



**UNIVERSIDAD MICHOACANA DE SAN
NICOLÁS DE HIDALGO**

FACULTAD DE INGENIERÍA ELÉCTRICA

TESIS:

**VISUALIZACIÓN DE TRAYECTORIAS DE
TORMENTAS UTILIZANDO VENTANAS
DE TIEMPO Y AGRUPAMIENTO
ESPACIAL BASADO EN K-MEANS**

Para obtener el Título de
INGENIERO EN COMPUTACIÓN

Presenta: Christian Viveros Tapia

Asesor de Tesis: MC. José Ortiz Bejar

Morelia, Michoacán, Diciembre del 2015

Índice general

Lista de figuras	5
Abstract	7
Resumen	8
1. INTRODUCCIÓN	9
1.1. ANTECEDENTES	10
1.2. DESCRIPCIÓN DEL PROBLEMA	11
1.3. OBJETIVOS	12
1.4. JUSTIFICACIÓN	13
2. HERRAMIENTAS DE DESARROLLO WEB	15
2.1. API Google Maps	16
2.2. MongoDB	18
2.3. NodeJS	20
2.4. JavaScript	21
2.5. Conclusiones	22
3. ALGORITMOS	23
3.1. Algoritmos de aprendizaje	23

3.2. K-MEANS	26
3.2.1. Implementación del algoritmo de K-means en JavaScript	31
3.2.2. Integración del algoritmo de K-means con Google Maps	33
3.3. Vincenty	36
3.4. Conclusiones	38
4. PRUEBAS Y RESULTADOS	39
4.1. DESCRIPCIÓN	39
4.2. Ejecución de la aplicación Web	42
4.3. Conclusiones	49
5. CONCLUSIONES	51
5.1. Conclusiones generales	51
5.2. Trabajo futuro	52
Bibliografía	55

Índice de figuras

2.1. Coordenadas geográficas	18
3.1. Inicialización	28
3.2. Asignación	29
3.3. Actualización	29
3.4. Repetición de segundo y tercer paso	30
3.5. Inicialización y primera iteracion, JavaScript	32
3.6. Reasignacion y actualización, Javascript	33
3.7. Icono descarga	34
3.8. Icono centroide	34
3.9. Implementación, Api Google Maps	35
4.1. Estructura de la App	40
4.2. Campos del documento en MongoDB	40
4.3. Campos del objeto JavaScript en Json	41
4.4. Icono nube	42
4.5. Icono rayo	42
4.6. Trayectoria para dos tormentas	43
4.7. Trayectoria para dos tormentas	44

4.8. Trayectoria para dos tormentas con descripción	45
4.9. Representación puntual de las descargas en las trayectorias de dos tormentas para cada ventana de tiempo	46
4.10. Representación puntual de las descargas en las trayectorias de dos tormentas	47
4.11. Representación puntual de las descargas en las trayectorias de dos tormentas con una mala elección de K	48
4.12. Trayectoria para tres tormentas con descripción	50

Abstract

In this work the development of a web application that displays storm tracks for studying atmospheric discharges is presented. Lightning are detected with an antenna Boltek LD-250 that captures the distance and angle at which the discharge occurs. In complement with a point representation of lightning, it is proposed to create the route with the aim of providing the Federal Electricity Commission a more complete analysis of lightning detected on the geographical region comprising the West Central Division.

With display in a graphical map reference of the areas with the highest incidence of atmospheric discharges will. And this information serve as a basis for future work, which may pose a probabilistic analysis in order to make decisions about risk assessment for the installation of new substations and transmission equipment, such as high-voltage lines, capacitors , banks reactors and transformers, etc.

During the course of this work the problem of plotting geographic locations is addressed through the Google Maps API, in order to have a real reference of the location of lightning, as well as for graphic identification of storms in space and obtaining paths storms in case of time.

Resumen

En este trabajo se presenta la elaboración de una aplicación web que permite visualizar trayectorias de tormentas para el estudio de Descargas Atmosféricas. Las Descargas Atmosféricas son detectadas con una antena Bolttek LD-250 que captura la distancia y ángulo al que ocurre la Descarga. En complemento con una representación puntual de las descargas atmosféricas, se propone generar la ruta con el objetivo de proporcionar a la Comisión Federal de Electricidad un análisis mas completo de las descargas atmosféricas detectadas sobre la región geográfica que comprende la División Centro Occidente.

Con la visualización en un mapa se tendrá una referencia gráfica de las zonas de mayor incidencia de Descargas Atmosféricas. Y esta información servirá como base para un trabajo futuro, que podrá plantear un análisis probabilístico con el fin de tomar decisiones de valoración de riesgos referentes a la instalación de nuevas subestaciones etc.

Durante el desarrollo de este trabajo se abordará la problemática de graficar puntos geográficos mediante el API de Google Maps, con la finalidad de tener una referencia real de la localización de las descargas atmosféricas, así como, para obtener la identificación gráfica de tormentas en el espacio, y la obtención de las trayectorias de estas tormentas en el caso del tiempo.

Palabras clave: Api, Google Maps, NodeJs, MongoDB, JavaScript, descarga atmosférica.

Capítulo 1

INTRODUCCIÓN

Las tormentas producen grandes cantidades de lluvia en periodos de tiempo muy corto, con ello causando inundaciones bruscas y repentinas. Además de que estas son difíciles de prever y son causa de graves daños materiales e incluso de vidas humanas.

En la actualidad, la actividad eléctrica en forma de rayos de una tormenta puede ser detectada y obtenida en tiempo real desde grandes distancias a través de antenas terrestres, con ello es posible aprovechar la estrecha correlación de la actividad eléctrica atmosférica y las lluvias.

La observación de rayos en tormentas eléctricas permite utilizar los datos detectados para proporcionar la localización, intensidad y comportamiento de precipitaciones. Aunque la sola observación de las descargas atmosféricas es extremadamente útil, es también necesario la separación espacial y temporal de la actividad eléctrica. Esto resulta de utilidad para describir el movimiento del fenómeno y generar la trayectoria realizada por el mismo.

Utilizando la información generada por el fenómeno de tormentas sería posible mejorar el proceso de predicción del clima. El pronóstico del clima se hace por lo general; considerando elementos como el viento, temperatura, presión atmosférica, precipitaciones y humedad. Siendo muy poco común que se utilicen los datos de descargas atmosféricas.

1.1. ANTECEDENTES

Se cuenta con tres antenas ubicadas en Morelia, lázaro cárdenas y Yurecuaro. Como fuente de datos se tienen archivos de texto, los cuales son generados por un software que viene ya incluido con las antenas utilizadas.

Como antecedentes existen dos trabajos previos que utilizan la información generada por las antenas. En el primero, el objetivo era respaldar la información contenida en los archivos de texto generados por el software de la antena, en una base de datos MySQL. El sistema implementado permite visualizar la información puntualmente de la localización aproximada de los eventos registrados, pero graficados sobre una fotografía del planeta a diferentes niveles de acercamiento. El nivel de acercamiento se debe realizar sobre la posición de la antena, específicamente sobre la antena instalada en la UMSNH. Los puntos se graficaban mediante puntos coloreados sobre las fotografías del territorio geográfico cercano a la antena. La implementación de acercamiento limita a sólo visualizar el área cercana a la antena y aumenta el nivel de acercamiento, el área geográfica visual disminuye. Esta implementación no da la posibilidad de desplazarse libremente sobre el mapa. Además se hace uso de geometría plana para determinar las componentes en x y y

de los vectores de posición, generando errores debido a la irregularidad de la forma del planeta.

En el segundo se muestra la ventaja que resulta el estimar la Función de Densidad de Probabilidad de las descargas atmosféricas para evaluar gráficamente las zonas de mayor incidencia de descargas sobre una región geográfica. Es clara la ventaja de usar estas gráficas de densidad con respecto a una representación puntual de las descargas atmosféricas. Se crearon y guardaron las imágenes georeferenciadas de la estimación de la Función de Densidad de Probabilidad con algunos métodos implementados y después se sobreponen sobre mapas estáticos.

1.2. DESCRIPCIÓN DEL PROBLEMA

La estructura del Sistema Eléctrico Nacional en el país es claramente susceptible a las inclemencias del tiempo, además es una estructura que cambia con el paso del tiempo y es importante llevar una estadística del comportamiento de la misma. En nuestro caso particular como inclemencia del tiempo de interés, son las descargas atmosféricas. En la CFE existe una entidad que está a cargo del control operativo del Sistema Eléctrico Nacional. El Centro Nacional de Control de Energía, que se encarga de dar seguimiento detallado a las incidencias de fallas en los equipos instalados debido a las inclemencias del tiempo.

Es común tener situaciones de indisponibilidad relacionadas al fenómeno de descargas atmosféricas y tormentas eléctricas. Prevenir estas situaciones

son la motivación principal del desarrollo de este trabajo, así como el tener estudios referentes al comportamiento de los eventos atmosféricos que puedan afectar el funcionamiento del sistema eléctrico. Otro interés a largo plazo es el tener pronósticos en tiempo real del comportamiento de las descargas atmosféricas. Esto para visualizar sobre mapas que identifiquen la localización geográfica de las instalaciones y de esta manera poder ver el posible cruce de estas instalaciones con las trayectorias de los siniestros, tales como ciclones o tormentas eléctricas.

Tal es el interés sobre estos estudios, que se desea la instalación de más antenas para la detección de descargas atmosféricas por parte de CFE de tal forma que se obtenga la mejor medición posible de la información recabada.

Las descargas eléctricas tienen un gran impacto sobre las fallas detectadas en el Sistema Eléctrico Nacional. Este proyecto aporta información de manera gráfica referente a la distribución espacial y temporal de las descargas atmosféricas. Con el objetivo de que se puedan tomar decisiones tácticas en tiempo.

1.3. OBJETIVOS

En la tesis se plantean los siguientes objetivos:

- Elegir las tecnologías adecuadas para desarrollar una aplicación web, utilizando las herramientas actuales que se consideren las más eficientes.

