Desarrollo de herramientas de simulación basadas en imágenes satelitales con aplicaciones en salud, formativas y recreativas

María Carolina Montivero

28 de junio de 2009

Índice general

Índice de figuras

Capítulo 1

Motivación

Los avances de la tecnología espacial han brindado amplios beneficios a la salud humana. Entre ellos se encuentra la *Epidemiología Panorámica* que es una herramienta de vigilancia sanitaria que utiliza información espacial en la construcción de modelos predictivos de riesgo de enfermedades humanas, pues vincula las mismas con aspectos ambientales monitoreados por satélites. Este enfoque caracteriza las áreas ecogeográficas donde una enfermedad puede transmitirse y su principal objetivo es el desarrollo de *Sistemas de Alerta Temprana en Salud*.

La Epidemiología Panorámica se basa en la relación existente entre variables obtenidas de manera remota, como la humedad, vegetación, altitud, temperatura, uso del suelo, etc y los vectores de enfermedades virales. Si bien las imágenes satélites no permiten visualizarlos directamente, posibilitan la medición de factores climáticos y ambientales que determinan su comportamiento biológico e identifican sus hábitat.

La Comisión Nacional de Actividades Espaciales (CONAE) en cooperación con el Ministerio de Salud de la Nación están trabajando con el objetivo de desarrollar herramientas que a través del monitoreo de aspectos ambientales (desde satélites) sean capaces de predecir las distribuciones geográficas de los vectores para así implementar acciones de control. De la misma forma la CONAE, en cooperación con los ministerios de educación y de ciencia y tecnología, está trabajando en el desarrollo de programas tendientes a acercar el uso de la imágenes satelitales a niños y jóvenes entre 8 y 16 años a través de herramientas adecuadas.

En nuestro país, el mosquito de la fiebre amarilla (Aedes aegypti)[23] es un mosquito que puede ser convertirse en vector del virus del dengue y el de la fiebre amarilla, así como de otras enfermedades. En febrero de 2009, un brote de dengue tuvo origen en la provincia de Chaco, Argentina, que hasta 31 de marzo la epidemia alcanzó a tener diversos casos de muerte por esta enfermedad, causada por un virus y transmitida por mosquitos. El vector, Aedes aegypti, está presente en Buenos Aires, Catamarca, Chaco, Córdoba, Corrientes, Entre Ríos, Formosa, Jujuy, La Pampa, La Rioja, Mendoza, Misiones, Salta, Santa Fe, Santiago del Estesro y Tucumán.[22]

Por lo tanto, el presente proyecto pretende sumarse a tales trabajos, brindando una herramienta de software que coopere con estas acciones y resulte un medio educativo y

de interés científico sobre aplicaciones en salud.

Capítulo 2

Fundamentos de Teledetección

2.1. Introducción

La observación remota de la superficie terrestre constituye el marco de estudio de la *teledetección*. Este vocablo es una traducción latina del término inglés *remote sensing*, ideado a principios de los sesenta para designar cualquier medio de observación remota. La teledetección no engloba solo los procesos que permiten obtener una imágen, sino también su posterior tratamiento, en el contexto de una determinada aplicación. Un sistema de teledetección espacial incluye los siguientes elementos:

1. **Fuente de energía**: que supone el origen de la radiación electro-magnética que detecta el sensor. Puede tratarse de un foco extremo a este, en cuyo caso se habla de teledetección pasiva, o de un haz energético emitido por el sensor, en cuyo caso se habla de teledetección activa. La fuente de energía más importante es el sol.

2. **Cubierta terrestre**: formada por distintas masas de vegetación, suelos, agua o construcciones humanas, que reciben la señal energética procedente de (1) y la reflejan o emiten de acuerdo a sus características físicas.

3. **Sistema sensor**: compuesto por el sensor, propiamente dicho, y la plataforma que lo alberga. Tiene como misión captar la energía procedente de las cubiertas terrestres, codificarla y grabarla o enviarla directamente al sistema de recepción.

4. **Sistema de recepción-comercialización**: en donde se recibe la información transmitida por la plataforma, se graba en un formato apropiado y tras las oportunas correcciones, se distribuye a los interesados.

5. **Intérprete**: que convierte esos datos en información temática de interés, ya sea visual o digitalmente, para facilitar la evaluación del problema en estudio.

6. **Usuario final**: encargado de analizar el documento fruto de la interpretación así como de dictaminar sobre las consecuencias que de el deriven.

2.2. Espectro electromagnético

Aunque la sucesión de valores de longitud de onda es continua, suelen establecerse una serie de bandas en donde la radiación electromagnética manifiesta un comportamiento similar. La organización de estas bandas de longitudes de onda se denomina *espectro electromagnético*.
Desde el punto de vista de la teledetección, conviene destacar una serie de bandas espectrales, que son las mas frecuentemente empleadas con la tecnología actual:

- **Espectro Visible (0,4 a 0,7 μm)**: se denomina así por tratarse de la única radiación electromagnética que pueden percibir nuestros ojos. Dentro de esta región, suelen distinguirse tres bandas elementales, que se denominan **Azul** (0,4-0,5 μm), **Verde** (0,5-0,6 μm) y **Rojo** (0,6-0,7 μm)

- **Infrarrojo Cercano (IRC 0,7-1,3 μm)**: también se denomina infrarrojo próximo, reflejado o fotográfico. Resulta de especial importancia por su capacidad para discriminar masas vegetales y concentraciones de humedad.

