuat pada bab sebelumnya.

BAB V. PENGUJIAN DAN EVALUASI

Bab ini menjelaskan kemampuan perangkat lunak dengan melakukan pengujian kebenaran dan pengujian kinerja dari aplikasi yang telah dibuat.

BAB VI. KESIMPULAN DAN SARAN

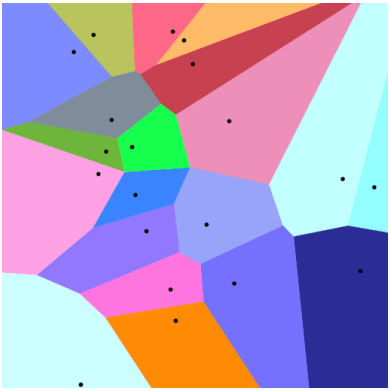
Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak ke depannya.

BAB 2
TINJAUAN PUSTAKA

Bab ini berisi penjelasan teori-teori yang berkaitan dengan metode yang diajukan pada pengimplementasian perangkat lunak. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap sistem yang dibuat dan berguna sebagai penunjang dalam pengembangan perangkat lunak.

2.1 Diagram Voronoi

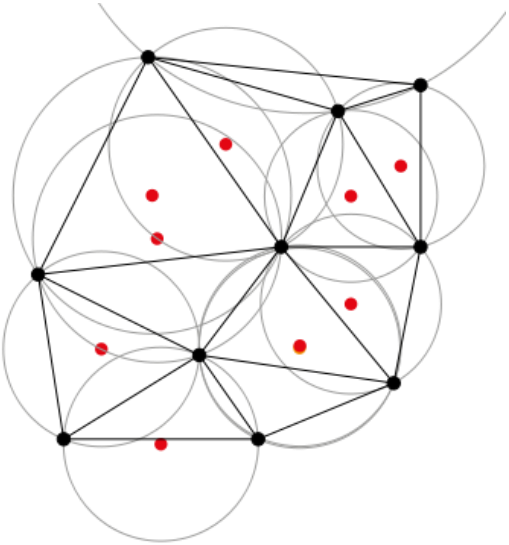
Diagram Voronoi adalah pembagian dari suatu luasan bidang menjadi beberapa bagian berdasarkan jarak dari titik-titik pada sebuah subset spesifik dari luasan bidang tersebut. Himpunan dari titik-titik tersebut ditentukan terlebih dahulu, dan untuk setiap titik memiliki sebuah wilayah masing-masing dimana setiap koordinat pada wilayah tersebut lebih dekat ke titik tersebut daripada titik-titik lainnya. Wilayah-wilayah ini disebut sel-sel Voronoi. Contoh diagram Voronoi dapat dilihat pada Gambar 2.1.



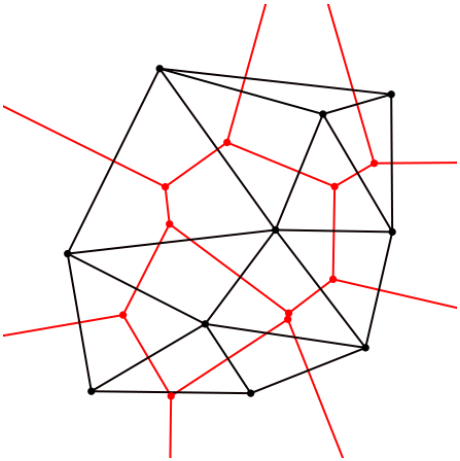
Gambar 2.1. Diagram Voronoi [1]

2.2 Delaunay Triangulation

Delaunay triangulation untuk sebuah himpunan titik pada suatu luasan bidang adalah sebuah *triangulation* dimana tidak ada titik berada di dalam *circumcircle* dari setiap segitiga yang ada pada *triangulation* tersebut. *Delaunay* triangulation merupakan *dual graph* dari diagram Voronoi untuk himpunan titik yang sama. Oleh karena itu, kita dapat membentuk diagram Voronoi dengan membentuk *Delaunay triangulation* terlebih dahulu dan menghubungkan titik tengah dari setiap circumcirle yang ada pada *Delaunay triangulation.* Contoh Delaunay Triangulation dapat dilihat pada Gambar 2.2 dan Gambar 2.3.



Gambar 2.2. Delaunay Triangulation beserta Semua Circumcirle-nya dan Pusat Lingkarannya [2]

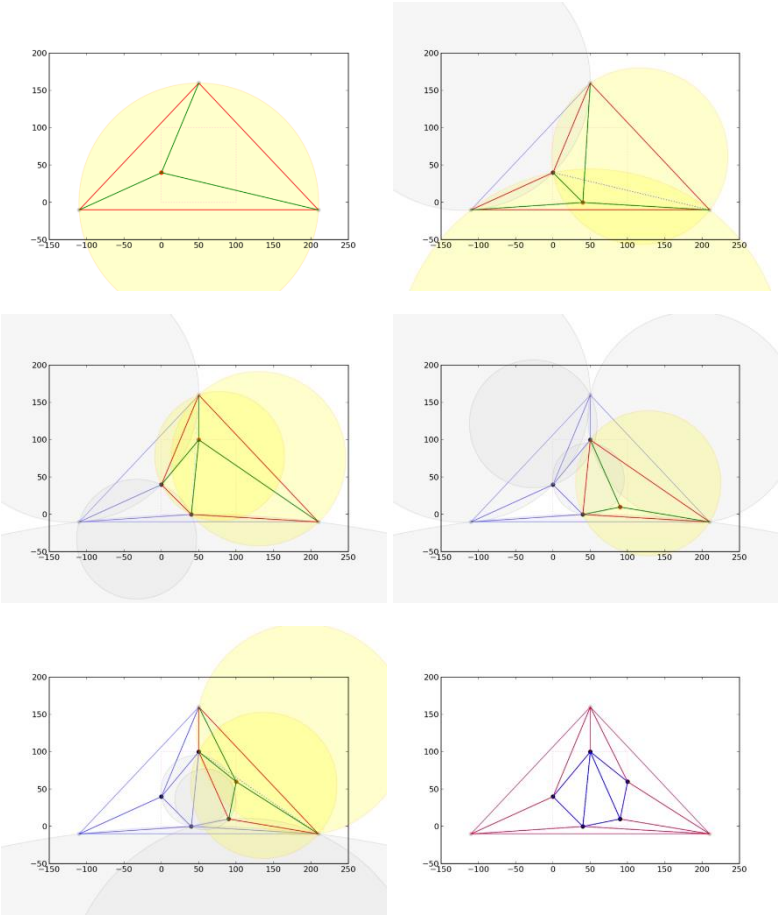


Gambar 2.3. Diagram Voronoi yang Dibentuk dari Delaunay triangulation [2]

2.3 Algoritma Bowyer-Watson

Salah satu algoritma yang digunakan untuk membentuk diagram *triangulation* adalah algoritma Bowyer-Watson. Algoritma Bowyer-Watson merupakan algoritma yang inkremental. Algoritma ini bekerja dengan menambahkan titik satu per satu ke dalam diagram hasil. Ilustrasi tahapan-tahapan algoritma ini dapat dilihat pada Gambar 2.4.

Hal pertama yang dilakukan untuk menghasilkan diagram triangulation adalah membuat sebuah segitiga besar, disebut segitiga super, dimana seluruh titik-titik masukan berada di dalam segitiga tersebut. Setelah segitiga super terbentuk, barulah titik-titik masukan akan diproses dan ditambahkan satu per satu ke dalam diagram hasil.



Gambar 2.4 Algoritma Bowyer-Watson dengan Lima Titik Masukan [3]

Dalam setiap proses penambahan titik-titik masukan ke dalam diagram hasil, beberapa segitiga yang ada pada diagram hasil akan menjadi invalid. Suatu segitiga akan menjadi invalid jika

titik yang ditambahkan berada di dalam *circumcirle* segitiga tersebut. Setiap segitiga yang invalid akan dihapus dari diagram hasil. Kemudian akan dibentuk beberapa segitiga baru dengan titik yang ditambahkan dan sisi-sisi segitiga yang invalid tersebut. Hal ini dilakukan sampai semua titik masukan selesai ditambahkan ke dalam diagram hasil. Tahapan terakhir dari algorima Bowyer-Watson adalah menghilangkan segitiga-segitiga yang memiliki satu atau lebih titik sudut yang sama dengan titik-titik sudut segitiga.

Untuk mengubah diagram *triangulation* menjadi diagram Voronoi, kita dapat memerhatikan setiap dua segitiga yang memiliki sisi sama. Untuk setiap dua segitiga yang memiliki sisi yang sama, maka titik pusat dari *circumcirle* dari kedua segitiga akan dihubungkan dengan garis. Setelah semua segitiga yang mempunyai sisi yang sama dengan segitiga lain dihubungkan, akan terbentuklah diagram Voronoi.

2.4 HTML

HyperText Markup Language (HTML) adalah bahasa *markup* standar untuk membuat halaman web dan aplikasi web. Dengan *Cascading Style Sheets* (CSS), dan JavaScript, ketiganya membentuk serangkai landasan teknologi untuk World Wide Web. *Web-browser* akan menerima dokumen HTML dari *web-server* atau penyimpanan lokal dan me-*render* mereka ke dalam halaman web. HTML juga mendeskripsikan struktur dari sebuah halaman web secara semantik.

HTML terdiri dari elemen-elemen HTML dan hal lainnya (misal teks). Elemen HTML adalah sebuah komponen individual dari sebuah dokumen HTML atau situs web ketika telah di-*parse* menjadi *Document Object Model*. Setiap elemen dapat mempunyai atribut-atribut tersendiri. Setiap elemen juga dapat mempunyai

elemen-elemen lainnya dan teks. Elemen HTML diwakilkan oleh sebuah *tag*, ditulis menggunakan kurung sudut. Contohnya, untuk mendefinisikan judul dari dokumen HTML digunakan *tag* <*title*>. *Browser* tidak akan menampilkan tag HTML, melainkan menggunakan *tag-tag* tersebut untuk menafsirkan isi dari halaman web tersebut.

HTML dapat disertakan dengan program-program yang ditulis dalam bahasa *scripting* seperti JavaScript yang dapat mempengaruhi perilaku dan isi dari halaman web. Untuk mendefinisikan tampilan dan tata letak konten, dapat dicantumkan CSS. Sejak tahun 1997, *World Wide Web Consortium* (W3C), pengelola dari standar HTML dan CSS, telah menganjurkan penggunaan CSS untuk pendefinisian tampilan dan tata letak dari halaman web daripada pendefinisian secara eksplisit di dokumen HTML.

2.5 HTML5

HTML5 adalah revisi kelima dari HTML. Tujuan utama pengembangan HTML5 adalah untuk memperbaiki teknologi HTML agar mendukung teknologi multimedia terbaru, mudah dibaca oleh manusia dan juga mudah dimengerti oleh mesin. HTML5 memperluas dan memperbaiki *markup* yang tersedia, serta memperkenalkan *markup* dan *application programming interface* (API) untuk aplikasi web yang kompleks. HTML5 juga menjadi calon untuk *cross-platform mobile application*, karena memuat fitur-fitur yang dirancang yang sesuai untuk perangkat bertenaga rendah.