- Inyectar la información contenida en los archivos de texto generados por el software de las antenas (NEXTORM) a una base de datos.
- La implementación del algoritmo K-means.
- Separación espacial utilizando K-means y ventanas de tiempo.
- Visualizar la trayectoria que siguen las tormentas.

1.4. JUSTIFICACIÓN

En el presente trabajo se busca contribuir con el desarrollo de una aplicación para la presentación ordenada de gran cantidad de datos de eventos atmosféricos. Aportando el algoritmo de K-means implementado para proporcionar la división e identificación de las tormentas, al igual que proporcionar la separación de datos en estampas de tiempo. Y obtener la trayectoria de estas para visualizarlas.

Capítulo 2

HERRAMIENTAS DE DESARROLLO WEB

Para este trabajo en un inicio se tuvo que discutir las posibles herramientas que se utilizarían, se buscaba que cumplieran algunos requisitos como la portabilidad para poder facilitar el uso e instalación por cualquier usuario, para esto al indagar en posibles tecnologías en este aspecto se eligió NodeJs para implementar la aplicación servidor, ya que es compatible con los sistemas operativos mas usados y se puede realizar la migración de un sistema a otro de forma simple y transparente.

Al igual se eligió como manejador de base de datos el esquema NoSQL de MongoDB del cual se tienen buenas referencias de eficiencia, además de que sus documentos se guardan en formato Json, que en MongoDB se llama Bson y que coincide con uno de los formatos compatibles que maneja el API de Google Maps.

El API de Google Maps es otra herramienta web que cuenta con gran cantidad de funciones y herramientas para visualización, además de ejemplos básicos para cada una de sus apartados y que facilitan el comienzo para desarrollar las implementaciones requeridas para este trabajo.

2.1. API Google Maps

El API de Google Maps es un servidor de aplicaciones de mapas en la web que pertenece a Google. Fue lanzado en febrero de 2005. Google Maps ofrece la capacidad de realizar acercamientos y alejamientos para mostrar el mapa. Ofrece imágenes de satélite, mapas de calles, vistas 360° panorámicas de calles (Street View), etc.[3][5]

La mayoría de las imágenes de las ciudades son de alta resolución, algunas son fotografías aéreas tomadas desde un avión mientras que la mayoría de imágenes son tomadas de satélites. Las imágenes disponibles de satélite se actualizan de forma regular.

Al igual que muchas otras aplicaciones web de Google, Google Maps utiliza JavaScript. Google maps sirve para habilitar las características del mapa de clase mundial en cualquier sitio web, incluyendo mapas con estilo, edificios 3D, planos de planta de interior, geocodificación, datos de lugares, y más utilidades posibles para implementar.

Después del éxito que se tuvo con Google Maps, Google lanzó el API de Google Maps en junio de 2005. El API permite a los desarrolladores integrar

Google Maps en sus sitios web como un servicio gratuito. La documentación de Google maps está enfocada para usuarios familiarizados con JavaScript y que tengan nociones de programación orientada a objetos, pero también se debe estar familiarizado con los mapas de Google desde el punto de vista del usuario.

La documentación es conceptual y diseñada para que se pueda empezar rápidamente la exploración y el desarrollo de aplicaciones. Toda aplicación que usa mapas debe cargar el API de Google Maps usando una clave de API. El uso de esta clave de API le permite a google supervisar el uso de mapas en la aplicación para regular su uso por parte de la aplicación, ya que existe un límite de uso.

Al utilizar el Google Maps API, es posible incrustar Google Maps en un sitio web externo. Se menciona el uso de este servicio por más de un millón de sitios web que lo hace el API de desarrollo de aplicaciones web mas utilizado. La API de Google Maps es gratuita siempre que el lugar sobre el que se está utilizando sea de acceso público, y este generando menos de veinticinco mil accesos al día.

El API consiste de archivos JavaScript que contienen las clases, métodos y propiedades que se usan para manipular el comportamiento de los mapas. Las coordenadas están expresadas usando números decimales separados por coma. La latitud siempre precede la longitud. En los mapas físicos, las coordenadas están expresadas en grados, así que hay que convertir los datos a coordenadas geográficas. Para nuestro caso la latitud siempre es positiva y

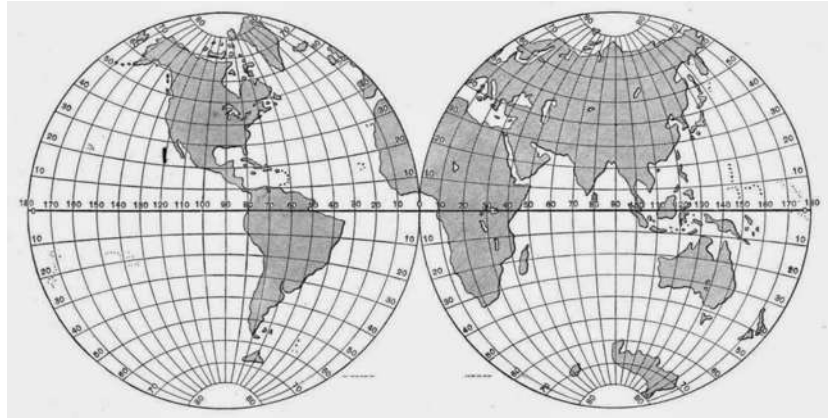


Figura 2.1: Coordenadas geográficas

la longitud siempre es negativa debido a que la posición de nuestro país se encuentra en la latitud Norte y Longitud Oeste como se aprecia en la imagen 2.1

2.2. MongoDB

MongoDB es un sistema de base de datos NoSQL orientado a documentos, desarrollado bajo el concepto de código abierto. MongoDB forma parte de la nueva familia de sistemas de base de datos NoSQL. En vez de guardar los datos en tablas como se hace en las base de datos relacionales, MongoDB guarda estructuras de datos en documentos tipo JSON con un esquema dinámico (MongoDB llama ese formato BSON), haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida.[11][7]

Algunas de las características de MongoDB son:

- Consultas Ad hoc, MongoDB soporta la búsqueda por campos, consul-

tas de rangos y expresiones regulares. Las consultas pueden devolver un campo específico del documento.

- Indexación, cualquier campo en un documento de MongoDB puede ser indexado, al igual que es posible hacer índices secundarios. El concepto de índices en MongoDB es similar a los encontrados en base de datos relacionales.
- MongoDB tiene la capacidad de realizar consultas utilizando JavaScript, haciendo que estas sean enviadas directamente a la base de datos para ser ejecutadas.
- Manipulación de datos, en colecciones y documentos.
- MongoDB guarda la estructura de los datos en documentos tipo JSON con un esquema dinámico llamado BSON, lo que implica que no existe un esquema predefinido.
- Los elementos de los datos se denominan documentos y se guardan en colecciones.
- Una colección puede tener un número indeterminado de documentos. Comparando con una base de datos relacional, se puede decir que las colecciones son como tablas y los documentos son registros en la tabla. La diferencia es que en una base de datos relacional cada registro en una tabla tiene la misma cantidad de campos, mientras que en MongoDB cada documento en una colección puede tener diferentes campos.
- En un documento, se pueden agregar, eliminar, modificar o renombrar

nuevos campos en cualquier momento, ya que no hay un esquema predefinido.

- En MongoDB la clave es el nombre del campo y el valor es su contenido, los cuales se separan mediante el uso de “:”. Como valor se pueden usar números, cadenas o datos binarios.

El modelo de datos de documentos de MongoDB hace que sea fácil almacenar datos de cualquier estructura y modificar dinámicamente el esquema. Los datos en MongoDB tienen un esquema flexible. A diferencia de las bases de datos SQL, en el que debe determinar y declarar el esquema de una tabla antes de insertar los datos.

En las colecciones de MongoDB no es obligatorio cumplir la estructura fiel del documento. Esta flexibilidad facilita el mapeo de documentos a una entidad o un objeto. Cada documento puede coincidir con los campos de datos de la entidad representada, incluso si los datos tienen una variación sustancial. En la práctica, sin embargo, los documentos en una colección comparten una estructura similar.

2.3. NodeJS

NodeJS fue creado por Ryan Dahl en 2009, es un entorno en tiempo de ejecución multiplataforma, de código abierto, para la capa del servidor pero no se limita a ello. Basado en el lenguaje de programación ECMAScript, asíncrono, con I/O de datos en una arquitectura orientada a eventos y basado

en el motor V8 de Google. Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables.[9][8]

Al contrario que la mayoría del código JavaScript, no se ejecuta en un navegador, sino en el servidor. NodeJs utiliza el motor V8 que es el ambiente de ejecución para JavaScript creado para Google Chrome. Dicho software es libre desde 2008, está escrito en C++ y compila el código fuente JavaScript en código de máquina en lugar de interpretarlo en tiempo real. Los módulos de terceros pueden extender nodeJS o añadir un nivel de abstracción, implementando varias utilidades para utilizar en aplicaciones web, como por ejemplo el Frameworks Express. Pese a que los módulos pueden instalarse como archivos simples, normalmente se instalan utilizando el Node Package Manager (NPM) que facilita la compilación, instalación y actualización de módulos así como la gestión de las dependencias.

Node.js puede ser combinado con una base de datos documental por ejemplo MongoDB que utilizamos para nuestra aplicación o CouchDB, y combinado con JSON lo que permite desarrollar en un ambiente de desarrollo JavaScript unificado.

2.4. JavaScript

JavaScript es un lenguaje de programación web, que se utiliza actualmente para desarrollo de paginas web, tiene funciones muy potentes que hace que una pagina web pueda ser dinámica, rápida y capaz de mejorar la vista de la interface web. JavaScript al igual que otros lenguajes de programación web,

es un lenguaje interpretado. Lo que significa que el navegador web descarga el código de una página web, y al encontrar en esta página código javascript, lo interpreta y lo ejecuta en la computadora del cliente.[15][6]

JavaScript es un lenguaje muy poderoso, utilizado para crear páginas web dinámicas utilizando pocos recursos de memoria. Es un lenguaje muy sencillo, tiene gran documentación en la web, y es totalmente libre.