- **Infrarrojo Medio (1,3-8 μm)**: en esta región se entremezclan los procesos de reflexión de la luz solar y de emisión de la superficie terrestre. La primera banda se sitúa entre 1,3 y 2,5 μm y se denomina *infrarrojo de onda corta* (Short wave infrared SWIR), que resulta idónea para estimar el contenido de humedad en la vegetación o los suelos. La segunda, comprendida principalmente en torno a 3,7 μm se conoce propiamente como *infrarrojo medio* (IRM), siendo determinante para la detección de focos de alta temperatura, como incendios o volcanes activos.

- **Infrarrojo Lejano o Térmico (IRT, 8-14 μm)**: incluye la porción emisora del espectro terrestre, en donde se detecta el calor proveniente de la mayor parte de las cubiertas terrestres.

- **Micro-ondas (M, por encima de 1 mm)**: son de gran interés por ser un tipo de energía bastante transparente a la cubierta nubosa.

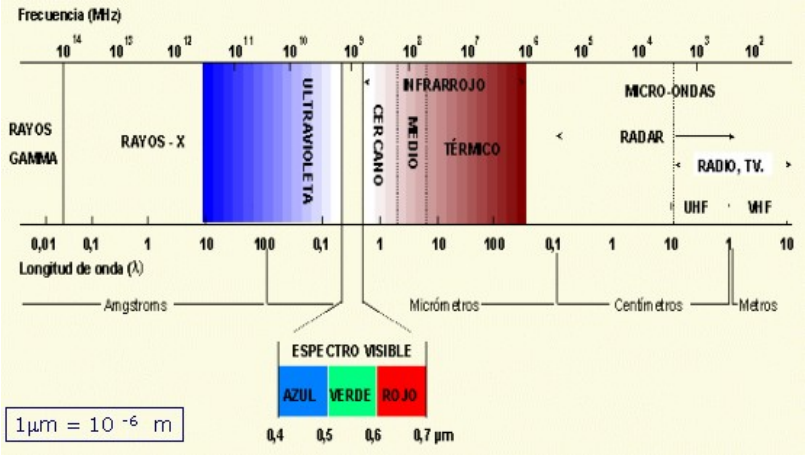

Figura 2.1: Espectro electromagnético
[26]

2.3. Satélites

2.3.1. Órbitas satelitates

Los satélites pueden operar con diferentes clases de órbitas terrestres. Las órbitas más comunes para satélites de medio ambiente son las geoestacionarias y las polares, pero algunos instrumentos también vuelan en órbitas inclinadas. Es posible tener otra clase de órbitas, como las órbitas de Molniya que se usan comúnmente para satélites soviéticos.

- **Órbitas Geoestacionarias:** Una órbita geoestacionaria [1] es aquella en la que el satélite siempre esta en la misma posición con respecto a la Tierra, que rota. El satélite orbita a una altura de aproximadamente 35790 Km. porque esto hace que el periodo orbital[2] sea igual al periodo de rotación de la Tierra (23h 56m 4.09s). Al orbitar al mismo ritmo y en la misma dirección que la Tierra, el satélite esta estacionario, es decir, está sincronizado con respecto a la rotación de la Tierra. Los satélites geoestacionarios proporcionan un panorama de observación muy grande permitiendo cubrir los sucesos relacionados con el tiempo. Esto es especialmente útil para observar tormentas locales severas y ciclones tropicales.

- **Órbitas Polares:** Los satélites que vuelan en órbitas polares proporcionan una visión más global de la Tierra, girando con una inclinación[3] cerca de la polar[4]. Orbitando a una altura de 700 u 800 Km., estos satélites cubren de mejor forma las partes del mundo mas difíciles de cubrir in situ. Estos satélites operan en una

[1] GEO = geosincronizada
[2] Período orbital es la duración de una órbita
[3] ángulo entre el plano ecuatorial y el plano de la órbita del satélite
[4] NOTA: una verdadera órbita polar tendría una inclinación de 90 grados

órbita sincronizada con el sol. El satélite pasa cada día el ecuador y cada latitud a la misma hora solar local, lo cual quiere decir que el satélite pasa por encima de nuestras cabezas a la misma hora solar a lo largo de todas las estaciones del año. Esta característica permite recolección regular de datos en horas consistentes así como comparaciones a largo plazo. El plano orbital de una órbita sincronizada con el sol debe también rotar aproximadamente un grado al día para mantenerse con respecto a la Tierra.

- **Órbitas inclinadas:** Las órbitas inclinadas están entre las dos anteriores. Tienen una inclinación entre 0 grados (órbita ecuatorial) y 90 grados (órbita polar). Estas órbitas pueden estar determinadas por la región de la Tierra que es de mayor interés. La altura de la órbita de estos satélites generalmente es del orden de unos cientos de kilómetros por lo que el periodo orbital es del orden de unas cuantas horas.