Banyak fitur-fitur baru yang disertakan pada HTML5. Untuk menangani multimedia dan konten grafis, elemen <*video*>, <*audio*> dan <*canvas*> ditambahkan. Untuk memperkaya konten semantik dari dokumen HTML, terdapat elemen-elemen struktural

yang baru seperti <*main*>, <*section*>, <*article*>, <*header*>, <*footer*>, <*aside*>, <*nav*>, dan <*figure*>. Ada beberapa atribut baru yang diperkenalkan, beberapa elemen dan atribut telah dihapus, dan elemen-elemen lain seperti <a>, <cite> dan <menu> telah diubah atau didefinisikan ulang.

2.6 Canvas HTML5

Pada HTML5, terdapat elemen baru, yaitu *canvas* yang dapat digunakan untuk menggambarkan grafis pada halaman web. Dengan elemen ini, kita dapat melakukan *dynamic scriptable rendering* dari objek 2 dimensi dan gambar. Elemen ini adalah model prosedural *low-level* yang memperbarui sebuah *bitmap* dan tidak mempunyai *built-in scene graph.*

Canvas terdiri atas area yang dapat digambar, didefinisikan dengan atribut tinggi dan lebar. Kode Javascript dapat mengakses area tersebut melalui fungsi-fungsi menggambar yang mirip dengan API 2D pada umumnya, sehingga untuk dapat menghasilkan konten grafis secara dinamis. Beberapa orang mengharapkan elemen ini dapat digunakan untuk membangun graf, animasi, *game*, dan komposisi gambar.

(Halaman ini sengaja dikosongkan)

BAB 3
ANALISIS DAN PERANCANGAN

Bab ini membahas tahap analisis dan perancangan dari aplikasi yang akan dibangun. Hal yang akan dibahas meliputi analisis kebutuhan dan fitur yang diperlukan oleh aplikasi, serta perancangan perangkat lunak.

3.1 Analisis Perangkat Lunak

Tahap analisis dibagi menjadi beberapa bagian, antara lain deskripsi umum aplikasi, analisis kebutuhan, dan kasus penggunaan.

3.1.1 Deskripsi Umum Aplikasi

Aplikasi yang akan dibuat dalam tugas akhir ini adalah sebuah aplikasi berbasis website yang memanfaatkan fitur dari elemen *canvas* pada HTML5. Aplikasi akan menerima sebuah masukan dari pengguna yang merupakan data lokasi sekolah-sekolah. Dengan masukan tersebut, aplikasi akan memetakan sekolah-sekolah yang ada ke dalam diagram Voronoi dan menampilkannya ke browser.

3.1.2 Analisis Kebutuhan

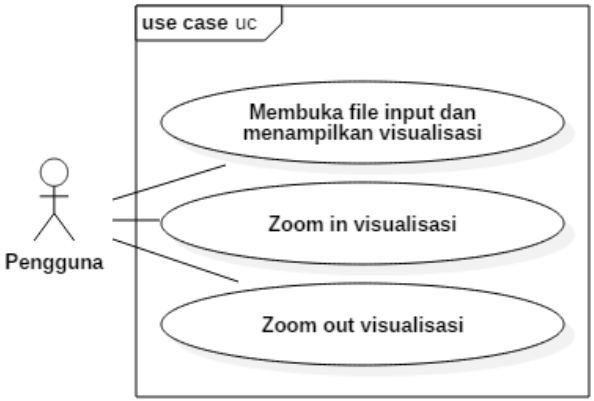
Pada aplikasi ini, terdapat beberapa kebutuhan fungsional. Kebutuhan fungsional dapat dilihat pada Tabel 3.1.

3.1.3 Kasus Penggunaan

Berdasarkan analisis pada kebutuhan fungsional, dibuatlah spesifikasi kasus penggunaan. Diagram kasus penggunaan dapat dilihat pada Gambar 3.1 dan spesifikasi kasus penggunaan dapat dilihat pada Tabel 3.2.

Tabel 3.1 Spesifikasi Kebutuhan Fungsional

Kode Kebutuhan	Kebutuhan Fungsional	Deskripsi
F-0001	Membuka file data sekolah	Pengguna dapat membuka file data sekolah sebagai masukan untuk aplikasi.
F-0002	Menampilkan visualisasi coverumlage sekolah	Aplikasi ini dapat membuat diagram voronoi berdasarkan data masukan dari pengguna dan menampilkannya bersama peta wilayah.
F-0003	*Zoom-in* dan *zoom-out* visualisasi	Pengguna dapat memperbesar dan memperkecil tampilan visualisasi.



Gambar 3.1. Diagram Kasus Penggunaan

Tabel 3.2 Spesifikasi Kasus Penggunaan

Kode Kasus Penggunaan	Nama
UC-0001	Membuka file input dan menampilkan visualisasi
UC-0002	Zoom-in visualisasi
UC-0003	Zoom-out visualisasi

Detail dari kasus-kasus penggunaan dari aplikasi yang dibangun dapat dilihat pada Tabel 3.3, Tabel 3.4, dan Tabel 3.5.

Tabel 3.3 Spesifikasi Kasus Membuka File Input dan Menampilkan Visualiasi

ID	UC-0001
Nama	Membuka file input dan menampilkan visualisasi
Deskripsi	Pengguna dapat membuka dokumen data sekolah dan aplikasi dapat menampilkan visualisasi.
Kondisi awal	Pengguna membuka aplikasi.
Aliran Kejadian Normal	1. Pengguna membuka file data sekolah dengan menggunakan bagan masukan yang ada pada aplikasi 2. Aplikasi membuat dan menampilkan visualisasi dari data yang dimasukkan pengguna.
Kondisi akhir	Terdapat tampilan visualisasi pemetaan sekolah.

Tabel 3.4 Spesifikasi Kasus Zoom-In Visualiasi

ID	UC-0002
Nama	Zoom-in visualisasi
Deskripsi	Pengguna dapat memperbesar kanvas, sehingga visualisasi tampak lebih besar.
Kondisi awal	Pengguna telah membuka dokumen data sekolah.
Aliran Kejadian Normal	1. Pengguna menekan tombol *zoom-in*
Kondisi akhir	Visualisasi tampak lebih besar.

Tabel 3.5 Spesifikasi Kasus Zoom-Out Visualiasi

ID	UC-0003
Nama	Zoom-out visualisasi
Deskripsi	Pengguna dapat memperkecil kanvas, sehingga visualisasi tampak lebih kecil.
Kondisi awal	Pengguna telah membuka dokumen data sekolah.
Aliran Kejadian Normal	1. Pengguna menekan tombol *zoom-out*
Kondisi akhir	Visualisasi tampak lebih kecil.

3.2 Perancangan Aplikasi

Tahap perancangan dibagi menjadi beberapa bagian antara lain perancangan arsitektur umum sistem, perancangan alur penggunaan, perancangan data, perancangan antarmuka, dan perancangan algoritma.

3.2.1 Perancangan Arsitektur Umum Sistem

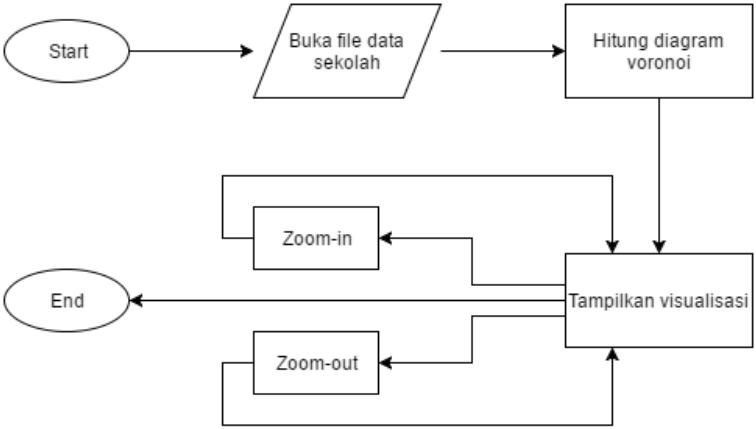
Arsitektur sistem pada aplikasi visualisasi *coverage* sekolah menggunakan diagram Voronoi dan HTML5 didukung oleh beberapa perangkat yaitu komputer, *mouse* dan *keyboard*.

3.2.2 Perancangan Alur Penggunaan

Pada perancangan ini dijelaskan alur penggunaan aplikasi oleh pengguna. Rancangan alur penggunaan ini dapat digunakan sebagai acuan untuk membuat antarmuka aplikasi. Rancangan alur penggunaan dapat dilihat pada Gambar 3.3.

3.2.3 Perancangan Data

Perancangan data berisi rancangan dari data wilayah dan sekolah yang akan digunakan sebagai masukan untuk aplikasi yang dibangun. Data akan disimpan dalam suatu dokumen teks dengan



Gambar 3.2 Rancangan Alur Penggunaan

format JSON. Dokumen data tersebut akan berisi objek wilayah/area dan daftar dari objek-objek sekolah. Adapun penjelasan atribut dari objek sekolah adalah sebagai berikut:

- "area" merupakan objek area.
    - "x1" dan "y1" merupakan koordinat paling kiri atas dari area.
    - "x2" dan "y2" merupakan koordinat paling kanan bawah dari area.
- "schools" merupakan daftar dari sekolah-sekolah yang ada pada area.
    - "name" merupakan nama dari sekolah.
    - "longitude" merupakan nilai koordinat sekolah pada garis bujur.
    - "latitude" merupakan nilai koordinat sekolah pada garis lintang.

Contoh dari data wilayah dan sekolah dapat dilihat pada Gambar 3.3.

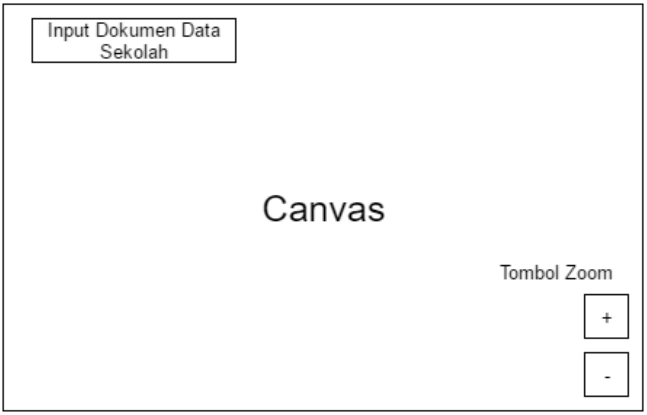
3.2.4 Perancangan Antarmuka

Perancangan antarmuka pengguna berisi rancangan tampilan dari aplikasi. Tampilan antarmuka dirancang dengan menggunakan HTML dan CSS. Rancangan antarmuka dapat dilihat pada Gambar 3.4. Elemen-elemen yang membangun antarmuka, antara lain:

a. *Input* dokumen data sekolah, merupakan *input* berjenis file untuk menerima dokumen data sekolah.
b. *Canvas*, merupakan tempat visualisasi akan ditampilkan.
c. Tombol *Zoom*, merupakan tombol yang akan digunakan pengguna untuk memperbesar dan memperkecil *canvas.*

```
{
  "area": {
    "x1": 112.760925,
    "y1": -7.262801,
    "x2": 112.800888,
    "y2": -7.313822,
  },
  "schools": [
    {
      "name": "SMA NEGERI 20 SURABAYA",
      "latitude": -7.307078,
      "longitude": 112.789178
    },
    {
      "name": "SMA 17 AGUSTUS 1945 SURABAYA",
      "latitude": -7.298123,
      "longitude": 112.768733
    },
    {
      "name": "SMA DR. SOETOMO SURABAYA",
      "latitude": -7.298868,
      "longitude": 112.764168
    }
  ]
}
```


Gambar 3.3 Contoh Data Wilayah dan Sekolah



Gambar 3.4 Rancangan Antarmuka

3.2.5 Perancangan Algoritma

Perancangan algoritma berisi rancangan algoritma yang digunakan aplikasi untuk menghitung dan membuat diagram Voronoi dari data masukan yang ada. Ada dua rancangan algoritma, antara lain algoritma diagram triangulation dan algoritma diagram Voronoi.