2.5. Conclusiones

En este apartado se describen las características generales de las herramientas que se utilizaron y la gran capacidad que se tiene con ellas para el desarrollo de aplicaciones, cada una cuenta con una documentación extensa por lo que solo se describen aspectos generales y no se mencionan temas específicos.

Capítulo 3

ALGORITMOS

3.1. Algoritmos de aprendizaje

Una de las tareas de la computación es construir sistemas que sean capaces de aprender. El aprendizaje no sólo se encarga de obtener el conocimiento, sino también la forma en que éste se representa. En los conjuntos de datos, se distinguen dos tipos, el conjunto de entrenamiento y el conjunto de prueba. El modelo o clasificador, es una conexión entre las variables de entrada (mediciones) y las que se van a predecir, usualmente las variables que se van a predecir se denominan variables dependientes y las restantes en variables independientes. Un algoritmo de aprendizaje, es cualquier procedimiento utilizado para construir un modelo a partir del conjunto de datos de entrenamiento.[1][12][13][2]

Existen diversas tareas que se pueden hacer con sistemas de aprendizaje, una de ellas es la segmentación de datos que usa algoritmos de clustering

tales como el algoritmo de K-means. La segmentación es la separación de los datos en subgrupos o clases de interés. Las clases pueden ser exhaustivas y mutuamente exclusivas o jerárquicas y con traslapes. Se puede utilizar con otras técnicas de minería de datos considerando cada subgrupo de datos por separado.

Existen diferentes tipos de aprendizajes, uno de ellos es el inductivo. El aprendizaje inductivo, es la capacidad de obtener conclusiones, más generales de información específica a partir de ejemplos. Busca patrones comunes que expliquen los ejemplos, se basa en el razonamiento, el conocimiento obtenido es nuevo no preserva la verdad lo que significa que nuevo conocimiento puede invalidar el ya obtenido.

Los algoritmos de aprendizaje se pueden dividir en dos grupos: los algoritmos supervisados y los no supervisados. Mientras que los algoritmos no supervisados consisten en encontrar la partición más adecuada del conjunto de entrada a partir de similitudes entre sus ejemplos, los algoritmos supervisados intentan extraer aquellas propiedades que permiten discriminar mejor la clase de cada ejemplo, y como consecuencia requieren de una clasificación previa llamada supervisión del conjunto de entrenamiento.

Para el aprendizaje Inductivo no Supervisado no existe una clasificación de los ejemplos. Se busca descubrir la manera más adecuada de particionar los ejemplos. El aprendizaje se guía por la similitud entre ejemplos. Existen criterios heurísticos para guiar la búsqueda. Tendrá de resultado una partición de los ejemplos y una descripción de la partición.

Algoritmo inductivo no supervisado “Clustering”, es el proceso de agrupar datos en clases o clusters de tal forma que los objetos de un cluster tengan una similitud alta entre ellos, y que sean muy diferentes con objetos de otros clusters.

Características del algoritmo inductivo no supervisado “Clustering”:

- Clusters de formas arbitrarias, los basados en distancias numéricas tienden a encontrar cluster esféricos.
- Capacidad de manejar diferentes tipos de atributos, numéricos (es lo más común), binarios, nominales, ordinales, etc.
- Capacidad de añadir restricciones.
- Manejo de ruido, muchos son sensibles a datos erróneos.
- Pueden funcionar eficientemente con alta dimensionalidad.
- Requerimientos mínimos para especificar parámetros como el número de clusters.
- Independientes del orden de los datos.
- Que los clusters sean interpretables y utilizables

El objetivo del clustering es identificar la clasificación característica de un conjunto de datos no etiquetados. Los algoritmos de clasificación de datos tienen numerosas aplicaciones en distintos ámbitos: Biología, Comercialización, Biblioteca, Seguros, Sismología, Urbanismo, etc. Ejemplos de su aplicabilidad

es el caso de la clasificación por especies de los distintos seres vivos. También en el estudio de los sismos, la reagrupación de los epicentros de los sismos observados permite determinar las zonas de riesgos, y poder ayudar a evitar catástrofes.

Definiciones de cluster:

- Cluster o grupo, es un conjunto de objetos que son “similares”, entre ellos y “diferentes”, de los objetos que pertenecen a los otros grupos.
- La palabra “cluster”, viene del inglés y significa agrupación. Desde un punto de vista general, el cluster puede considerarse como la búsqueda automática de una estructura o de una clasificación en una colección de datos “no etiquetados”.

3.2. K-MEANS

El algoritmo de K-means, creado por MacQueen en 1967 es el algoritmo de clustering más conocido y utilizado ya que es de muy simple aplicación y eficaz. Es un método de agrupamiento, que tiene por objetivo la partición de un conjunto de n observaciones en K grupos. En donde cada observación n es un dato capturado y pertenece al grupo K mas cercano a la media. K-means es traducido como K-medias.[10][14][16]

K-means tiende a encontrar grupos de extensión espacial similares, aunque con el uso de heurísticas a conveniencia es posible que los grupos tengan formas diferentes. K-means sigue un procedimiento simple de clasificación de

un conjunto de objetos en un determinado número K de clústeres, donde K es determinado a priori. K-means representa cada uno de los clusters por la media o media ponderada de sus puntos, es decir, por su centroide.

La representación mediante centroides tiene la ventaja de que tiene un significado gráfico y estadístico inmediato. Cada cluster por tanto es caracterizado por su centro o centroide que se encuentra en el centro o en medio de los elementos que lo componen. Se utilizan centroides de cada grupo como representantes.

Describiendo un poco, dado un conjunto de observaciones

$$(x_1, x_2, \dots, x_n)$$

donde cada observación es un vector real, k-means construye una partición de las n observaciones en los K conjuntos

$$(k \leq n)S = (S_1, S_2, \dots, S_k)$$

El algoritmo del K-means se realiza en 4 etapas:

1. Primera etapa: Partiendo de una selección inicial de K centroides aleatoriamente que forman así los K clusters iniciales, que pueden ser K elementos de la colección seleccionados al azar, o los que se obtengan mediante la aplicación de alguna técnica de inicialización. El algoritmo k-means mas común utiliza una técnica de refinamiento iterativo. El primer paso es generar aleatoriamente los centroides iniciales, como se muestra en la figura

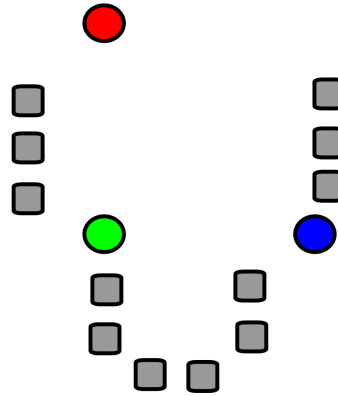


Figura 3.1: Inicialización

3.1

2. Segunda etapa o paso de asignación: Asigna o reasigna los objetos del cluster, cada uno de los elementos de la colección se asigna al grupo con el centroide más cercano. Cada objeto x se asigna al cluster del centroide más próximo al objeto, según una medida de distancia, normalmente la distancia euclidiana como se ve en la figura 3.2
3. Tercera etapa o paso de actualización: En seguida de que todos los objetos son asignados, calculan o recalculan los centroides de cada uno de los K cluster resultantes. En los primeros pasos se obtienen las mayores diferencias entre los centroides originales y los calculados luego de las reasignaciones. Los puntos de la colección vuelven a asignarse al grupo del centroide más cercano. La actualización se muestra en la figura 3.3
4. Cuarta etapa: Se repiten la segunda y tercera etapas hasta que no se hagan

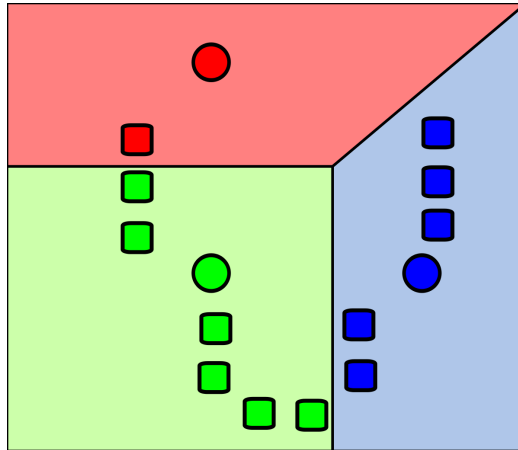


Figura 3.2: Asignación

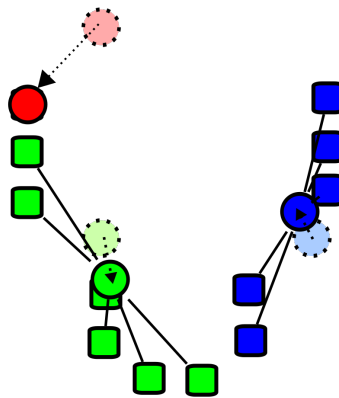


Figura 3.3: Actualización

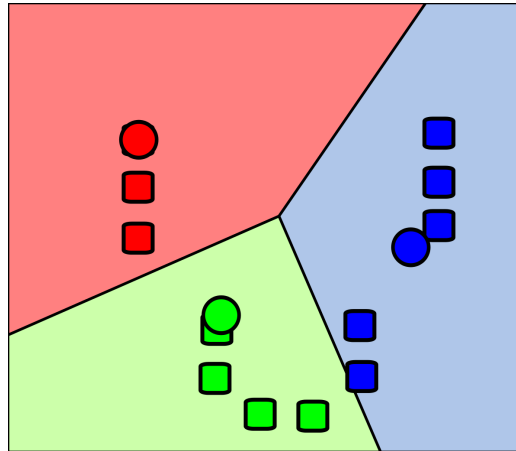


Figura 3.4: Repetición de segundo y tercer paso

más reasignaciones. Estos pasos se repiten hasta que los K centroides no cambian luego de una iteración que es equivalente a decir que el valor de la función utilizada como criterio de optimización no varía. Y se considera que el algoritmo ha convergido como se muestra en la figura 3.4

El algoritmo K-means encuentra una categorización que representa un óptimo local del criterio elegido. Aunque el algoritmo termina siempre, no se garantiza el obtener la solución óptima. En efecto, el algoritmo es muy sensible a la elección aleatoria de los K centros iniciales. Esta es la razón por la que se ejecuta el algoritmo del K-means numerosas veces sobre un mismo conjunto de datos para intentar minimizar este efecto, sabiendo que con centros iniciales lo más espaciados posibles dan mejores resultados.