2.3.2. Landsat TM 5/7

Los Landsat son una serie de satélites construidos y puestos en órbita por EE. UU. para la observación en alta resolución de la superficie terrestre.[5]

Orbitan alrededor de la Tierra en órbita circular heliosincrónica[6], a 705 km de altura, con una inclinación de 98.2º respecto del Ecuador y un período de 99 minutos. La órbita de los satélites está diseñada de tal modo que cada vez que éstos cruzan el Ecuador lo hacen de Norte a Sur entre las 10:00 y las 10:15 de la mañana hora local. Los LandSat están equipados con instrumentos específicos para la teledetección multiespectral.

El primer satélite LandSat que en principio fué denominado ERTS-1, se lanzó el 23 de julio de 1972. El último de la serie es el LandSat 7, puesto en órbita en 1999, y es capaz de conseguir una resolución espacial de 15 metros. En el año 1985 fue puesto en orbita el Landsat 5 cuyo programa es financiado por el gobierno de los Estados Unidos y operado por la NASA[7] y la USGS[8].

Este satélite esta dotado de cámaras que proporcionan imágenes de gran resolución y de un sistema de sensores de diversa longitud de onda que registran la radiación electromagnética emitida/reflejada desde el suelo, con lo cual es posible poner en evidencia detalles de la superficie terrestre que de otra manera serían invisibles. Este explorador de barrido, denominado Thematic Mapper (TM) esta diseñado para la cartografía temática. Este satélite tiene una resolución espacial[9] de 30m, resolución espectral[10] de

[5] http://landsat.gsfc.nasa.gov/about/

[6] Órbita circular heliosincrónica significa que tiene visión de toda la superficie terrestre, en un tiempo de 15 días, realizando 232 orbitas

[7] National Space and Space Administration

[8] United States Geological Survey

[9] Este concepto designa al objeto más pequeño que se puede distinguir en la imágen. Está determinada por el tamaño del píxel, medido en metros sobre el terreno, esto depende de la altura del sensor con respecto a la Tierra, el ángulo de visión, la velocidad de escaneado y las características ópticas del sensor.

[10] Consiste en el número de canales espectrales (y su ancho de banda) que es capaz de captar un

7 bandas y resolución radiométrica [11] de 8 bits.

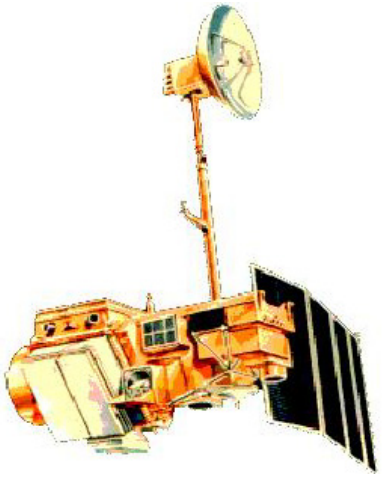

Figura 2.2: LandSAT 5
[27]

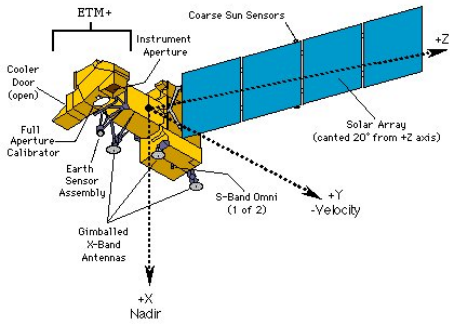

Figura 2.3: LandSAT 7
[28]

sensor.

[11] Se la llama a veces también resolución dinámica, y se refiere a la cantidad de niveles de gris en que se divide la radiación recibida para ser almacenada y procesada posteriormente.

2.3.3. SPOT

Los SPOT son satélites artificiales desarrollados por el Centro Nacional de Estudios Espaciales francés (CNES) en colaboración con Bélgica y Suecia y fabricados por EADS Astrium. El primero de ellos se lanzó en 1986 y en la actualidad el sistema SPOT[12] opera con una constelación de cuatro satélites de observación: SPOT 1, SPOT 2, SPOT 4 y SPOT 5. Todos ellos en órbita polar, circular y heliosincrónica sobre la Tierra.

La altura orbital es de 822 km, con una inclinación de 98º y un período orbital de 101 minutos. La resolución temporal[13] es de 26 días. Una de las ventajas con que cuentan los satélites SPOT es la posibilidad de observación no vertical de su sensor HRV [14] en SPOT 2, HRVIR [15] en SPOT 4 y HRG [16] en SPOT 5. HRV, HRVIR y HRG son lo que se denominan sensores enfocables que permiten adquirir datos de zonas fuera de su órbita mediante el movimiento, de hasta 27º a ambos lados del nadir [17], de un dispositivo instalado en el equipo óptico.

También en los SPOT 4 y 5 se incorporó VEGETATION, de observación terrestre de amplio campo y alta resolución radiométrica. Esta serie de satélites tienen como misión el seguimiento de la vegetación, organización de la superficie habitable, cartografía plana y restitución del relieve y vigilancia ecológica.