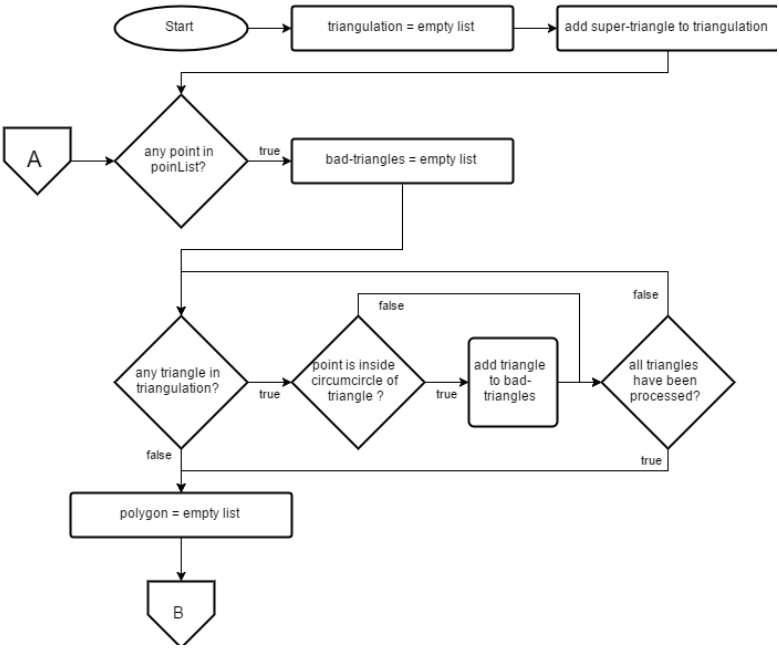
3.2.5.1 Rancangan Algoritma Diagram Triangulation

Pada aplikasi ini, algoritma yang digunakan untuk membentuk diagram *triangulation* adalah algoritma Bowyer-Watson. Rancangan dari algoritma diagram *triangulation* dapat dilihat pada Gambar 3.6 dan Gambar 3.7.

Pada aplikasi ini, tahapan terakhir ini tidak dilakukan untuk mempermudah proses pembuatan diagram Voronoi. *Pseudocode* dari fungsi untuk menghasilkan diagram *triangulation* ditunjukkan pada Gambar 3.5.

```
GenerateTriangulation(pointList, width, height)
triangulation := empty list
super-triangle := createSuperTriangle(width, height)
add super-triangle to triangulation
for each point in pointList do
  bad-triangles := empty list
  for each triangle in triangulation do
    if point is inside circumcircle of triangle
      add triangle to bad-triangles
  polygon := empty list
  for each triangle in bad-triangles do
    for each edge in triangle do
      if edge is not shared by any other triangles in bad-triangles
        add edge to polygon
  for each triangle in bad-triangles do
    remove triangle from triangulation
  for each edge in polygon do
    new-triangle := a new triangle from edge to point
return triangulation
```

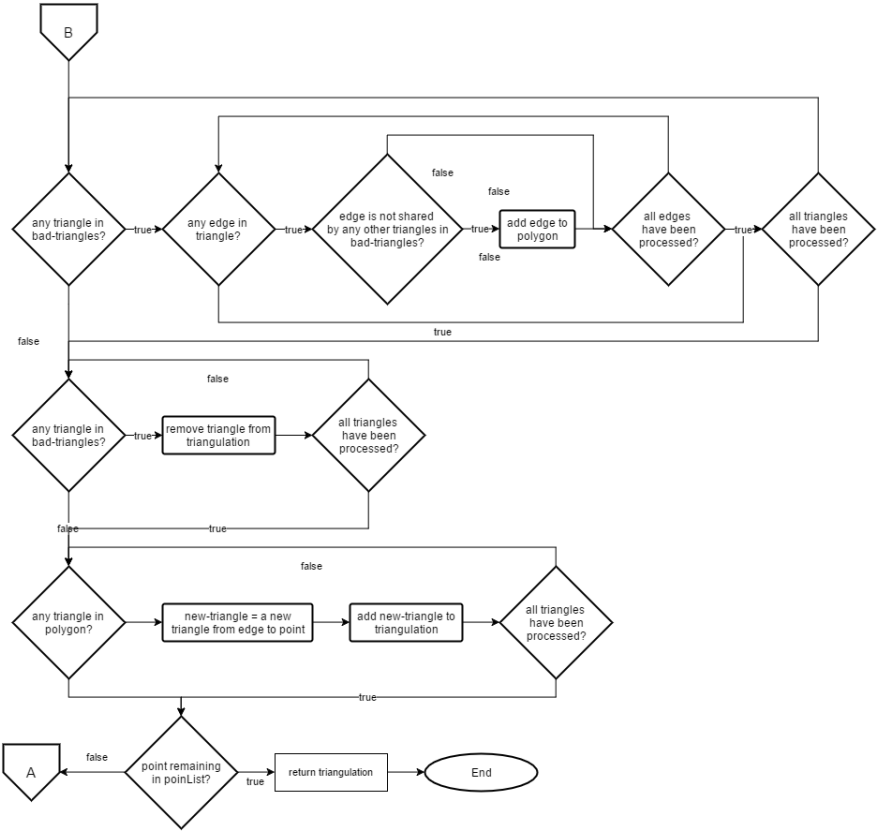
Gambar 3.5 Pseudocode Algoritma Diagram Triangulation



Gambar 3.6 Rancangan Algoritma Diagram Triangulation (1)

3.2.5.2 Rancangan Algoritma Diagram Voronoi

Untuk mengubah diagram *triangulation* menjadi diagram Voronoi, aplikasi akan menghubungkan kedua titik pusat dari *circumcirle* dari kedua segitiga yang memiliki sisi yang sama. *Pseudocode* dari fungsi untuk menghasilkan diagram Voronoi ditunjukkan pada Gambar 3.8. Rancangan dari algoritma diagram *triangulation* dapat dilihat pada Gambar 3.9.



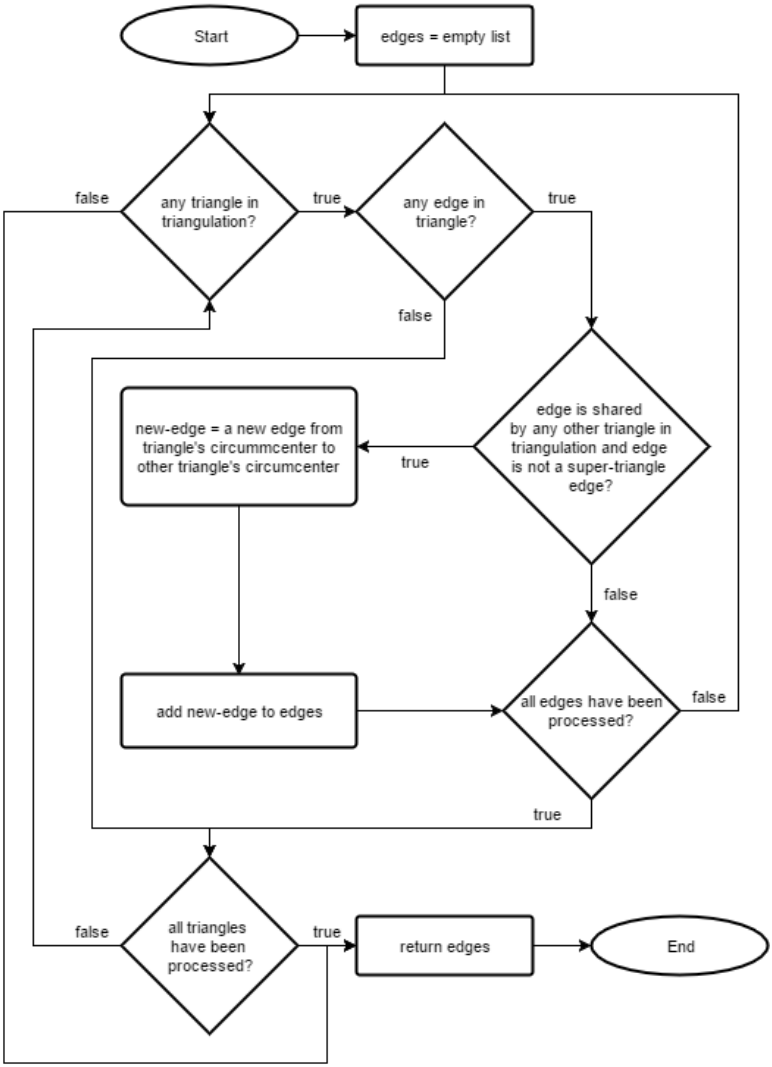
Gambar 3.7 Rancangan Algoritma Diagram Triangulation (2)

```
GenerateVoronoi(pointList, width, height)
edges := empty list
super-triangle := CreateSuperTriangle(width,
height);
triangulation := GenerateTriangulation(pointList,
width, height)
for each triangle in triangulation do
  for each edge in triangle do
    if edge is shared by any other triangle in
triangulation and edge is not a super-triangle
edge
      new-edge := a new edge from triangle's
circummcenter to other triangle's circumcenter
      add new-edge to edges
return edges
```

Gambar 3.8 Pseudocode Algoritma Diagram Voronoi

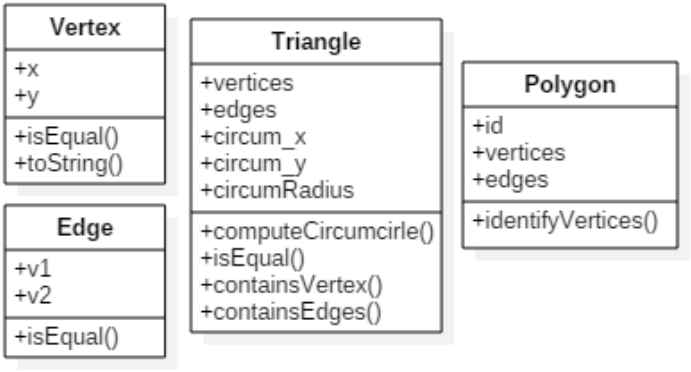
3.2.5.2 Rancangan Tambahan

Untuk mendukung kedua fungsi algoritma sebelumnya, perlu dibentuk empat buah kelas objek, antara lain kelas objek *Vertex*, *Edge*, *Triangle*, dan *Polygon.* Masing-masing dari kelas objek tersebut mewakili data titik, sisi/garis, segitiga, dan poligon.