Entre las características de éstos métodos pueden destacarse las siguientes:

- Pueden ser no deterministas, partiendo del mismo agrupamiento inicial, los métodos llegarán siempre a la misma solución, sin embargo, casi

todos los métodos para la selección inicial son no deterministas. El algoritmo evaluará diferentes agrupamientos cada vez que se lo ejecute, y si los grupos no están claramente separados podrá llegar a soluciones distintas.

- Pueden corregir errores cometidos en pasos anteriores, en cada paso del algoritmo los objetos de la colección se asignan al grupo más apropiado según el criterio de optimización, de esta manera el algoritmo va refinando el agrupamiento en cada iteración.
- Pueden implementarse en forma eficiente, las restricciones de recursos son la causa principal por la que se utilizan este tipo de métodos. La mayoría de las variantes pueden implementarse de forma que sus tiempos de ejecución sean $O(n)$.
- Si bien el algoritmo K-means es mucho más eficiente que los algoritmos jerárquicos, es dependiente de la selección inicial de centroides.
- Sus resultados pueden ser bastante pobres y suelen variar mucho si se aplica varias veces a la misma colección de documentos, ya que si la selección de centroides al azar es mala, la solución encontrada también lo será.

3.2.1. Implementación del algoritmo de K-means en JavaScript

Para probar el buen funcionamiento del algoritmo de K-means implementado en Javascript, se utiliza para graficar los resultados, una librería de

google que se llama ScatterChart para gráficas. En esta implementación del algoritmo K-means se realizan pruebas con datos artificiales.

Se realizan las primeras pruebas del algoritmo de K-means con el fin de obtener resultados visuales, antes de implementarlo con el API de Google Maps. Las primeras pruebas muestran las etapas y pasos claramente, con diferentes valores de K:

- a) Inicialización y primera iteración. Con K igual a tres en la iteración uno de tres, en la figura 3.5 se muestra la elección aleatoria de los K centroides que se muestran en color negro y los puntos en color gris. También se observa la primera iteración con la partición de las observaciones de acuerdo con el Diagrama de Voronoi generado por los centroides.

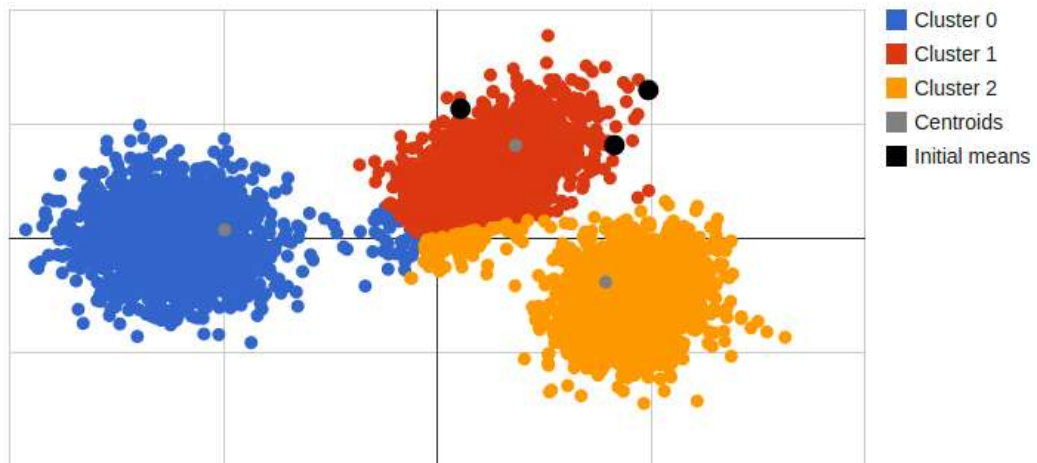


Figura 3.5: Inicialización y primera iteración, JavaScript

- b) Reasignación y actualización. El paso de actualización y asignación se repiten, y para la última iteración en este caso la tercera iteración se muestra

en color gris. Los centroides finales calculados y los datos separados en tres cluster como pueden observar en la figura 3.6

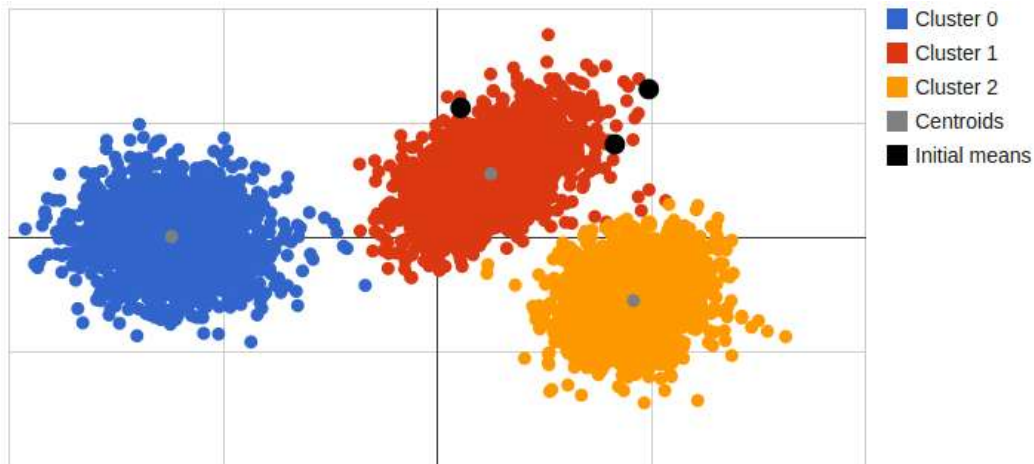


Figura 3.6: Reasignacion y actualización, Javascript

3.2.2. Integración del algoritmo de K-means con Google Maps

Se hace la intergración para probar el algoritmo con datos reales y verificar que se pueden graficar las observaciones de descargas mediante el Api de Google Maps. Las pruebas se realizaron con datos reales de la fecha del 1° de enero de 2012, con una cantidad de cinco mil puntos que cubren todos los datos del día. Los colores se eligen al azar por lo que no coinciden en las imágenes de inicio y de fin de las iteraciones de K-means. . El símbolo de rayo representa las descargas, figura 3.7 el símbolo de circulo representa los centroides, figura 3.8 y los colores representan las observaciones asignadas al clúster indicado.

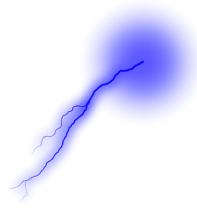


Figura 3.7: Icono descarga

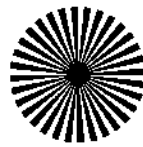
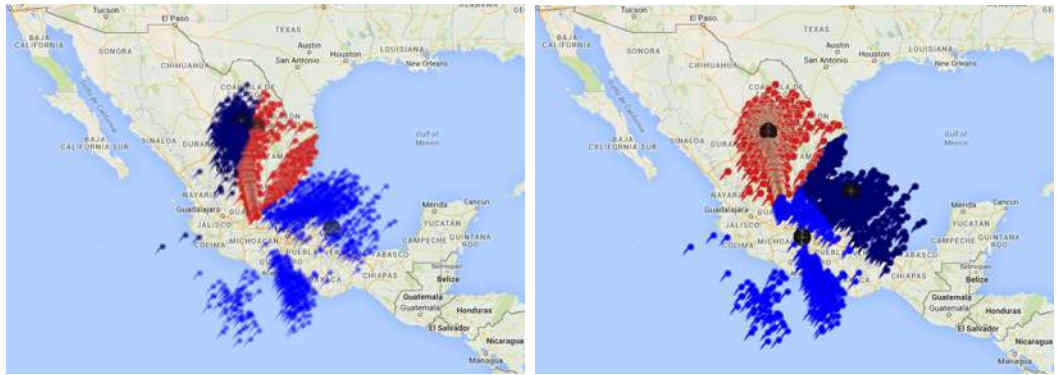


Figura 3.8: Icono centroide

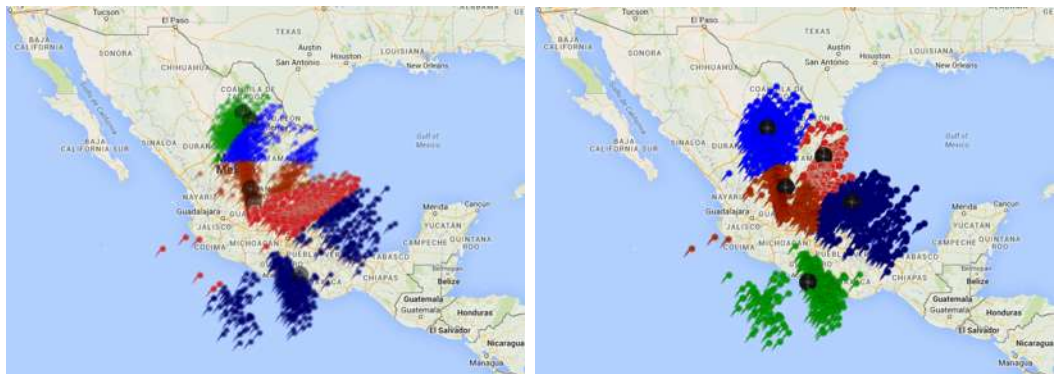
Se prueba para una elección de K igual a tres, igual cinco e igual a 10 centroides sobre los mismos datos. Se detiene en la primera iteración para ver la elección aleatoria de los centroides iniciales y en que lugar se grafican. Ya que se eligen estos primeros centroides se observa en las imágenes el cambio de color, se hace el primer paso de asignación. En las figuras: 3.9a, 3.9c, 3.9e se puede apreciar como forman las líneas divisorias del Diagrama de Voronoi en la separación de los colores.