[12]Satellite Probatoire pour l'Observation de la Terre

[13]Es la frecuencia de paso del satélite por una mismo punto de la superficie terrestre. Es decir cada cuanto tiempo pasa el satélite por la misma zona de la Tierra. Este tipo de resolución depende básicamente de las características de la órbita.

[14]Alta resolución visible

[15]Alta Resolución Visible Infrarroja

[16]Alta Resolución Geométrica

[17]En astronomía se denomina nadir .ºpuesto.ª la intersección entre la vertical del observador y la esfera celeste. Es decir: si imaginamos una recta que pasa por el centro de la Tierra y por nuestra ubicación en su superficie, el nadir se encuentra sobre esa recta, por debajo de nuestros pies.

Figura 2.4: SPOT 5
[29]

2.3.4. ENVISAT

El satélite Envisat (Environmental Satellite) es un satélite de observación terrestre construido por la Agencia Espacial Europea (ESA). Fue lanzado el 1 de marzo de 2002 en un cohete Ariane 5 en una órbita polar síncrona con el Sol a una altura de 790 km aproximadamente. Orbita la Tierra en un periodo de cerca de 101 minutos con un periodo de repetición de ciclos cada 35 días.

El Envisat es el mayor observatorio de la atmósfera y superficie terrestre lanzado hasta la fecha (junio 2005) disponiendo de 9 instrumentos para obtener información sobre la superficie de la Tierra, los océanos y la atmósfera.

- **ASAR**: Advanced Synthetic Aperture Radar. Opera en banda C y puede detectar cambios de altura en superficies con presición sub-milimétrica
- **MERIS**: Medium Resolution Imaging Spectrometer. Mide la reflectancia de la Tierra (la superficie y la atmósfera) en el rango espectral solar (390 a 1040 nm) y transmite 15 bandas espectrales al segmento terrestre.
- **AATSR**: Advanced Along Track Scanning Radiometer. Puede medir la temperatura de la superficie del mar

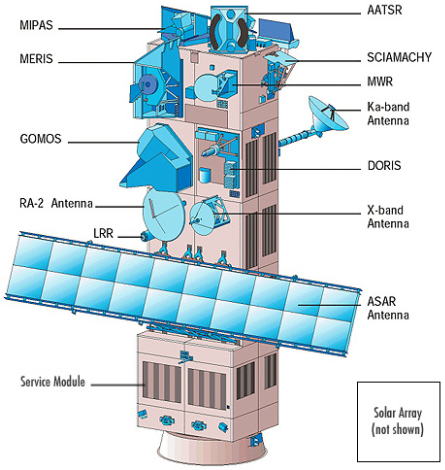

Figura 2.5: ENVISAT
[30]

2.3.5. SAC-C

La serie SAC se ha desarrollado por la **CONAE** junto con la empresa argentina **INVAP** y una serie de universidades locales, en estrecha cooperación con la **NASA**. También ha incluido la participación de Brasil, Dinamarca, Italia y Alemania. Actualmente tres satélites argentinos se han puesto en marcha: **SAC-B**, que se dedicó a la investigación científica; **SAC-A**, un satélite de demostración tecnológica, y **SAC-C**, el primer satélite de observación de la Tierra de Argentina. El satélite SAC-C fue puesto en órbita el 21 de noviembre de 2000 (cohete Delta) y actualmente continúa en funcionamiento. Lanzado como parte de la Constelación de AM junto con la NASA Landsat 7, Terra y EO-1. Lleva sensor MMRS (Multispectral mid resolution sensor), una HRTC (High Resolution technological camera), una HSTC (high sensitivity camera) y un GPS GOLPE (GPS Occultation and Passive reflection experiment). Todos los instrumentos fueron provistos por la **CONAE**, salvo GOLPE, que fue facilitada por la **NASA**

Figura 2.6: SAC-C
[31]

2.4. Procesamiento de Imágenes

2.4.1. Filtros

Los filtros se aplican en el análisis digital para aislar componentes de interés. Mediante técnicas de filtraje se pretende suavizar o reforzar estos contrastes espaciales, de tal forma que los niveles digitales (ND) de la imágen se asemejen o diferencien más de los correspondientes a los píxeles que los rodean.

Con el filtrado se pretende mejorar la visualización de las imágenes, ya sea para eliminar valores anómalos o para resaltar rasgos lineales de interés. Se suelen distinguir dos tipos de filtros de acuerdo al objeto que se persiga: filtros de **paso bajo** y los de **paso alto**

- **Filtro de paso bajo**: tienden a destacar el componente de homogeneidad en la imágen, subrayando aquellas áreas donde la frecuencia de cambio es baja. Tienen por objetivo suavizar los contrastes espaciales presentes en la imágen. Se trata de semejar el ND de cada píxel al de los píxeles vecinos, reduciendo la variabilidad espacial de la escena. Este tipo de filtraje se utiliza para restaurar los errores aleatorios que pueden presentarse en los ND de la imágen, fruto de un defecto en la adquisición o recepción de los datos. El filtro de paso bajo puede obtenerse a partir de diversas matrices de filtrado. Algunas de las más habituales son

1. 1.00 1.00 1.00
   1.00 1.00 1.00
   1.00 1.00 1.00

2. 1.00 1.00 1.00
   1.00 2.00 1.00
   1.00 1.00 1.00

3. 0.25 0.50 0.25
   0.50 1.00 0.50
   0.25 0.50 0.25

La primera es un simple promedio de los 9 píxeles que componen la ventana de filtraje, mientras las dos siguientes pondera un poco el valor centrar para evitar una excesiva pérdida de detalle.