Gambar 3.9 Rancangan Algoritma Diagram Voronoi

Diagram kelas dari keempat kelas objek tersebut dapat dilihat pada Gambar 3.7.



Gambar 3.10 Rancangan Diagram Kelas Vertex, Edge, Triangle, dan Polygon

BAB 4
IMPLEMENTASI

Pada bab ini dibahas mengenai implementasi yang dilakukan berdasarkan rancangan yang telah dijabarkan pada bab sebelumnya.

4.1 Lingkungan Implementasi

Lingkungan implementasi merupakan lingkungan dimana aplikasi akan dibangun. Lingkungan implementasi dibagi menjadi dua, yaitu lingkungan implementasi berupa perangkat keras dan lingkungan implementasi berupa perangkat lunak.

4.1.1 Perangkat Keras

Perangkat keras yang digunakan dalam pengembangan aplikasi ini adalah komputer dengan spesifikasi sebagai berikut:

- Tipe : Lenovo IdeaPad Z410
- Prosesor : Intel(R) Core(TM) i5-4200M CPU@2.50 GHz
- Memori : 8 GB

4.1.2 Perangkat Lunak

Perangkat lunak yang digunakan dalam pengembangan aplikasi ini adalah sebagai berikut:

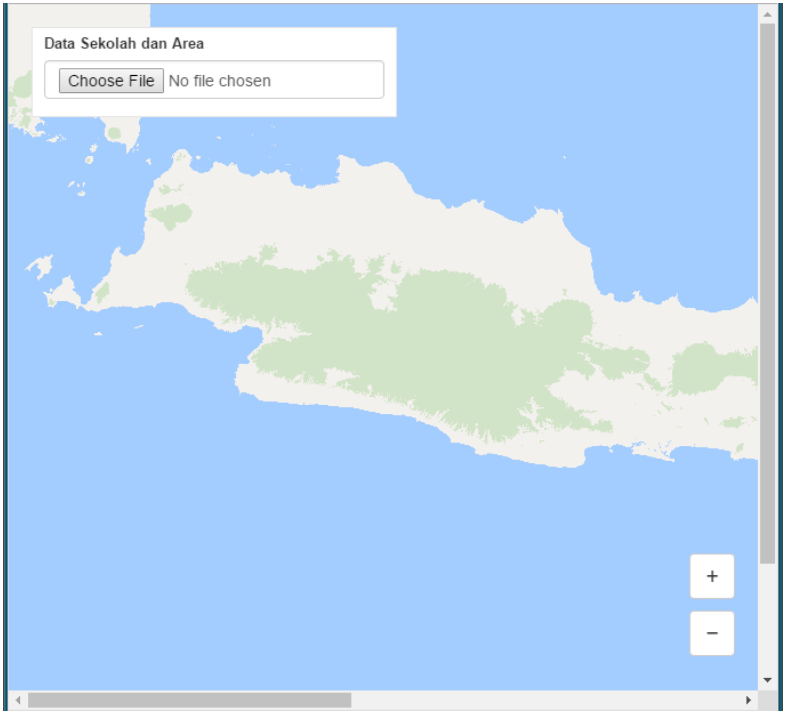
- Sistem Operasi : Sistem Operasi Ubuntu 16.04
- *Text Editor* : Sublime Text 3
- *Browser* : Google Chrome versi 55

4.2 Implementasi

Tahap implementasi yang dilakukan dibagi menjadi dua bagian, antara lain implementasi antarmuka dan implementasi algoritma.

4.2.1 Implementasi Antarmuka

Pada tahap ini dijelaskan implementasi dari rancangan desain antarmuka yang telah dirancang pada Bab 3. Pada halaman antarmuka, digunakan empat elemen utama antara lain: satu elemen *input* bertipe *file* untuk menerima masukan data sekolah, satu elemen *canvas* untuk menampilkan visualisasi diagram, dan dua buah elemen *button* untuk melakukan proses *zoom*. Antarmuka yang telah diimplementasi dapat dilihat pada Gambar 4.1.



Gambar 4.1 Implementasi Antarmuka

Pada implementasi antarmuka, diimplementasikan juga beberapa fungsi yang digunakan untuk menampilkan visualisasi pada elemen *canvas*. Implementasi fungsi-fungsi tersebut dapat dilihat pada kode-kode sumber berikut ini.

```
function drawVertices(vertices) {
  ctx.fillStyle = "#F44336";
  for (i in vertices) {
    ctx.beginPath();
    ctx.arc(
      geoToMapX(vertices[i].x),
      geoToMapY(vertices[i].y),
      lineWidth * 2, 0, Math.PI * 2
    );
    ctx.closePath();
    ctx.fill();
  }
}
```

Kode Sumber 4.1 Implementasi Fungsi drawVertices

Fungsi *drawVertices* digunakan untuk menggambarkan titik-titik pada *canvas*. Titik-titik tersebut menunjukkan letak dari sekolah-sekolah pada peta.

```
function drawPolygons(polygons) {
  var coverageLevel = [];
  coverageLevel[0] = Math.max.apply(Math,
coverage);
  for (var i = 1; i < 9; i++) {
```

```
    coverageLevel[i] = coverageLevel[0] / 8 *
(8 - i);
  }

  for (var i = 0; i < 8; i++) {
    $("#coverage-"+i).html((coverageLevel[i]
* 100).toFixed(2) +
      " &#8804; d &#8804; " +
(coverageLevel[i-1] * 100).toFixed(2));
  }
  $("#coverage-8").html("d &#8804; " +
(coverageLevel[7] * 100).toFixed(2));
  $("#legenda").show();

  for (var i in polygons) {
    ctx.beginPath();
    ctx.moveTo(
      geoToMapX(polygons[i].vertices[0].x),
      geoToMapY(polygons[i].vertices[0].y)
    );
    for (var j = 1; j <
polygons[i].vertices.length; j++) {
      ctx.lineTo(
        geoToMapX(polygons[i].vertices[j].x),
        geoToMapY(polygons[i].vertices[j].y)
      );
    }

    if (coverage[i] > coverageLevel[0] / 8 *
7) {
```

```
      ctx.fillStyle = "rgba(183, 28, 28,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8 * 6) {
      ctx.fillStyle = "rgba(156, 39, 176,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8 * 5) {
      ctx.fillStyle = "rgba(121, 85, 72,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8 * 4) {
      ctx.fillStyle = "rgba(255, 152, 0,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8 * 3) {
      ctx.fillStyle = "rgba(33, 150, 243,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8 * 2) {
      ctx.fillStyle = "rgba(76, 175, 80,
0.5)";
    }
    else if (coverage[i] > coverageLevel[0] /
8) {
      ctx.fillStyle = "rgba(255, 235, 59,
0.5)";
```

```
    }
    else {
      ctx.fillStyle = "rgba(200, 200, 200,
0.5)";
    }
    ctx.fill();

    ctx.strokeStyle = "#212121";
    ctx.lineWidth = lineWidth;
    ctx.lineCap = 'round';
    ctx.stroke();
  }
}
```

Kode Sumber 4.2 Implementasi Fungsi drawPolygons

Fungsi *drawPolygons* digunakan untuk menggambarkan poligon-poligon pada *canvas*. Poligon-poligon tersebut menunjukkan wilayah *coverage* dari masing-masing sekolah.

```
function drawTransformed() {
  if (scale < 3) {
    canvas.width = scale * viewWidth;
    canvas.height = scale * viewHeight;
  }

  ctx.clearRect(0, 0, canvas.width,
canvas.height);
  ctx.save();
  ctx.scale(scale, scale);
```

```
  if (scale < 3) {
    ctx.drawImage(map, 0, 0);
  }
  else {
    if (sx + viewWidth > map.width) {
      sx = map.width - viewWidth;
      sxGeo = 105;
      sxGeo = mapToGeoX(sx);
    }
    if (sy + viewHeight > map.height) {
      sy = map.height - viewHeight;
      sxGeo = -5;
      syGeo = mapToGeoY(sy);
    }
    ctx.drawImage(map, sx, sy,
      viewWidth, viewHeight, 0, 0,
      viewWidth, viewHeight
    );
  }

  if (polygons.length) {
    drawPolygons(polygons);
    drawVertices(vertices);
  }

  ctx.restore();
}
```

Kode Sumber 4.3 Implementasi Fungsi drawTransformed

Fungsi drawTransformed merupakan fungsi utama untuk menggambarkan dan menampilkan visualisasi pada *canvas*. Fungsi drawTransformed menggambarkan peta, titik-titik sekolah, dan garis-garis batas dengan ukuran yang didasarkan pada *scaleLevel*.

4.2.2 Implementasi Alur Penggunaan

Pada tahap ini dijelaskan implementasi dari rancangan alur penggunaan yang telah dirancang pada Bab III.

4.2.2 Implementasi Proses Membuka Data Sekolah

Implementasi proses membuka data sekolah dapat dilihat pada Kode Sumber 4.4.

```
fr.onload = function(e) {
  $("#loader").show(100, function(){
    vertices.length = 0;
    polygons.length = 0;

    var data = JSON.parse(fr.result);
    schools = data.schools;

    for (var i = 0; i < schools.length; i++)
{
      var vertex = new Vertex(
        schools[i].longitude,
        schools[i].latitude
      );
      vertices.push(vertex);
    }
```

```
    var area = data.area;
    area.height *= -1;
    polygons = generateVoronoi(vertices,
area);

    for (polygon of polygons) {
      polygon.identifyVertices(area);
    }
    coverage = doSampling(polygons, area);

  drawTransformed();
    $("#loader").hide(100);
  });
};
```

Kode Sumber 4.4 Implementasi Proses Membuka Data Sekolah

Pada tahap implementasi ini, disematkan sebuah *event on-change* pada *input* data sekolah. Ketika pengguna membuka data sekolah, aplikasi akan membuka data sekolah secara local dengan menggunakan kelas *FileReader*. Setelah dibaca, data sekolah yang berformat JSON akan di-*parsing* dan digunakan sebagai parameter untuk mendapatkan diagram Voronoi.

4.2.2 Implementasi Zoom

Implementasi *zoom-in* dan *zoon-out* dapat dilihat pada Kode Sumber 4.5 dan Kode Sumber 4.6.

```
$("#zoom-in-btn").click(function() {
  if (scale < 2) {
    centerX = window.scrollX +
window.innerWidth  / 2;
    centerY = window.scrollY +
window.innerHeight / 2;

    if (scale > 1.5) {
      sx = window.scrollX / scale;
      sy = window.scrollY / scale;
      sxGeo = mapToGeoX(sx);
      syGeo = mapToGeoY(sy);
      areaWidth /= 2;
      areaHeight /= 2;
      viewWidth /= 2;
      viewHeight /= 2;
    }
    scale *= 2;
    lineWidth /= 2;
    drawTransformed();
    window.scrollTo(
      centerX * 2 - window.innerWidth  / 2 -
sx * scale,
      centerY * 2 - window.innerHeight / 2 -
sy * scale
    );
  }
});
```

Kode Sumber 4.5 Implementasi Zoom-In

```
$("#zoom-out-btn").click(function() {
  if (scale > 0.2) {
    centerX = window.scrollX +
window.innerWidth  / 2;
    centerY = window.scrollY +
window.innerHeight / 2;

    if (scale > 3) {
      sxGeo = 105;
      syGeo = -5;
      areaWidth *= 2;
      areaHeight *= 2;
      viewWidth *= 2;
      viewHeight *= 2;
    }

    scale /= 2;
    lineWidth *= 2;    2

    drawTransformed();

    window.scrollTo(
      centerX / 2 - window.innerWidth  / 2 +
sx * scale,
      centerY / 2 - window.innerHeight / 2 +
sy * scale
    );

    if (scale > 3) {
      sx = 0;
```

```
        sy = 0;
      }
    }
});
```

Kode Sumber 4.6 Implementasi Zoom-Out

4.2.3 Implementasi Algoritma

Pada tahap ini dijelaskan implementasi dari rancangan algoritma yang telah dirancang pada Bab III.