Mientras en las figuras: 3.9b, 3.9d, 3.9f, se observa la convergencia para los tres casos y se puede observar la separación completa de todos los datos asignados a sus respectivos centroides. Se percibe como es que, independientemente de la forma de los datos el algoritmo de K -means trata de generar el número de cluster que se le indican con la elección de K .



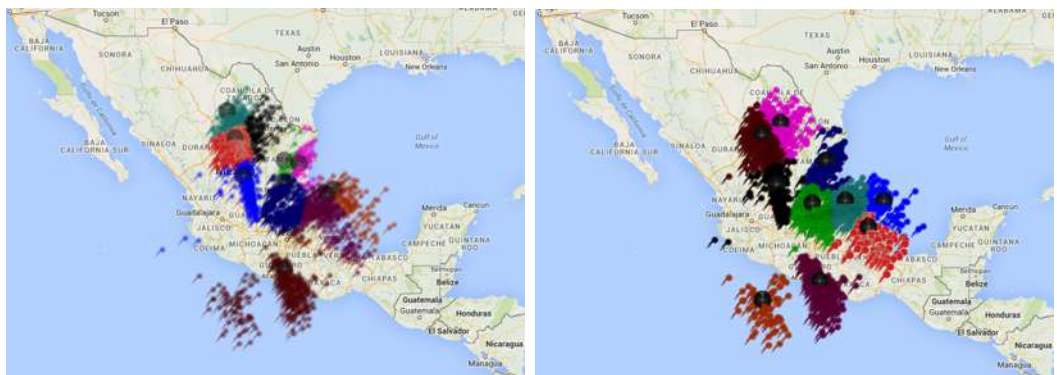
(a) Inicio con K = 3

(b) Final con K = 3



(c) Inicio con K = 5

(d) Final con K = 5



(e) Inicio con K = 10

(f) Final con K = 10

Figura 3.9: Implementación, Api Google Maps

3.3. Vincenty

Para este trabajo los datos con los que contamos son las coordenadas geográficas del lugar donde se encuentra instalada la antena y la distancia y el ángulo que se forma entre la antena y la descarga atmosférica. Por lo que tenemos nuestros eventos atmosféricos en coordenadas polares. A partir de los datos de cada evento podemos calcular los puntos geográficos correspondientes a descarga. El calcular los datos utilizando geometría plana trae consigo algunos problemas como un margen de error muy considerable, entonces, el problema se centra en calcular las coordenadas geográficas que corresponden a cada descarga atmosférica.

Esta problemática es solucionada mediante la implementación de los algoritmos formulados por Theodore Vincenty en la década de los 70's[4]. Aplicando la fórmula directa de Vincenty. En el algoritmo 1 pasamos como argumentos la localización geográfica de nuestra antena como primer punto en coordenadas geográficas y la información de la base de datos referente a la posición en coordenadas polares de cada descarga atmosférica, y como resultado se obtiene las coordenadas geográficas correspondientes a cada una de los datos de las descargas atmosféricas.

Al tener todas nuestras observaciones georeferenciadas podemos graficar de manera puntual y precisa mediante el API de Google Maps.

Algoritmo 1 Pseudocódigo de la fórmula directa de Vincenty.

Entrada: $\varphi_1, \lambda_1, s, \alpha_1$

Salida: Coordenadas del punto destino.

1: a, b semiejes mayor y menor del elipsoide.
2: $f = \frac{a-b}{a}$ Anchamiento de la elipsoide
3: $\tan(\psi_1) = \frac{(1-f)\tan(\varphi_1)}{\cos(\alpha_1)}$ aplicando la identidad $\tan(\Psi) = \frac{b}{a}\tan(\varphi)$.
4: $\cos(\psi_1) = 1/\sqrt{(1 + \tan^2(\psi_1))}$ por la identidad trigonométrica $\tan^2(x) + 1 = 1/\cos^2(x)$
5: $\sin(\psi_1) = \tan(\psi_1)\cos(\psi_1)$ por la identidad trigonométrica $\tan(x) = \sin(x)/\cos(x)$
6: $\sigma_1 = \text{atan2}(\tan(\psi_1), \cos(\alpha_1))$
7: $\sin(\alpha) = \cos(\psi_1)\sin(\alpha_1)$
8: $\cos^2(\alpha) = 1 - \sin^2(\alpha)$ por la identidad trigonométrica $\cos^2(x) + \sin^2(x) = 1$
9: $u^2 = \cos^2(\alpha) \frac{a^2 - b^2}{b^2}$
10: $A = 1 + u^2/16384\{4096 + u^2[-768 + u^2(320 - 175u^2)]\}$
11: $B = u^2/1024\{256 + u^2[-128 + u^2(74 - 47u^2)]\}$
12: $\sigma = s/b.A$ (1er aproximación) , $\sigma' = 2\pi$
13: $iterations = 0$ (limite del ciclo)
14: **mientras** $abs(\sigma - \sigma') > 10^{-12}$ && $iterations < 200$ **hacer**
15: $\cos(2\sigma_m) = \cos(2\sigma_1 + \sigma)$
16: $\Delta\sigma = B.\sin(\sigma)\{\cos(2\sigma_m) + B/4[\cos(\sigma)(-1 + 2\cos^2(2\sigma_m)) - B/6\cos(2\sigma_m)(-3 + 4\sin^2(\sigma))(-3 + 4\cos^2(2\sigma_m))]\}$
17: $\sigma' = \sigma$
18: $\sigma = s/b.A + \Delta\sigma$
19: $iterations+ = 1$ (incrementar)
20: **fin mientras**
21: **si** $iterations \geq 200$ **entonces**
22: **devolver** Error de conversión
23: **fin si**
24: $\varphi_2 = \text{atan2}(\sin(\psi_1)\cos(\sigma)\cos(\alpha_1), (1-f)\sqrt{\sin^2(\alpha) + (\sin(\psi_1)\sin(\sigma) - \cos(\psi_1)\cos(\sigma)\cos(\alpha_1))^2})$
25: $\lambda = \text{atan2}(\sin(\sigma)\sin(\alpha_1), \cos(\psi_1)\cos(\sigma) - \sin(\psi_1)\sin(\sigma)\cos(\alpha_1))$
26: $C = \frac{f}{16\cos^2(\alpha)[4+f(4-3\cos^2(\alpha))]}$
27: $L = \lambda - (1 - C).f.\sin(\alpha)\{\sigma + C.\sin(\sigma)[\cos(2\sigma_m) + C.\cos(\sigma)(-1 + 2\cos^2(2\sigma_m))]\}$
28: $\alpha_2 = \text{atan2}(\sin(\alpha), -\sin(\psi_1)\sin(\sigma) + \cos(\psi_1)\cos(\sigma)\cos(\alpha_1))$
29: **devolver** $p_2(\varphi_2, \lambda_1 + L)$

3.4. Conclusiones

En este capítulo se describió en que consiste el algoritmo de K-means, así como sus ventajas y desventajas, también sus variaciones aunque para este trabajo se utilizó la forma básica del algoritmo.

Se realizó la implementación del algoritmo en JavaScript y se mostró que el algoritmo funciona de manera correcta. Adicionalmente se integró el algoritmo desarrollado con el API de Google Maps, así como una primera prueba con datos reales. Dejando esto como referente para continuar con las demás implementaciones.

También en esta etapa se implementó para este proyecto el algoritmo desarrollado por Theodore Vincenty, y a partir de esta implementación se pueden calcular las coordenadas geográficas correspondientes a cada descarga registrada.

Capítulo 4

PRUEBAS Y RESULTADOS

4.1. DESCRIPCIÓN

Nuestra aplicación web está desarrollada con varias tecnologías como lo es NodeJS, el cuál se utilizo para desarrollar la aplicación servidor. Para la base de datos se utilizó MongoDB que es una manejador de base de datos no relacional o NoSQL y se clasifican por su forma de almacenar los datos en la categoría de bases de datos documentales. Y para la parte del cliente se utilizo el lenguaje de programación JavaScript, y también el API Google Maps quedando la estructura mostrada en la figura: 4.1

Como ya se menciona MogoDB almacena sus datos a manera de documentos que se encuentran en formato Json o nombrados por MongoDB como Bson, nuestros datos en este formato representados como documentos se pueden apreciar en la figura: 4.2

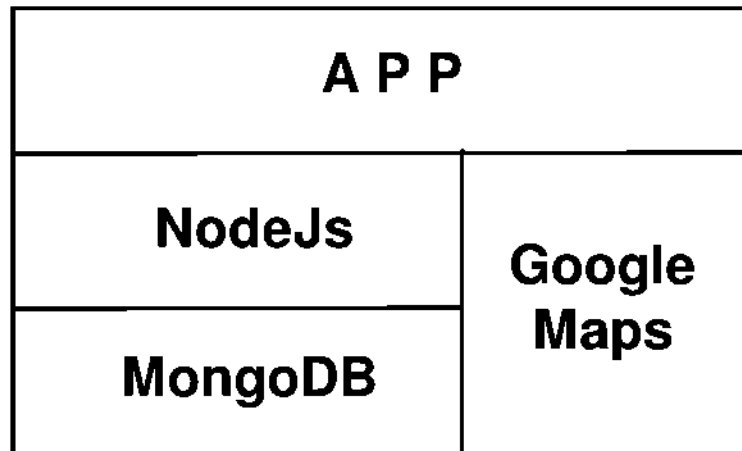


Figura 4.1: Estructura de la App

```
{
  "id" : ObjectId("561367be800e55380f263072"),
  "ciudad" : "Morelia",
  "fecha" : "2015-10-4",
  "hora" : 16,
  "minuto" : 47,
  "segundo" : 2,
  "bearing" : 274.8025711249259,
  "bearingRAW" : 274.8025711249259,
  "distance" : 400.659552,
  "distanceRAW" : 400.659552,
  "tipo" : "-CG"
}
```

Figura 4.2: Campos del documento en MongoDB

De igual manera una de las formas de manejo de datos en el API de Google Maps, como también se menciona antes, es por medio del formato Json. En

la figura: 4.3 se muestran los campos para cada objeto de JavaScript.

```
[{  
  "ciudad": "Morelia",  
  "fecha": "2012-5-1",  
  "hora": "0",  
  "minuto": "00",  
  "segundo": "1",  
  "lat": 25.95070007020186,  
  "lng": -103.52413389285701,  
  "tipo": "+CG",  
  "bearing": 341.5,  
  "dis": 733  
}]
```

Figura 4.3: Campos del objeto JavaScript en Json

Del API de Google se utilizaron varias herramientas como son: los marcadores, trayectorias, animación de marcadores, animación de trayectorias, dibujo de figuras geométricas, creación de iconos, incorporación imágenes como icono, etc.