- **Filtro de paso alto**: pretende aislar los componentes de alta frecuencia en una imágen. En términos espaciales, eso supone remarcar digitalmente los contrastes espaciales entre píxeles vecinos, enfatizando los rasgos lineales presentes en la imágen, como carreteras, parcelas o accidentes geológicos. Se intenta reforzar los contornos entre áreas homogéneas, evidenciando cualquier discontinuidad. Varios métodos pueden conseguir este objetivo. El más sencillo pasa por restar de la imágen original la obtenida por un filtro de paso bajo. Si lo que se pretende es aislar los componentes de alta frecuencia (alta variabilidad), basta restar de la imágen original aquellos correspondientes a la baja frecuencia y sólo quedarían los requeridos que se añadirían en la imágen original. Más común es, sin embargo, emplear matrices de filtraje similares a las anteriores. En este caso, los coeficientes de filtraje (CF) se disponen de tal modo que se refuerza el contraste entre el píxel central y los vecinos. Dos matrices frecuentemente utilizadas son:

  1. -1.00 -1.00 -1.00
     -1.00 9.00 -1.00
     -1.00 -1.00 -1.00

  2. 0.00 -1.00 0.00
     -1.00 4.00 -1.00
     0.00 -1.00 0.00

  Ambas derivan del análisis de gradientes de cambio en la imágen. El segundo filtro, denominado Laplaciano, se recomienda para el realce de rasgos lineales en la ciudad.

2.4.2. Textura y Bordes

Una de las principales ventajas del análisis visual sobre el digital es la capacidad de incorporar a la interpretación de la imágen criterios complejos. En el análisis visual se pueden utilizar otros elementos, como por ejemplo la textura y los bordes. Esta variable hace referencia al contraste espacial entre los elementos que componen una determinada cubierta. Visualmente se manifiesta como la rugosidad o suavidad de los tonos gris. Cuanto mas similares sean, la tonalidad será más homogénea en el interior de la cubierta y la textura será más lisa. Por el contrario, si existe una alta heterogeneidad en los niveles de gris de esa cubierta aparecerá como muy rugosa, con textura grosera. La textura de una cubierta procede de la relación entre el tamaño de los objetos que la forman y la resolución del sensor. En función del tamaño de los objetos que forman una cubierta suelen distinguirse tres tipos de textura:

1. **Textura grosera**: cuando los objetos están comprendidos entre 0,25 y 1 mm^2 a la escala de la imágen.

2. **Textura media**: objetos comprendidos entre 0,04 y 0,25 mm^2.

3. **Textura fina**: objetos inferiores a 0,04 mm^2.

2.4.3. Clasificación

Para que una fotografía aérea pueda ser interpretada en detalle es preciso que exista una experiencia previa, que permita identificar cada una de las categorías de interés por una serie de rasgos, como son tono, textura situación o tamaño. La clasificación digital se inicia caracterizando los patrones que definen en la imágen distintas categorías objetivo. Por cuanto se trata de una clasificación basada en los valores numéricos de los píxeles, esta caracterización también debe ser numérica; esto es, se trata de obtener el ND, o mejor aún el rango de ND que identifica a cada categoría, para todas las bandas que intervienen en la clasificación. En el proceso de clasificación se trata de definir con rigor cada una de las categorías que pretenden discriminarse, teniendo en cuenta su propia variabilidad en la zona de estudio. Este objetivo se logra seleccionando una muestra de píxeles de la imágen, que representen adecuadamente a las categorías de interés. A partir de esos píxeles pueden calcularse los ND medios y la variabilidad numérica de cada categoría, en todas las bandas que intervienen. Al igual que en otras aplicaciones del muestreo, las estimaciones posteriores se basan sobre la muestra seleccionada, por lo que una incorrecta selección de ésta conducirá inexorablemente a pobres resultados en la clasificación posterior. Se han dividido los métodos de clasificación en dos grupos: **supervisado** y **no supervisado** Dentro de cada una de estos métodos suelen distinguirse dos clases que pueden intervenir: **informacionales** y **espectrales**. Las primeros son las que constituyen la leyenda de trabajo que pretende deducir el intérprete, como por ejemplo tipos de ocupación de suelo y las segundas corresponden a los grupos de valores espectrales homogéneos dentro de la imágen en función de ofre-

cer una reflectividad similar para las bandas consideradas y en la fecha concreta de la imágen.