4.2.3.1 Implementasi Algoritma Triangulation

Implementasi fungsi untuk menghasilkan diagram *triangulation* dapat dilihat pada Kode Sumber 4.7.

```
function generateTriangulation(vertices,
area) {
  var triangulation = [];

  var super_triangle =
createSuperTriangle(area);
  triangulation.push(super_triangle);

  for (var i in vertices) {
    var vertex = vertices[i];

    var bad_triangles = [];

    for (var j in triangulation) {
      var triangle = triangulation[j];
```

```
      if (triangle.circumcircleContains(vertex)) {
        bad_triangles.push(triangle);
      }
    }

    polygon = [];
    for (var j in bad_triangles) {
      var triangle1 = bad_triangles[j];

      for (var k = 0; k < 3; k++) {
        var edge = triangle1.edges[k];

        var nice_edge = true;
        for (var l in bad_triangles) {
          var triangle2 = bad_triangles[l];

          if (triangle1 !== triangle2 && triangle2.containsEdge(edge)) {
            nice_edge = false;
            break;
          }
        }
        if (nice_edge) {
          polygon.push(edge);
        }
      }
    }
```

```
    triangulation =
removeTriangles(bad_triangles,
triangulation);

    for (var j in polygon) {
      var edge = polygon[j];

      var v1 = edge.v1;
      var v2 = edge.v2;
      var v3 = vertex;

      var e1 = edge;
      var e2 = new Edge(v2, v3);
      var e3 = new Edge(v3, v1);

      var new_triangle = new Triangle([v1,
v2, v3], [e1, e2, e3]);
      triangulation.push(new_triangle);
    }
  }

  return triangulation;
}
```

Kode Sumber 4.7 Implementasi Algoritma Diagram Triangulation

4.2.3.2 Implementasi Algoritma Diagram Voronoi

Implementasi fungsi untuk menghasilkan diagram Voronoi dapat dilihat pada Kode Sumber 4.8.

```
function generateVoronoi(vertices, area) {
  var voronoi = [];

  var super_triangle =
createSuperTriangle(area);

  var triangulation =
generateTriangulation(vertices, area);

  for (var i in triangulation) {
    var triangle = triangulation[i];

    var edges = triangle.edges;
    for (var j in edges) {
      var edge = edges[j];

      for (var k = +i + 1; k <
triangulation.length; k++) {
        var t = triangulation[k];

        if (t.containsEdge(edge) && !(
super_triangle.containsVertex(edge.v1) ||
super_triangle.containsVertex(edge.v2) ))
        {
          var newEdge = new Edge(
            new Vertex(triangle.circum_x,
triangle.circum_y),
```

```
            new Vertex(t.circum_x,
t.circum_y)
          );
          newEdge =
validateEdgeByArea(newEdge, area);

          if (newEdge.v1.isEqual(newEdge.v2))
{
            break;
          }

          var v1Flag = false;
          var v2Flag = false;
          for (l in voronoi) {
            var polygon = voronoi[l];

            if (polygon.id.isEqual(edge.v1))
{
              polygon.edges.push(newEdge);
              v1Flag = true;
            }
            if (polygon.id.isEqual(edge.v2))
{
              polygon.edges.push(newEdge);
              v2Flag = true;
            }
          }
          if (!v1Flag) {
            var newPolygon = new
Polygon(edge.v1);
```

```
            newPolygon.edges.push(newEdge);
            voronoi.push(newPolygon);
          }
          if (!v2Flag) {
            var newPolygon = new
Polygon(edge.v2);
            newPolygon.edges.push(newEdge);
            voronoi.push(newPolygon);
          }
          break;
        }
      }
    }
  }

  return voronoi;
}
```

Kode Sumber 4.8 Implementasi Algoritma Diagram Voronoi

4.2.3.2 Implementasi Tambahan

Sesuai dengan rancangan algoritma pada Bab III, diperlukan empat buah kelas, antara lain kelas *Vertex*, *Edge*, *Triangle* dan *Polygon*, untuk mendukung algoritma yang ada. Implementasi dari ketiga kelas tersebut dapat dilihat pada Kode Sumber 4.9, Kode Sumber 4.10, Kode Sumber 4.11, dan Kode Sumber 4.12.

```
function Vertex(x, y) {
  this.x = x;
  this.y = y;

  this.isEqual = function(v) {
    return (this.x == v.x && this.y == v.y);
  }

  this.toString = function() {
    return this.x.toString() + "," +
this.y.toString();
  }
}
```

Kode Sumber 4.9 Implementasi Kelas Vertex

```
function Edge(v1, v2) {
  this.v1 = v1;
  this.v2 = v2;

  this.isEqual = function(e) {
    return (this.v1.isEqual(e.v1) &&
this.v2.isEqual(e.v2)) ||
           (this.v1.isEqual(e.v2) &&
this.v2.isEqual(e.v1));
  }
}
```

Kode Sumber 4.10 Implementasi Kelas Edge

```
function Triangle(vertices, edges) {
  this.vertices = vertices;
  this.edges = edges;

  this.computeCircumcirle = function() {
    var v1 = this.vertices[0];
    var v2 = this.vertices[1];
    var v3 = this.vertices[2];

    var ab = Math.pow(v1.x, 2) +
Math.pow(v1.y, 2);
    var cd = Math.pow(v2.x, 2) +
Math.pow(v2.y, 2);
    var ef = Math.pow(v3.x, 2) +
Math.pow(v3.y, 2);

    this.circum_x =   (ab * (v3.y - v2.y) +
cd * (v1.y - v3.y) + ef * (v2.y - v1.y)) /
    (v1.x * (v3.y - v2.y) + v2.x * (v1.y -
v3.y) + v3.x * (v2.y - v1.y)) / 2;
    this.circum_y =   (ab * (v3.x - v2.x) +
cd * (v1.x - v3.x) + ef * (v2.x - v1.x)) /
    (v1.y * (v3.x - v2.x) + v2.y * (v1.x -
v3.x) + v3.y * (v2.x - v1.x)) / 2;
    return Math.sqrt(Math.pow(v1.x -
this.circum_x, 2) + Math.pow(v1.y -
this.circum_y, 2));
  }
  this.circumRadius =
this.computeCircumcirle();

```

```
  this.circumcircleContains = function(v) {
    var dist = Math.sqrt(Math.pow(v.x -
this.circum_x, 2) + Math.pow(v.y -
this.circum_y, 2));
    return dist <= this.circumRadius;
  }

  this.isEqual = function(t) {
    var v1 = this.vertices;
    var v2 = t.vertices;

    return v1[0].isEqual(v2[0]) &&
v1[1].isEqual(v2[1]) && v1[2].isEqual(v2[2])
||
           v1[0].isEqual(v2[0]) &&
v1[1].isEqual(v2[2]) && v1[2].isEqual(v2[1])
||
           v1[0].isEqual(v2[1]) &&
v1[1].isEqual(v2[0]) && v1[2].isEqual(v2[2])
||
           v1[0].isEqual(v2[1]) &&
v1[1].isEqual(v2[2]) && v1[2].isEqual(v2[0])
||
           v1[0].isEqual(v2[2]) &&
v1[1].isEqual(v2[0]) && v1[2].isEqual(v2[1])
||
           v1[0].isEqual(v2[2]) &&
v1[1].isEqual(v2[1]) && v1[2].isEqual(v2[0]);
  }

  this.containsVertex = function(v) {
```

```
    for (var i in this.vertices) {
      var vertex = this.vertices[i];

      if (v.isEqual(vertex)) {
        return true;
      }
    }
    return false;
  }

  this.containsEdge = function(e) {
    for (var i in this.edges) {
      var edge = this.edges[i];

      if (edge.isEqual(e)) {
        return true;
      }
    }
    return false;
  }
}
```

Kode Sumber 4.11 Implementasi Kelas Triangle

```
function Polygon(vertex) {
  this.id = vertex;
  this.vertices = [];
  this.edges = [];
```

```
  this.identifyVertices = function(area) {
    this.vertices.push(
      new Vertex(
        this.edges[0].v1.x,
        this.edges[0].v1.y
      )
    );
    this.vertices.push(
      new Vertex(
        this.edges[0].v2.x,
        this.edges[0].v2.y
      )
    );

    var used = [];
    var visit = new Set();

    for (var j = 0; j < this.edges.length;
j++) {
      var now = this.edges[j].v1;
      if (visit.has(now.toString())) {
        visit.delete(now.toString());
      }
      else {
        visit.add(now.toString());
      }
      var now = this.edges[j].v2;
      if (visit.has(now.toString())) {
```

```
        visit.delete(now.toString());
      }
      else {
        visit.add(now.toString());
      }
    }

    if (visit.size) {
      var arr = [];
      for (var j of visit) {
        var coor = j.split(",");
        arr.push(new Vertex(+coor[0],
+coor[1]));
      }

      if (arr[0].x == area.x1 && arr[1].y ==
area.y1) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x1, area.y1)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x1, area.y1)));
      }
      else if (arr[0].x == area.x1 &&
arr[1].y == area.y2) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x1, area.y2)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x1, area.y2)));
      }
      else if (arr[0].x == area.x2 &&
arr[1].y == area.y1) {
```

```
        this.edges.push(new Edge(arr[0], new
Vertex(area.x2, area.y1)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x2, area.y1)));
      }
      else if (arr[0].x == area.x2 &&
arr[1].y == area.y2) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x2, area.y2)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x2, area.y2)));
      }
      else if (arr[1].x == area.x1 &&
arr[0].y == area.y1) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x1, area.y1)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x1, area.y1)));
      }
      else if (arr[1].x == area.x1 &&
arr[0].y == area.y2) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x1, area.y2)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x1, area.y2)));
      }
      else if (arr[1].x == area.x2 &&
arr[0].y == area.y1) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x2, area.y1)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x2, area.y1)));
```

```
      }
      else if (arr[1].x == area.x2 &&
arr[0].y == area.y2) {
        this.edges.push(new Edge(arr[0], new
Vertex(area.x2, area.y2)));
        this.edges.push(new Edge(arr[1], new
Vertex(area.x2, area.y2)));
      }
      else {
        this.edges.push(new Edge(arr[0],
arr[1]));
      }
    }

    used[0] = true;
    var now = this.edges[0].v2;

    for (var j = 1; j < this.edges.length;
j++) {
      for (var k = 1; k < this.edges.length;
k++) {
        if (used[k]) {
          continue;
        }
        if (this.edges[k].v1.isEqual(now)) {
          now = this.edges[k].v2;
          this.vertices.push(
            new Vertex(now.x, now.y)
          );
          used[k] = true;
```

```
            break;
          }
         else if
(this.edges[k].v2.isEqual(now)) {
            now = this.edges[k].v1;
            this.vertices.push(
              new Vertex(now.x, now.y)
            );
            used[k] = true;
            break;
          }
        }
      }
    }
}
```