Es importante mencionar que para las representaciones puntuales de las descargas atmosféricas se crea un objeto “marker”, de Google para cada descarga registrada por lo que hay que tener cuidado al elegir rangos de tiempo muy grandes que puedan contener demasiadas decenas de miles de documentos ya que, dependiendo de la capacidad (en cuestiones de hardware) de la computadora usada para utilizar la app tendrá un límite máximo para consultas grandes, aunque, será poco probable que se realice consultas de esa magnitud.

Para la aplicación se crearon iconos usando el API de Google. Los iconos creados fueron los siguientes:

- Nube: Cada nube representa el centroide de un cluster (una tormenta en un intervalo de tiempo), estos clusters formados por el algoritmo K-means en cada intervalo de tiempo, se puede ver el icono de nube en la figura: 4.4



Figura 4.4: Icono nube

- Rayo: Es la representación puntual para cada descarga atmosférica registrada, se puede ver en la figura: 4.5



Figura 4.5: Icono rayo

4.2. Ejecución de la aplicación Web

A continuación se muestran las pruebas realizados para diferentes casos:

- Caso 1: Para este ejemplo se tiene un archivo generado con un total de 19783 eventos de descargas atmosféricas con fecha 2015-10-30 que

simulan el comportamiento de dos tormentas; una del lado del océano pacifico y la otra del océano atlántico. En este ejemplo ya se conoce a priori el número de tormentas que generaron las descargas, lo anterior implica que ya se conoce el valor de K . Se procede a correr nuestra aplicación indicando el valor de k , el cual, para este caso es 2. Adicionalmente se introducen los valores para la fecha y hora de inicio, así como la ventana de tiempo (45 minutos). Como resultado de ejecutar la aplicación se obtiene el desplazamiento de cada uno de los centroides en el de tiempo (para cada ventana) , es decir como evolucionan las trayectorias de las tormentas. La trayectoria generada con la separación espacial por K-means y la temporal por nuestra ventana de tiempo se muestra en la figura 4.6

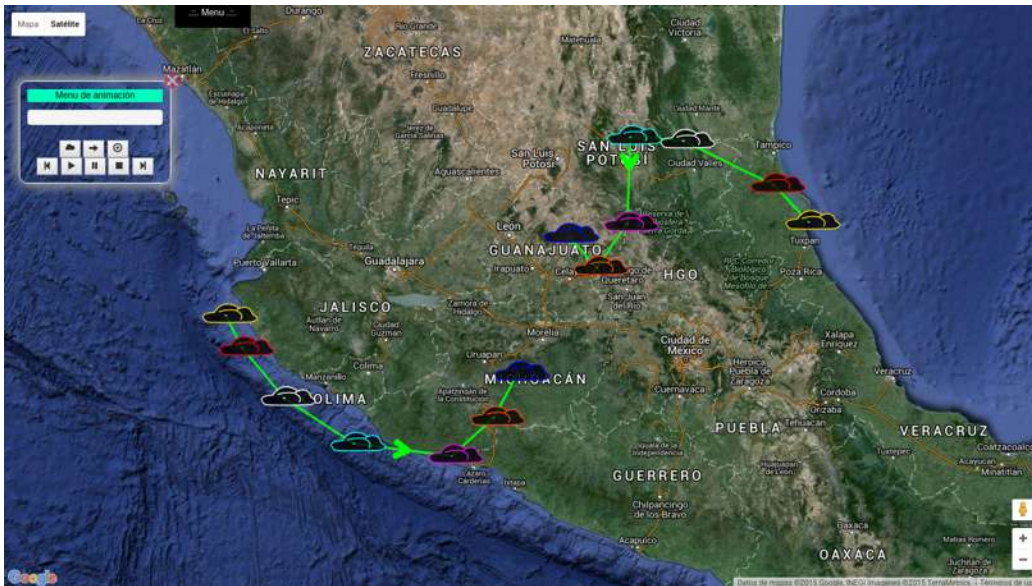
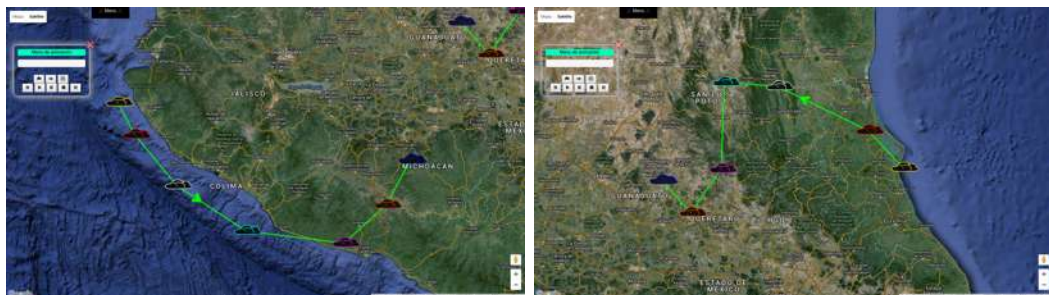


Figura 4.6: Trayectoria para dos tormentas

Como se puede observar el resultado muestra las dos tormentas esperadas, en la figura las nubes del mismo color indican que es el estado de la tormenta en el mismo intervalo de tiempo. También se observa que se tiene siete ventanas de tiempo. La trayectoria se representa en color verde y la flecha indica la dirección que lleva la tormenta. Una tormenta se mueve del océano pacifico hacia el centro del país y la segunda se mueve del océano atlántico también hacia el centro del país.

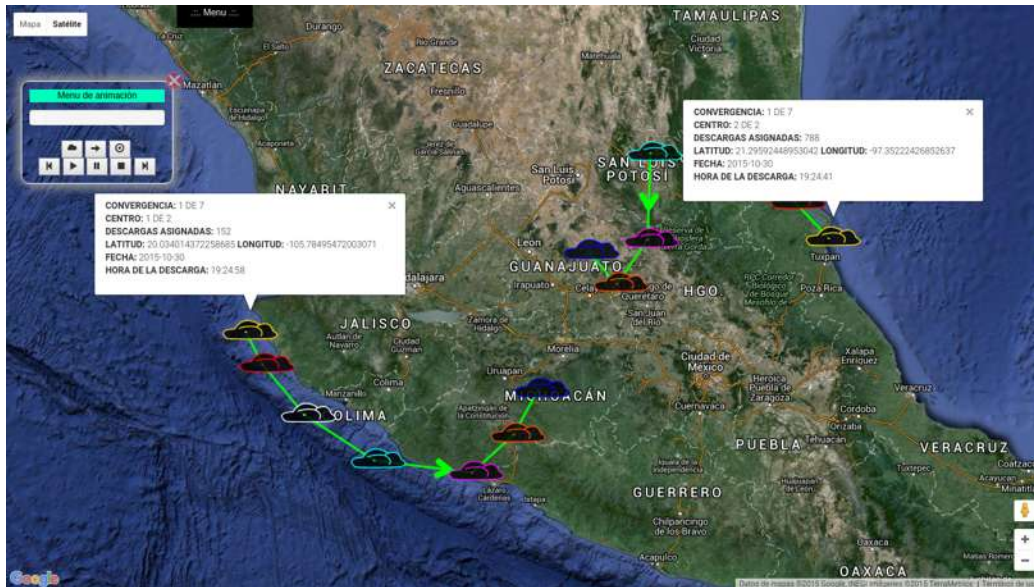
En las figuras 4.7a y 4.7b se observa un acercamiento a cada una de las trayectorias. En la figura 4.8a dentro de los cuadros de texto se encuentra la descripción para la primera de siete convergencias. En el centroide de la izquierda se puede ver que le fueron asignadas 152 descargas y al centroide de la derecha 788. En el recuadro también se especifica la latitud y longitud de estos, así como la fecha y hora en que se registro el evento. En la figura 4.8b se muestra la descripción para la ultima convergencia donde se asignan 633 y 472 descargas al centroide de lado izquierdo y derecho respectivamente.



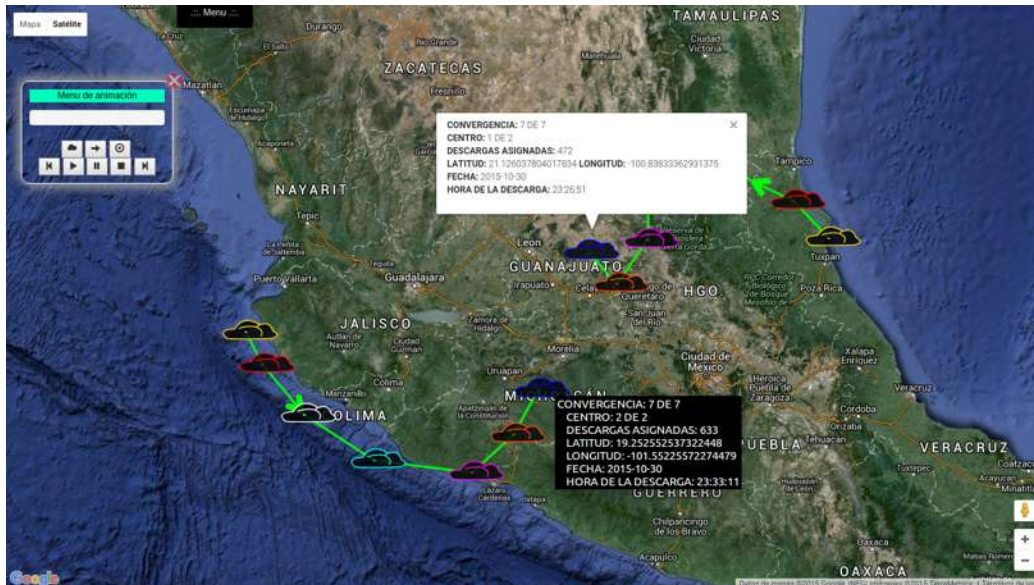
(a)

(b)

Figura 4.7: Trayectoria para dos tormentas



(a)



(b)

Figura 4.8: Trayectoria para dos tormentas con descripción

Se tiene también la opción para observar la representación puntual de las descargas para cada convergencia calculada por el algoritmo de K-means. Como se puede ver en las figuras 4.9 y 4.9 las descargas eléctricas de igual color fueron asignadas al mismo centroide, en este caso existen siete ventanas de tiempo por lo que se muestran las siete convergencias finales de estas ventanas.