- **Método supervisado**: parte de un cierto conocimiento de la zona de estudio, adquirido por experiencia previa. Esta mayor familiaridad con el área de interés permite al intérprete delimitar sobre la imágen unas áreas suficientemente representativas de cada una de las categorías que componen la leyenda. A estas áreas se las denomina **training fields**. A partir de ellas se caracterizan cada una de las clases, para asignar más tarde el resto de los píxeles de la imágen a una de esas categorías en función de la similitud de sus ND con los extraídos como referencia. Acabada la delimitación de un training field, pueden seleccionarse otras para la misma categoría, o bien culminar con ella la definición de esa clase. En general, resulta conveniente seleccionar varias áreas por categoría, a fin de reflejar adecuadamente su variabilidad en la zona de estudio. Finalizada la selección, se calculas las estadísticas elementales de cada categoría: media, rango, desviación típica, matriz de varianza-covarianza, etc, a partir de los ND de todos los píxeles incluidos en los training fields, aplicando este cálculo a todas las bandas que intervengan en la clasificación. Se requiere seleccionar un mínimo de *m+1* píxeles por categoría, siendo *m* el número de bandas que integran el análisis.

- **Método no supervisado**: se dirige a definir las clases espectrales presentes en la imágen. No implica ningún conocimiento del área de estudio, por lo que la intervención humana se centra más en la interpretación que en la consecución de los resultados. En esta estrategia se asume que los ND de la imágen forman una serie de agrupaciones o conglomerados *(clusters)*, más o menos nítidos según los casos. Estos grupos equivaldrían a píxeles con un comportamiento espectral homogéneo y, por tanto, debería definir clases temáticas de interés. El método para definir los agrupamientos espectrales es muy similar al empleado en otras técnicas de clasificación automática de datos. Se basa en la selección de tres parámetros: variables que intervienen en el análisis; cirterio para medir la similitud o distancia entre casos y criterio para agrupar casos similares.

2.4.4. NDVI

El NDVI (Normalized Difference Vegetation Index) representa la distribución de la vegetación. Es un índice usado para estimar la cantidad, calidad y desarrollo de la vegetación en base a la medición, por medio de sensores remotos instalados comúnmente desde una plataforma espacial, de la intensidad de la radiación de ciertas bandas del espectro electromagnético que la vegetación emite o refleja. El índice diferencial de vegetación normalizado, NDVI, se calcula a partir de estas medidas individuales de la siguiente manera:

$$NDVI = \frac{IRC-R}{IRC+R}$$

en donde las variables **IRC** y **R** están definidas por las medidas de reflexión espectral adquiridas en las regiones del rojo e infrarrojo cercano, respectivamente.

2.4.5. Transformación Tasseled Cap

Esta transformación se dirige a obtener unas nuevas bandas por combinación lineal de las originales, con el objeto de realzar algunos rasgos de interés en la escena. Pone en más evidencia el comportamiento espectral de la vegetación y el suelo.
Los componentes deducidos a través de este tipo de transformación tienen un significado preciso, independientemente de las condiciones de la escena, puesto que se apoyan sobre las características del sensor y no sobre la radiometría de la imagen. Por ejemplo, algunos componentes que se distinguen en una imagen LandSat-TM son:

- Brillo (Brightness)
- Verdor (Greeness)
- Humedad (Wetness)

Capítulo 3

Requerimientos

3.1. Introducción a UML y StarUML

3.1.1. UML

UML (Unified Modeling Language) es el lenguaje de modelado de sistemas de software orientado a objetos más conocido y utilizado en la actualidad; está respaldado por el OMG [1] (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema.
UML ofrece un estándar para describir un plano del sistema (modelo), incluyendo aspectos conceptuales tales como funciones del sistema, y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. En UML 2.0 hay 13 tipos diferentes de diagramas divididos en 3 grupos [24]:

- Diagramas de estructura
- Diagramas de comportamiento
- Diagramas de interacción

Los **diagramas de estructura** enfatizan los elementos que deben existir en el sistema modelado. En el presente trabajo final, los que utilizaremos serán:

- **Diagrama de clases:** es un tipo de diagrama estático que describe la estructura de un sistema mostrando sus clases, atributos y las relaciones entre ellos. Los diagramas de clases son utilizados durante el proceso de análisis y diseño de los sistemas, donde se crea el diseño conceptual de la información que se manejará en el sistema, y los componentes que se encargaran del funcionamiento y la relación entre uno y otro.

[1] El Object Management Group u OMG (de sus siglas en inglés Grupo de Gestión de Objetos) es un consorcio dedicado al cuidado y el establecimiento de diversos estándares de tecnologías orientadas a objetos, tales como UML, XMI, CORBA. Es una organización sin ánimo de lucro que promueve el uso de tecnología orientada a objetos mediante guías y especificaciones para las mismas

- **Diagrama de componentes:** Un diagrama de componentes representa cómo un sistema de software es dividido en componentes y muestra las dependencias entre estos componentes. Los componentes físicos incluyen archivos, cabeceras, librerías compartidas, módulos, ejecutables, o paquetes. Debido a que estos son más parecidos a los diagramas de casos de usos estos son utilizados para modelar la vista estática y dinamica de un sistema. Muestra la organización y las dependencias entre un conjunto de componentes. No es necesario que un diagrama incluya todos los componentes del sistema, normalmente se realizan por partes. Cada diagrama describe un apartado del sistema. Uno de los usos principales es que puede servir para ver qué componentes pueden compartirse entre sistemas o entre diferentes partes de un sistema.