Kode Sumber 4.12 Implementasi Kelas Polygon

Selain mengimplementasikan ketiga kelas di atas, terdapat tiga fungsi tambahan yang digunakan oleh kedua algoritma utama. Ketiga fungsi tersebut diimplentasikan agar aplikasi menjadi lebih modular. Ketiga fungsi tersebut, antara lain fungsi untuk membuat segitiga super, fungsi untuk menghapus segitiga dari *list*, dan fungsi untuk validasi *edge* berdasarkan area sehingga letak *edge* hanya di dalam area. Implementasi dari ketiga fungsi tersebut dapat dilihat pada Kode Sumber 4.13, Kode Sumber 4.14, dan Kode Sumber 4.15.

```
function createSuperTriangle(area) {
  var v1 = new Vertex(area.x1 - area.width,
area.y1 - area.height);
  var v2 = new Vertex(area.x2 + 2 *
area.width, area.y1 - area.height);
  var v3 = new Vertex(area.x1 - area.width,
area.y2 + 2 * area.height);

  var e1 = new Edge(v1, v2);
  var e2 = new Edge(v1, v3);
  var e3 = new Edge(v2, v3);

  return new Triangle([v1, v2, v3], [e1, e2,
e3]);
}
```

Kode Sumber 4.13 Implementasi Fungsi Membuat Segitiga Super

```
function removeTriangles(triangles,
triangulation) {
  var new_triangulation = [];
  for (i in triangulation) {
    var triangle1 = triangulation[i];

    var goodTriangle = true;
    for (j in triangles) {
      var triangle2 = triangles[j];

      if(triangle1.isEqual(triangle2)) {
        goodTriangle = false;
```

```
          break;
        }
      }
      if(goodTriangle) {
        new_triangulation.push(triangle1);
      }
    }
    return new_triangulation;
}
```

Kode Sumber 4.14 Implementasi Fungsi Menghapus Segitiga dari List

```
function validateEdgeByArea(edge, area) {
  var x1 = edge.v1.x;
  var y1 = edge.v1.y;
  var x2 = edge.v2.x;
  var y2 = edge.v2.y;

  if (edge.v1.x < area.x1) {
    edge.v1.x = area.x1;
    edge.v1.y = (area.x1 - x1) * (y2 - y1) /
(x2 - x1) + y1;
  }
  else if (edge.v1.x > area.x2) {
    edge.v1.x = area.x2;
    edge.v1.y = (area.x2 - x1) * (y2 - y1) /
(x2 - x1) + y1;
  }
```

```
  if (edge.v1.y > area.y1) {
    edge.v1.x = (area.y1 - y1) * (x2 - x1) /
(y2 - y1) + x1;
    edge.v1.y = area.y1;
  }
  else if (edge.v1.y < area.y2) {
    edge.v1.x = (area.y2 - y1) * (x2 - x1) /
(y2 - y1) + x1;
    edge.v1.y = area.y2;
  }

  if (edge.v2.x < area.x1) {
    edge.v2.x = area.x1;
    edge.v2.y = (area.x1 - x1) * (y2 - y1) /
(x2 - x1) + y1;
  }
  else if (edge.v2.x > area.x2) {
    edge.v2.x = area.x2;
    edge.v2.y = (area.x2 - x1) * (y2 - y1) /
(x2 - x1) + y1;
  }
  if (edge.v2.y > area.y1) {
    edge.v2.x = (area.y1 - y1) * (x2 - x1) /
(y2 - y1) + x1;
    edge.v2.y = area.y1;
  }
  else if (edge.v2.y < area.y2) {
    edge.v2.x = (area.y2 - y1) * (x2 - x1) /
(y2 - y1) + x1;
    edge.v2.y = area.y2;
```

```
  }

  return edge;
}
```

Kode Sumber 4.15 Implementasi Fungsi Validasi Edge Berdasarkan Area

Untuk mendapatkan nilai *coverage* dari setiap area Voronoi, dilakukan non-uniform distribution sampling. Implementasi fungsi untuk melakukan sampling dapat dilihat pada Kode Sumber 4.16.

```
function doSampling(polygons, area) {
  var samples = [];
  var n = polygons.length * 500;

  var samplesX = [];
  while (samplesX.length < n){
    var randomnumber = Math.random() *
(area.x2 - area.x1) + area.x1;
    if (samplesX.indexOf(randomnumber) > -1)
{
      continue;
    }
    samplesX[samplesX.length] = randomnumber;
  }

  var samplesY = [];
  while (samplesY.length < n){
```

```
    var randomnumber = Math.random() *
(area.y1 - area.y2) + area.y2;
    if (samplesY.indexOf(randomnumber) > -1)
{
      continue;
    }
    samplesY[samplesY.length] = randomnumber;
  }

  var sum = [];
  var sumLength = [];
  for (var i = 0; i < polygons.length; i++) {
    sum[i] = 0;
    sumLength[i] = 0;
  }

  for (var i = 0; i < n; i++) {
    var min = Math.sqrt(
      Math.pow(polygons[0].id.x -
samplesX[i], 2) +
      Math.pow(polygons[0].id.y -
samplesY[i], 2)
    );
    var index = 0;
    for (var j in polygons) {
      var distance = Math.sqrt(
        Math.pow(polygons[j].id.x -
samplesX[i], 2) +
        Math.pow(polygons[j].id.y -
samplesY[i], 2)
```

```
        );
        if (distance < min) {
          min = distance;
          index = j;
        }
      }
      sum[index] += min;
      sumLength[index]++;
    }

    var result = [];
    for (var i = 0; i < polygons.length; i++) {
      if (sumLength[i]) {
        result[i] = sum[i] / sumLength[i];
      }
      else {
        result[i] = 0;
      }
    }

    return result;
}
```

Kode Sumber 4.16 Implementasi Fungsi Sampling

BAB 5
PENGUJIAN DAN EVALUASI

Pada bab ini dijelaskan tentang uji coba dan evaluasi dari implementasi yang telah dilakukan.

5.1 Lingkungan Pengujian

Lingkungan uji coba yang digunakan adalah sebuah komputer dengan spesifikasi sebagai berikut:

1. Perangkat Keras
   a. Tipe : Lenovo IdeaPad Z410
   b. Prosesor : Intel(R) Core(TM) i5-4200M CPU @2.50GHz
   c. Memori : 8 GB
2. Perangkat Lunak
   a. Sistem Operasi : Sistem Operasi Ubuntu 16.04
   b. *Text Editor* : Sublime Text 3
   c. *Browser* : Google Chrome versi 55

5.2 Pengujian Aplikasi

Pengujian aplikasi dilakukan untuk mengetahui kesesuaian keluaran dari tiap tahap atau langkah penggunaan fitur terhadap skenario yang dipersiapkan. Berikut ini penjabaran skenario dan hasil uji coba yang dilakukan terhadap aplikasi yang dibangun.

5.2.1 Pengujian Fungsionalitas

Pengujian fungsionalitas dilakukan dengan menyiapkan beberapa skenario pengujian sebagai tolok ukur keberhasilan pengujian.

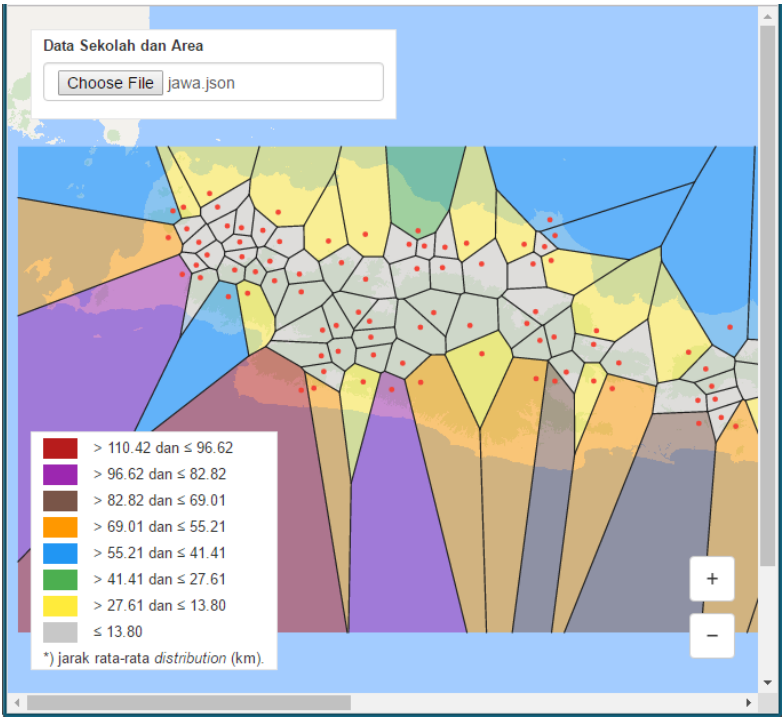
5.2.1.1 Pengujian Membuka Data Sekolah dan Menampilkan Visualisasi

Pengujian membuka data sekolah dan menampilkan visualisasi dilakukan untuk mengetahui keberhasilan aplikasi

dalam membuka dokumen berisi data sekolah yang dipilih oleh pengguna dan kemudian menampilkan visualisasi sesuai dengan data masukan. Hasil pengujian dapat dilihat pada Tabel 5.1 dan Gambar 5.1.

Tabel 5.1 Uji Coba Membuka Data Sekolah dan Menampilkan Visualisasi

ID	UJ-01
Nama	Uji Coba Membuka Data Sekolah dan Menampilkan Visualisasi
Tujuan Uji Coba	Pengguna dapat membuka dokumen data sekolah dan aplikasi dapat menampilkan visualisasi.
Kondisi awal	Pengguna membuka aplikasi.
Skenario	Pengguna membuka dokumen data sekolah dengan menggunakan bagan masukan.
Keluaran yang diharapkan	Aplikasi membaca dokumen data sekolah, menghitung diagram Voronoi, dan menampilkan visualisasi.
Hasil uji coba	Berhasil dengan catatan.
Kondisi akhir	Terdapat tampilan visualisasi pemetaan sekolah.
Kendala	Visualisasi belum mampu menangani perbatasan area seperti perbatasan kota atau perbatasan pulau dan laut.