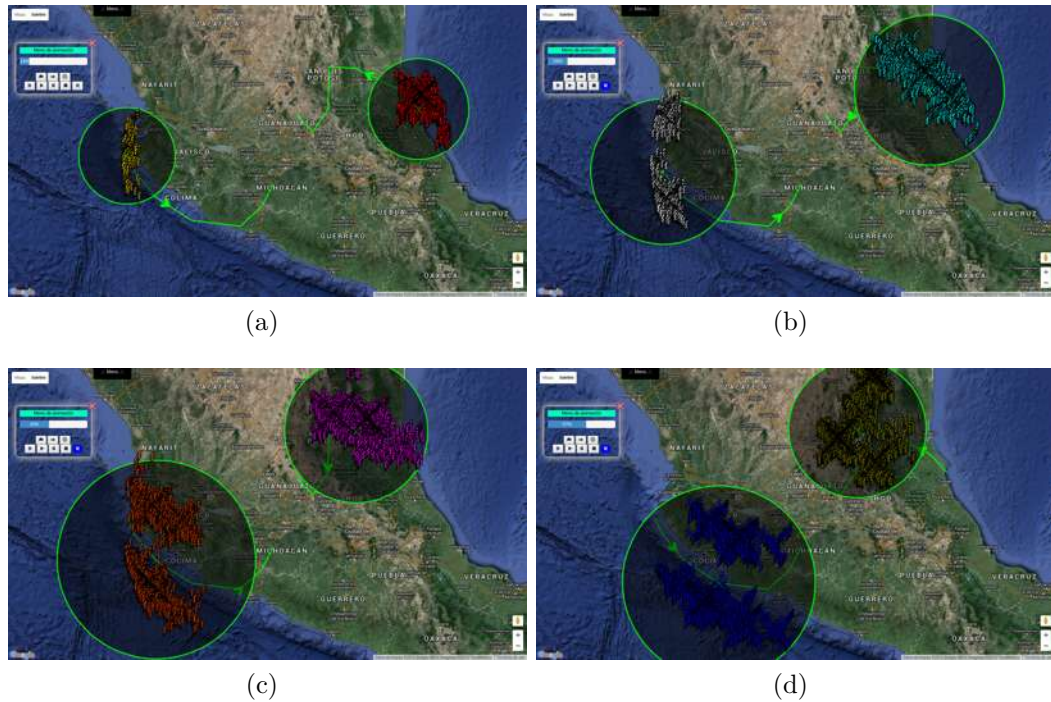


Figura 4.9: Representación puntual de las descargas en las trayectorias de dos tormentas para cada ventana de tiempo

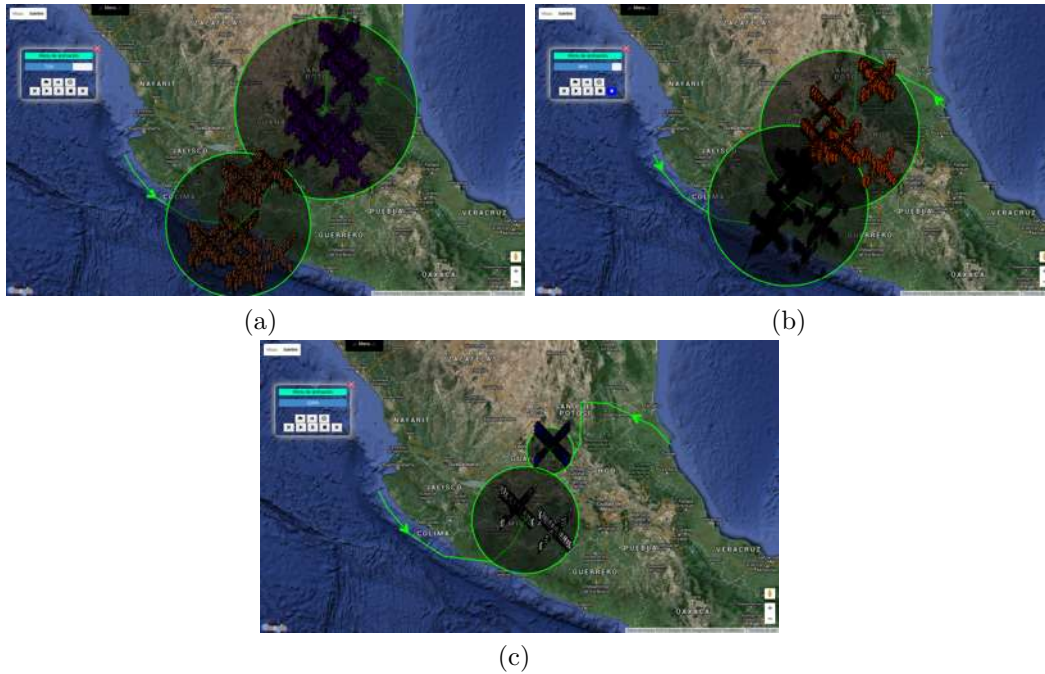


Figura 4.10: Representación puntual de las descargas en las trayectorias de dos tormentas

Como se ha mencionado es de mucha importancia elegir correctamente el valor de K , en la figura 4.11 se ejemplifica para el mismo caso una mala elección de este valor, se observa como el algoritmo asigna eventos de descarga para tres tormentas cuando en realidad existen dos. Los datos son separados pero no se obtendrá la trayectoria correcta.

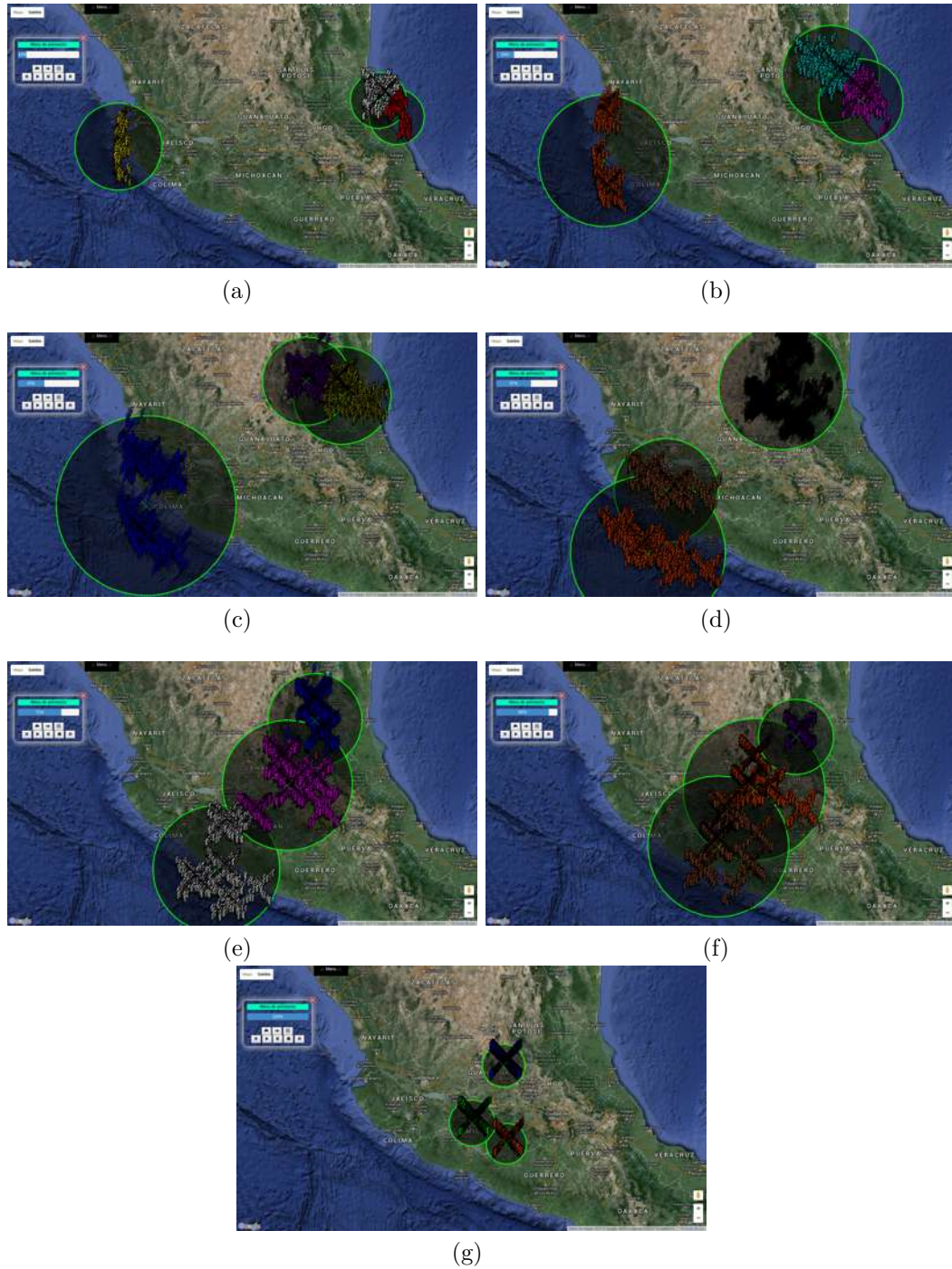


Figura 4.11: Representación puntual de las descargas en las trayectorias de dos tormentas con una mala elección de K

- Caso 2: Para este segundo ejemplo se corre el algoritmo para separar tres tormentas por lo que se elegirá el valor de K igual a tres, en la figura 4.12a se muestra el resultado para la fecha “2015-11-04”, con un tiempo de observación de “10:00”, a “12:30”, y separando en estampas de tiempo de 50 minutos.

se obtienen tres ventanas de tiempo con sus respectivas convergencias, en el inciso b se puede ver el cuadro de descripción para la primera ventana de tiempo en las tres tormentas identificadas las cuales contienen una asignación final de 1178, 1571 y 940 eventos de descarga,

4.3. Conclusiones

Se hace uso de la aplicación para realizar las pruebas donde se puede observar que el algoritmo categoriza de forma correcta los eventos de descargas atmosféricas, al igual que una correcta separación de las ventanas de tiempo. Por lo que la aplicación permite la representación geográfica de la ruta que siguen las tormentas en el tiempo de interés.