Los **diagramas de comportamiento** enfatizan lo que debe suceder en el sistema modelado. En el presente trabajo final, los que utilizaremos serán:

- **Diagrama de casos de uso:** muestra la funcionalidad provista por el sistema en términos de sus actores, los objetivos y las dependencias entre ellos.

Los **diagramas de interacción** son un subtipo de diagramas de comportamiento que enfatiza el flujo de control y datos entre los elementos del sistema modelado. En el presente trabajo final, los que utilizaremos serán:

- **Diagrama de secuencia:** muestra la interacción de un conjunto de objetos en una aplicación a través del tiempo y se modela para cada método de la clase.

StarUML es una herramienta para el modelado de sistemas que soporta la notación UML. Se utilizará esta herramienta para el desarrollo de los diagramas presentados en este trabajo final. Ver [4] para más referencias.

3.2. Requerimientos del sistema

3.2.1. Requerimientos no funcionales

Un Requerimiento no funcional es un requerimiento que especifica criterios que pueden usarse para juzgar la operación de un sistema en lugar de sus comportamientos específicos, ya que éstos corresponden a los requerimientos funcionales [2]. Los requerimientos no funcionales incluyen restricciones y cuestiones de calidad de la herramienta, como por ejemplo:

1. Dedicado a una problemática de salud.
2. Proveer herramientas para seleccionar distintos algoritmos.
3. Ser multiplataforma.
4. Ser mantenible y extensible.

3.2.2. Requerimientos funcionales

Un *requerimiento funcional* define el comportamiento interno del software: cálculos, detalles técnicos, manipulación de datos y otras funcionalidades específicas que muestran cómo los casos de uso serán llevados a la práctica. Son complementados por los *requerimientos no funcionales*. A continuación, presentaremos algunos requerimientos funcionales en lenguaje coloquial:

1. El algoritmo que se implementará está dado por la ecuación 4.6.

2. Se debe notificar al usuario si alguno de los campos de configuración fué dejado en blanco por error o si la cantidad de horas para la simulación no fue ingresada.

3. El sistema debe contar con un método de graficación para poder desplegar los resultados de la simulación.

4. Al finalizar la ejecución del algoritmo, se mostrará un mensaje de éxito o error, ya sea si el algoritmo termina de forma exitosa o si termina con alguna falla.

5. El usuario debe poder elegir los nombres de los archivos, la cantidad de horas, el tipo de diferencia finita y el directorio donde se almacenarán los resultados para cada simulación.

6. Cada uno de los algoritmos debe poseer una interfaz gráfica de modo que pueda ser configurado por el usuario.

7. Cada algoritmo debe tener una *configuración por defecto*, esto es un conjunto de valores precargados para facilitar su uso.

8. El sistema debe ser capaz de leer y almacenar datos provenientes de imágenes satelitales o imágenes que sean resultado de un procesamiento con ENVI.

9. Los resultados de la ejecución deben mostrarse al usuario de forma gráfica.

10. El usuario debe poder seleccionar desde el menú un archivo de configuración dado.

11. El usuario debe poder guardar una configuración dada.

Matriz de identificación de requerimientos

A continuación, expresaremos algunos de los requerimientos dados anteriormente con una matriz de trazabilidad:

Código	Descripción
R_0	El algoritmo que se implementará está dado por la ecuación 4.6
R_1	Se debe notificar al usuario si alguno de los campos de configuración fué dejado en blanco.
R_2	El sistema debe contar con un método de graficación para poder desplegar los resultados de la simulación.
R_3	Se debe notificar al usuario cuando la simulación haya finalizado.
R_4	El usuario debe poder elegir los nombres de los archivos, la cantidad de horas, el tipo de diferencia finita y el directorio donde se almacenarán los resultados para cada simulación.

Código	Descripción
R_0_0	Crear un filtro que represente al fenómeno de transporte dado por la ecuación 4.1.
R_0_1	Crear un filtro que represente al fenómeno de rugosidad del paisaje dado por la ecuación 4.3
R_0_2	Crear un filtro que represente al fenómeno de fuerzas atractivas y repulsivas dado por la ecuación 4.2
R_0_3	Crear un filtro que represente a los nacimientos dados por la ecuación 4.4
R_0_3	Crear un filtro que represente a las muertes dadas por la ecuación 4.5

Código	Descripción
R_0_0_0	Para representar las derivadas parciales utilizar el método de diferencias finitas dado por la ecuación 4.8 o la ecuación 4.10 o la ecuación 4.12

Código	Descripción
R_1_0	Antes de comenzar la ejecución del algoritmo, se verificará que todos los campos no estén vacíos y que contengan nombres de archivos existentes.
R_1_1	Antes de comenzar la ejecución del algoritmo, se verificará que el tiempo seleccionado para la simulación sea mayor que cero.

Código	Descripción
R_2_0	Realizar lectura de imágenes.
R_2_1	Realizar escritura de imágenes

Código	Descripción
R_2_0_0	Se utilizarán funciones provistas por **libpng** para la lectura de imágenes con formato **png**

Código	Descripción
R_2_1_0	Se utilizarán funciones provistas por **libpng** para la escritura de imágenes con formato **png**

Código	Descripción
R_3_0	Se creará un diálogo con el cuál se notificará al usuario que la ejecución del algoritmo ha finalizado.