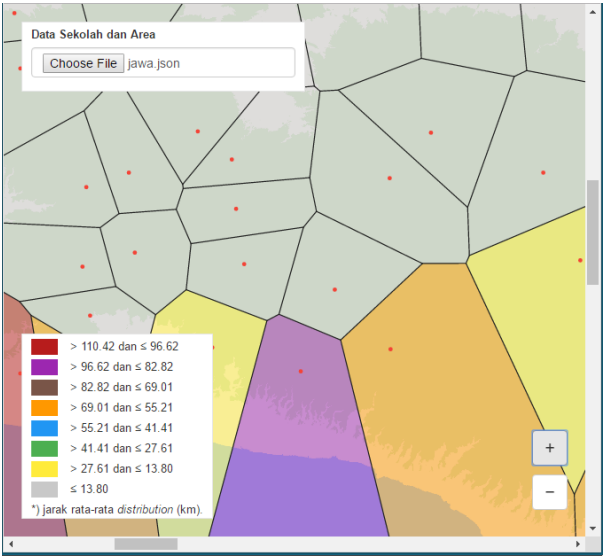
Gambar 5.1 Hasil Uji Coba Membuka Data Sekolah dan Menampilkan Visualisasi

5.2.1.2 Pengujian Zoom-In

Pengujian *zoom-in* dilakukan untuk mengetahui keberhasilan aplikasi dalam memperbesar kanvas sehingga kanvas menjadi dua kali lebih besar dari sebelumnya. Hasil pengujian dapat dilihat pada Tabel 5.2 dan Gambar 5.2.

Tabel 5.2 Uji Coba Zoom-In

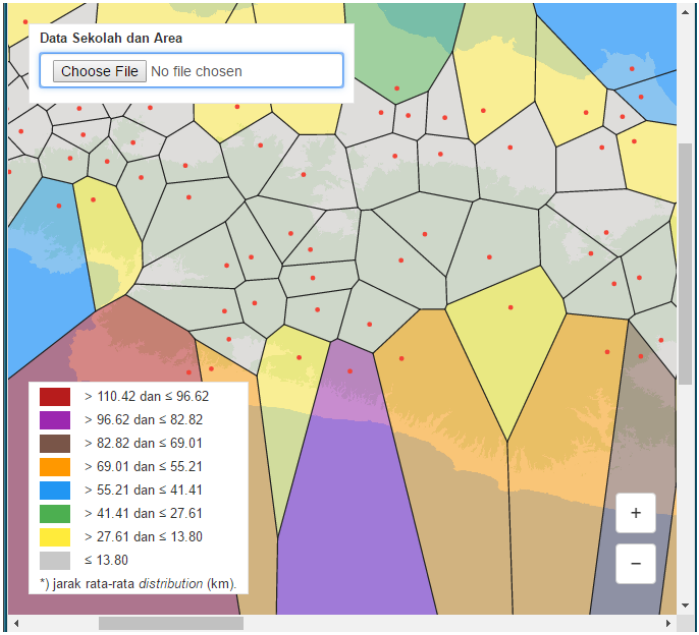
ID	UJ- 02
Nama	Uji Coba Zoom-In
Tujuan Uji Coba	Pengguna dapat memperbesar kanvas, sehingga visualisasi tampak lebih besar.
Kondisi awal	Pengguna telah membuka dokumen data sekolah.
Skenario	Pengguna menekan tombol *zoom-in*
Keluaran yang diharapkan	Kanvas menjadi dua kali lebih besar dari sebelumnya.
Hasil uji coba	Berhasil dengan catatan.
Kondisi akhir	Visualisasi tampak lebih besar.
Kendala	Pembesaran visualisasi terbatas oleh batas maksimum pixel yang dapat ditampung oleh elemen *canvas* HTML5.



Gambar 5.2 Hasil Uji Coba Zoom-In

5.2.1.2 Pengujian Zoom-Out

Pengujian *zoom-out* dilakukan untuk mengetahui keberhasilan aplikasi dalam memperkecil kanvas sehingga kanvas menjadi dua kali lebih kecil dari sebelumnya. Hasil pengujian dapat dilihat pada Tabel 5.3 dan Gambar 5.3.



Gambar 5.3 Hasil Uji Coba Zoom-Out

Tabel 5.3 Uji Coba Zoom-Out

ID	UJ- 03
Nama	Uji Coba Zoom-Out
Tujuan Uji Coba	Pengguna dapat memperkecil kanvas, sehingga visualisasi tampak lebih kecil.
Kondisi awal	Pengguna telah membuka dokumen data sekolah.
Skenario	Pengguna menekan tombol *zoom-out*
Keluaran yang diharapkan	Kanvas menjadi dua kali lebih kecil dari sebelumnya.
Hasil uji coba	Berhasil
Kondisi akhir	Visualisasi tampak lebih kecil.
Kendala	-

5.2.2 Pengujian Aplikasi Terhadap Data Masukan

Pengujian aplikasi dilakukan dengan menyiapkan beberapa data masukan (data sekolah) sebagai contoh kasus. Aplikasi akan digunakan dengan beberapa data masukan tersebut. Ada lima data masukan yang digunakan yang terdiri atas data SMA di Kecamatan Sukolilo dan empat data *dummy*. Hasil pengujian atau visualisasinya dapat dilihat pada lampiran.

5.3 Evaluasi

Tahap evaluasi akan dibagi menjadi dua bagian, yaitu evaluasi pengujian fungsionalitas dan evaluasi pengujian aplikasi terhadap data masukan.

5.3.1 Evaluasi Pengujian Fungsionalitas

Rangkuman mengenai hasil pengujian fungsionalitas dapat dilihat pada Tabel 5.4. Berdasarkan data pada tabel tersebut, dapat disimpulkan bahwa semua skenario pengujian berhasil dijalankan.

Sehingga dapat disimpulkan bahwa fungsionalitas dari aplikasi telah bekerja sesuai dengan yang diharapkan.

Tabel 5.4 Evaluasi Pengujian Fungsionalitas

ID	Nama	Hasil
UJ-01	Uji Coba Membuka Data Sekolah dan Menampilkan Visualisasi	Berhasil
UJ-02	Uji Coba Zoom-In	Berhasil
UJ-03	Uji Coba Zoom-Out	Berhasil

5.3.2 Evaluasi Pengujian Aplikasi Terhadap Data Masukan

Rangkuman mengenai hasil pengujian apllikasi terhadap data masukan dapat dilihat pada Tabel 5.5. Berdasarkan data pada tabel tersebut, dapat disimpulkan bahwa data masukan untuk pengujian berhasil dijalankan dan dibuat visualisasi pemetaan *coverage*-nya. Sehingga dapat disimpulkan bahwa aplikasi telah bekerja sesuai dengan yang diharapkan.

Tabel 5.5 Evaluasi Pengujian Apliasi Terhadap Data Masukan

No.	Nama	Hasil
1.	Data SMA di Kecamatan Sukolilo	Berhasil
2.	Data *dummy* (1)	Berhasil
3.	Data *dummy* (2)	Berhasil
4.	Data *dummy* (3)	Berhasil
5.	Data *dummy* (4)	Berhasil

(Halaman ini sengaja dikosongkan)

BAB 6
KESIMPULAN DAN SARAN

Pada bab ini dijelaskan mengenai kesimpulan yang dapat diambil dari hasil pengujian yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, juga diberi saran untuk pengembangan aplikasi kedepannya.

6.1 Kesimpulan

Dari hasil pengamatan selama proses pengerjaan mulai dari proses perancangan, implementasi, dan pengujian yang telah dilakukan terhadap aplikasi, diambil kesimpulan sebagai berikut:

1. Aplikasi telah berhasil menghitung diagram Voronoi dari beberapa titik dan sebuah luasan bidang.
2. Aplikasi telah berhasil membuat visualisasi pemetaan *coverage* sekolah.
3. Elemen *canvas* pada HTML5 dapat digunakan untuk menampilkan atau membuat sebuah visulisasi, gambar, atau diagram pada halaman web. Namun, elemen tersebut belum mampu untuk menampilkan konten grafis 3D.
4. Elemen *canvas* pada HTML5 mempunyai batasan untuk besar gambar yang dapat dimuat, sehingga ada kekurangan untuk menampilkan konten grafis dengan ukurannya sangat besar.

6.2 Saran

Adapun saran yang diberikan untuk mengembangkan aplikasi kedepannya antara lain:

1. Batasan area tidak berupa segi empat, namun bisa sesuai dengan perbatasan area seperti perbatasan kota atau perbatasan pulau dengan laut.

2. Menambahkan beberapa *event* yang dapat dilakukan oleh pengguna terhadap *canvas* seperti klik atau *hover* untuk menampilkan lebih banyak informasi.
3. Melakukan perhitungan pada sebuah server, bukan di *browser* pengguna.
4. Menambahkan fitur simpan/muat data dengan sebuah server sehingga visualisasi yang pernah dibuat dapat dibuka kembali oleh pengguna.
5. Mengganti algoritma untuk menghitung diagram Voronoi dengan algoritma yang lebih efisien/cepat, sehingga aplikasi dapat melakukan visualisasi untuk data yang lebih besar.

DAFTAR PUSTAKA

[1] F. Aurenhammer, "Voronoi Diagrams - A Survey of a Fundamental Geometric Data Structure," 1991.

[2] "Bowyer–Watson algorithm - WIkipedia," [Online]. Available: https://en.wikipedia.org/wiki/Bowyer%E2%80%93Watson_algorithm. [Accessed 12 12 2016].

[3] "Canvas element - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Canvas_element. [Accessed 12 12 2016].

[4] "Delaunay triangulation - WIkipedia," [Online]. Available: https://en.wikipedia.org/wiki/Delaunay_triangulation. [Accessed 12 12 2016].

[5] "HTML - WIkipedia," [Online]. Available: https://en.wikipedia.org/wiki/HTML. [Accessed 12 12 2016].

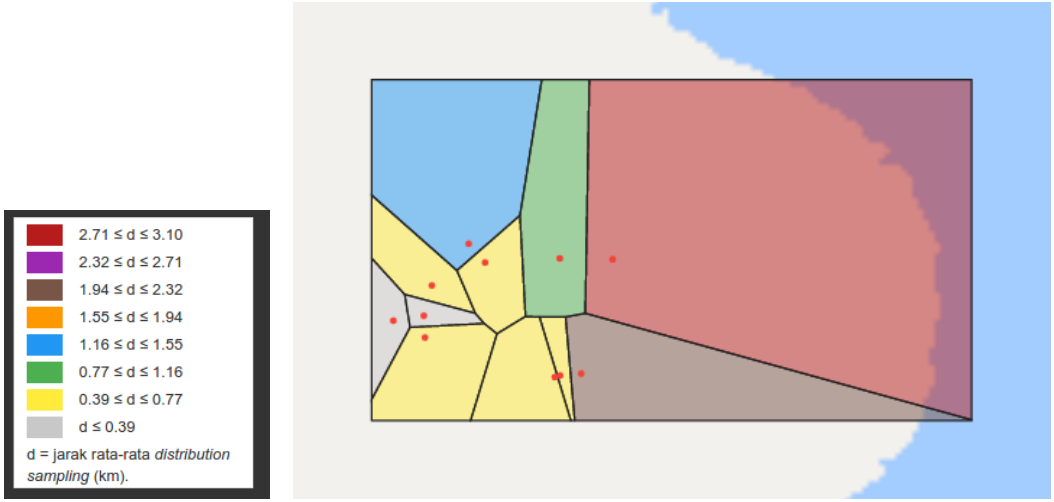
[6] "HTML element - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/HTML_element. [Accessed 12 12 2016].

[7] "HTML5 - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/HTML5. [Accessed 12 12 2016].