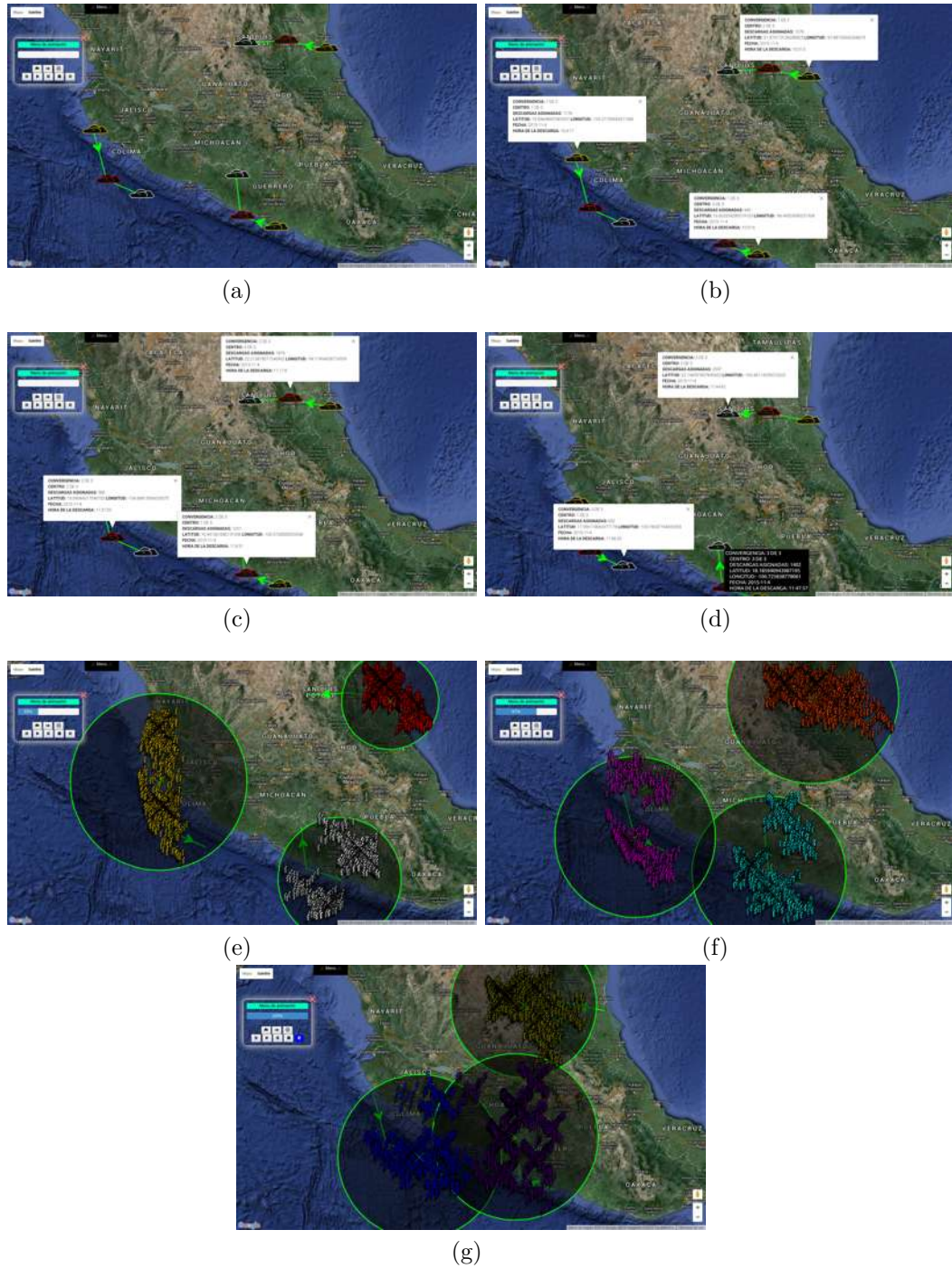


Figura 4.12: Trayectoria para tres tormentas con descripción

Capítulo 5

CONCLUSIONES

5.1. Conclusiones generales

En este trabajo se propusieron diversas tecnologías de desarrollo web, se llevó a cabo el ejercicio de la discusión para elegir las mas adecuadas para desarrollar el proyecto, donde surgieron suposiciones de las necesidades primarias que se tenían que cubrir.

El API de google ofrece una gran cantidad de herramientas para trabajar con mapas. Estas herramientas facilitaron el desarrollo de la aplicación implementada y cubrieron en su totalidad las necesidades para tal implementación.

Se utilizó el algoritmo de K-means para la clusterización de datos descriptivos de las descargas, esto con la finalidad de realizar una separación de las tormentas con respecto a su posición geográfica. La separación de datos en

ventanas y la ejecución de K-means para cada una de ellas permite, además de la clusterización, la obtención de la ruta o rutas que siguen las tormentas existente en el tiempo específico.

En las pruebas realizadas con la aplicación se pudo comprobar que se cumple con el objetivo del trabajo; el cual es poder visualizar como se agrupan las tormentas, sus trayectorias y la representación puntual de cada evento de descarga atmosférica en el mapa. También se puede ver en los resultados que la selección del número de K cambia totalmente el resultado obtenido.

5.2. Trabajo futuro

- Una propuesta es utilizar la observación de rayos en tormentas eléctricas para mejorar la predicción de la localización, intensidad y evolución temporal de precipitaciones y, de esta forma, prevenir afectaciones. Como lo es para el caso de la CFE que utilizaría el análisis probabilístico con el fin de tomar decisiones de valoración de riesgos referentes a la instalación de nuevas subestaciones o de equipo de transmisión, como las líneas de alto voltaje, capacitores, bancos de reactores o transformadores, etc.
- Otro trabajo futuro sería resolver la problemática de elegir el número de K correcto para cada caso. El algoritmo de K-means supone que se conoce el número de clusters y esto no es así, ya que el número de K

es una variante que elige el usuario y por tanto no es muy eficiente. Actualmente el usuario elige el número de K y enseguida se observan los resultados de esta elección. Los resultados observados son los que influyen para decidir si se aumenta o disminuye el número de centroides para la formación de los clusters. Entonces, la ayuda de algunos posibles criterios es buena propuesta para trabajar en una metodología que pueda identificar el número K conveniente y excluir al usuario de la decisión.

- Establecer la corrección correspondiente si en los registros de las diferentes antenas existe la posibilidad de que pertenezcan a el mismo evento atmosférico. Ya que se cuenta con la problemática de identificar si una descarga atmosférica es registrada sólo por una de las antenas o que la descarga haya sido registrada por alguna de las otras dos antenas instaladas actualmente.
- Probar con técnicas de estimación de la Función de densidad de probabilidad como el método de k -vecinos mas cercanos.
- Existe el interés en poder tener mapas georeferenciados con la localización de instalaciones eléctricas y poder observar en tiempo real el comportamiento de fenómenos atmosféricos que pudieran presentar un riesgo para la integridad de sus equipos.

Bibliografía

- [1] CAMBRONERO, C. G., AND MORENO, I. G. Algoritmos de aprendizaje: knn & kmeans. *Inteligencia en Redes de Comunicación, Universidad Carlos III de Madrid* (2006).
- [2] CIENCIORAMA, U. cienciorama, 2015.
- [3] DEVELOPERS, G. Google maps javascript api google developers, 2015.
- [4] ES.WIKIPEDIA.ORG. Formulas de vincenty, 2015.
- [5] ES.WIKIPEDIA.ORG. Google maps, 2015.
- [6] ES.WIKIPEDIA.ORG. Javascript, 2015.
- [7] ES.WIKIPEDIA.ORG. MongoDB, 2015.
- [8] ES.WIKIPEDIA.ORG. Node.js, 2015.
- [9] FOUNDATION, N. Node.js, 2015.
- [10] LÜCKEHEIDE, S., VELÁSQUEZ, J. D., AND CERDA, L. Segmentación de los contribuyentes que declaran iva aplicando herramientas de clustering. *Revista de Ingeniería de Sistemas 21* (2007), 87–110.

- [11] MONGODB.ORG. MongoDB for giant ideas mongodb, 2015.
- [12] PASCUAL, D., PLA, F., AND SÁNCHEZ, S. Algoritmos de agrupamiento. *Método Informáticos Avanzados* (2007).
- [13] PAÑAS, E. Observar los rayos para predecir las tormentas del mediterráneo, 2015.
- [14] VALENGA, F., FERNÁNDEZ, E., MERLINO, H., RODRÍGUEZ, D., PROCOPIO, C., BRITOS, P., AND GARCÍA-MARTÍNEZ, R. Minería de datos aplicada a la detección de patrones delictivos en argentina. In *IIISIC* (2008), pp. 31–40.
- [15] W3SCHOOLS.COM. Javascript tutorial, 2015.
- [16] YOLIS, E., BRITOS, P., PERICHISKY, G., AND GARCÍA-MARTÍNEZ, R. Algoritmos genéticos aplicados a la categorización automática de documentos. *Revista Eletrônica de Sistemas de Informação ISSN 1677-3071 doi: 10.5329/RESI 2, 2* (2003).