[8] "Voronoi diagram - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Voronoi_diagram. [Accessed 2016 12 2016].

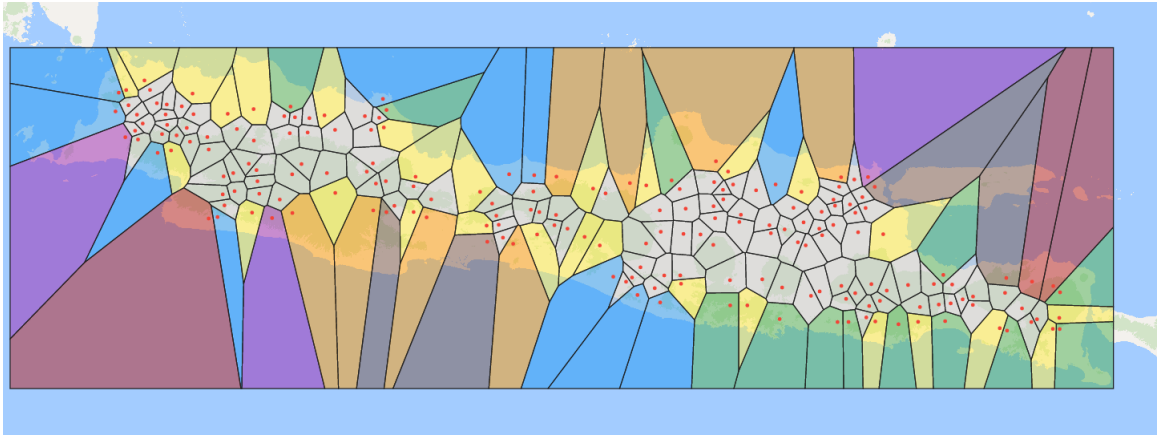
(Halaman ini sengaja dikosongkan)

LAMPIRAN

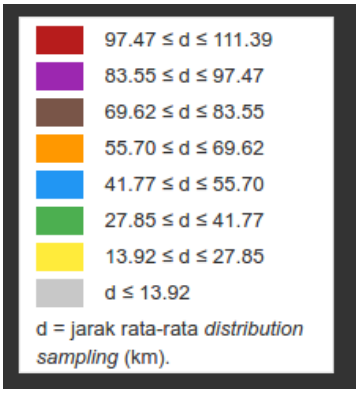


Gambar 1. Visualisasi coverage SMA di Kecamatan Sukolilo

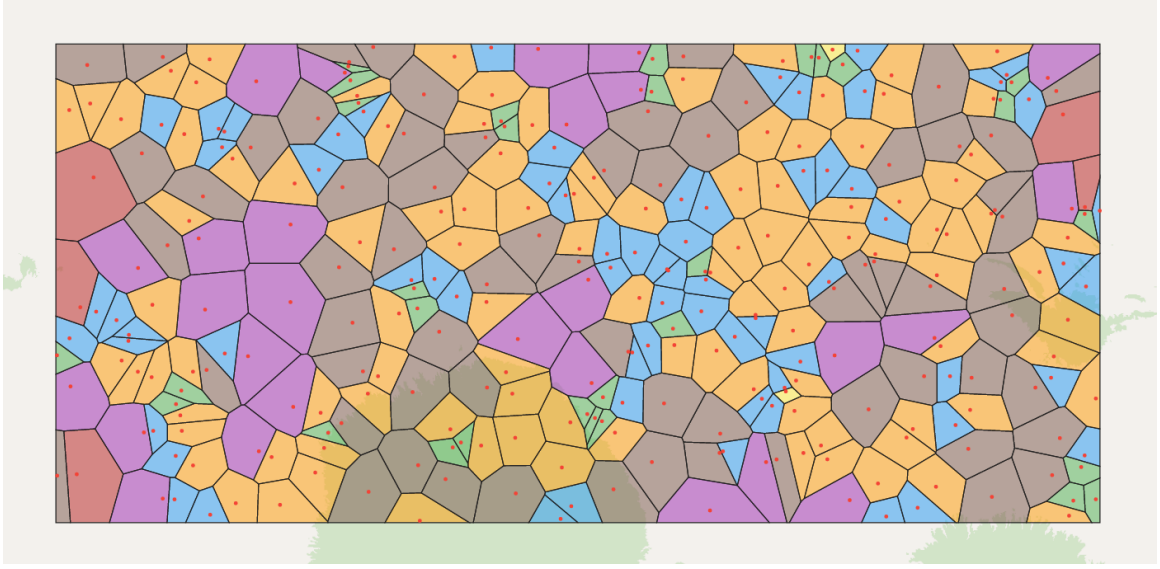
(Halaman ini sengaja dikosongkan)



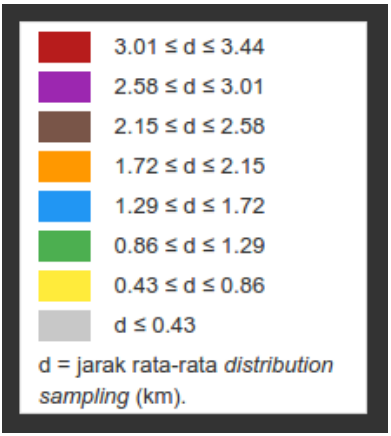
Gambar 2. Visualisasi coverage data dummy (1)



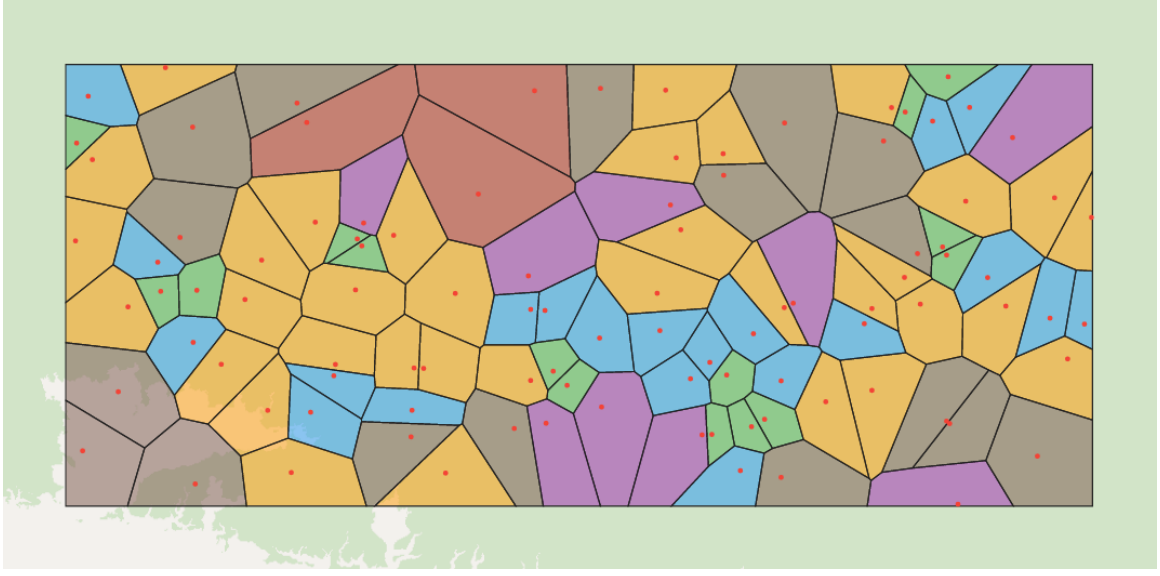
Gambar 3. Legenda visualisasi coverage data dummy (1)



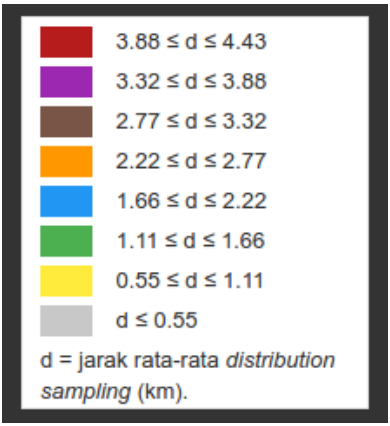
Gambar 4. Visualisasi coverage data dummy (2)



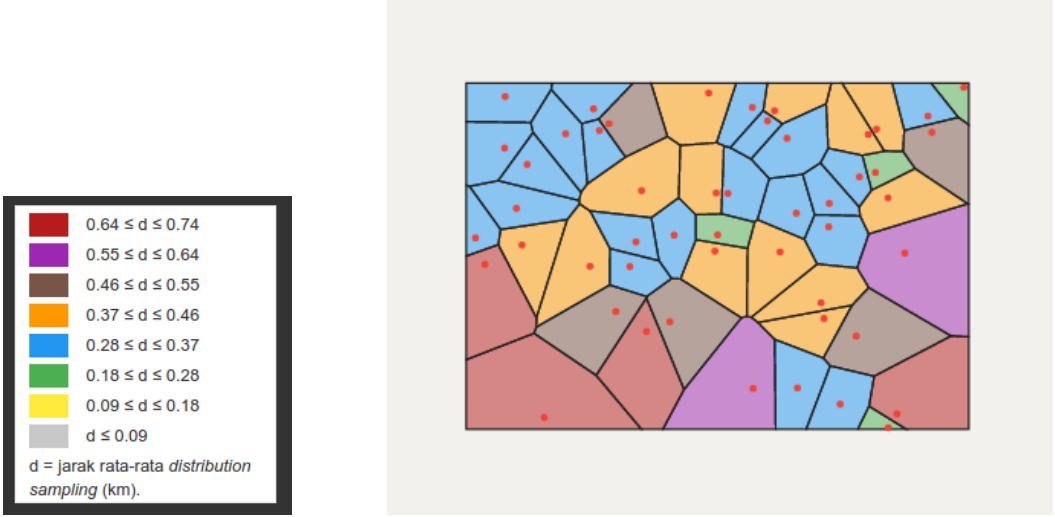
Gambar 5. Legenda visualisasi coverage data dummy (2)



Gambar 6. Visualisasi coverage data dummy (3)



Gambar 7. Legenda visualisasi coverage data dummy (3)



Gambar 8. Visualisasi coverage data dummy (4)

(Halaman ini sengaja dikosongkan)

BIODATA PENULIS

Penulis, Andrys Daniel Silalahi, lahir pada tanggal 23 November 1994 di Medan, Sumatera Utara.

Penulis menempuh pendidikan formal di SD St. Antonius VI Medan (2000-2006), SMP St. Maria Medan (2006-2009), SMA Negeri 1 Medan (2009-2012), dan S1 Teknik Informatika FTIf Institut Teknologi Sepuluh Nopember Surabaya, Jawa Timur.

Di jurusan Teknik Informatika, penulis mengambil bidang minat Interaksi Grafika dan Seni atau biasa disingkat dengan IGS dan memiliki ketertarikan dalam bidang desain web dan perancangan perangkat lunak. Selama masa perkuliahan, penulis turut aktif dalam Himpunan Mahasiswa Teknik Computer-Informatika (HMTC) ITS dan Badan Eksekutif Mahasiswa Fakultas Teknologi Informasi. Penulis dapat dihubungi melalui alamat surel andrys.silalahi@gmail.com.