Código	Descripción
R_4_0	En el diálogo se crearán los correspondientes casilleros para completar con las rutas (absolutas o no) a cada uno de los archivos necesarios y la ruta del directorio de resultados.
R_4_1	En el diálogo se creará un combo para seleccionar la cantidad de horas de simulación.
R_4_2	En el diálogo se creará un combo para seleccionar el método de diferencia finita a utilizar.

3.2.3. Casos de uso

Un *caso de uso* es una técnica para la captura de requisitos potenciales de un nuevo sistema o una actualización de software. Cada caso de uso proporciona uno o más escenarios que indican cómo debería interactuar el sistema con el usuario o con otro sistema para conseguir un objetivo específico. Los casos de uso están representados por elipses y los actores están representados por las figuras humanas [25].

A continuación, describiremos los casos de uso de este sistema:

1. El usuario selecciona la opción *Configurar modelo* para comenzar con el proceso de configuración de la herramienta.

2. Se despliega el diálogo de configuración en el cual está dado un conjunto de condiciones iniciales por defecto sobre el cual, el usuario puede utilizarlas en la simulación o bien puede ingresar los nombres de los archivos correspondiente al conjunto de *condiciones iniciales* que considere necesario.

3. El usuario puede reiniciar a cero la configuración del modelo.

4. El usuario puede aceptar la configuración elegida.

5. El usuario puede cancelar la configuración elegida.

6. El usuario puede seleccionar la cantidad de horas sobre las cuales ejecutar el algoritmo.

7. El usuario puede seleccionar qué tipo de método de aproximación utilizar en la simulación.

8. El usuario puede seleccionar *Ejecutar modelo* para proceder con la ejecución del algoritmo habiendo o no seleccionado un conjunto de condiciones iniciales. Esto se puede hacer dado que el sistema tiene establecido por defecto un conjunto de condiciones iniciales.

9. El usuario puede seleccionar *Graficar* para visualizar el resultado de la simulación

10. El usuario puede seleccionar guardar el resultado de una simulación.

11. El usuario puede guardar una configuración.

12. El usuario puede abrir una configuración dada.

A continuación, presentaremos los diagramas de casos de uso para algunos de los requerimientos descriptos anteriormente. Estos diagramas se realizaron utilizando la herramienta StarUML [4].

Caso 1: El diagrama mostrado a continuación corresponde al item 1 de la sección anterior. Este diagrama muestra los pasos a seguir por el usuario para poder configurar la herramienta.

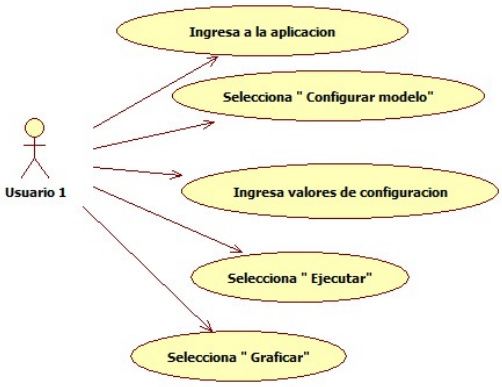

Figura 3.1: Caso 1

Caso 2: El diagrama mostrado a continuación corresponde al item 10 de la sección anterior. En el mismo se muestra la secuencia de pasos a seguir por usuario para poder guardar la imágen de resultado de la simulación.

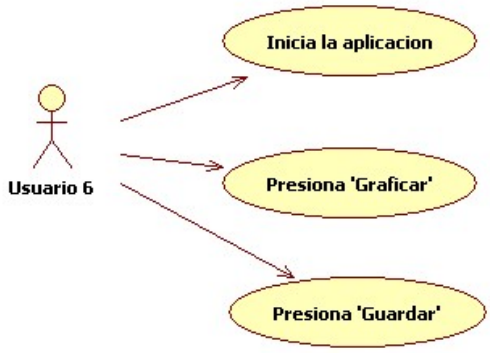

Figura 3.2: Caso 2

Caso 3: El diagrama mostrado a continuación corresponde al item 11 de la sección anterior. En el mismo se muestra la secuencia de pasos a seguir por el usuario para poder guardar la configuración seleccionada.

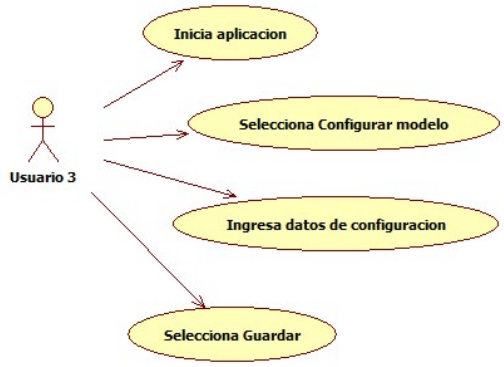

Figura 3.3: Caso 3

Caso 4: El diagrama mostrado a continuación corresponde al item 12 de la sección anterior. En el mismo se describe la secuencia de pasos a seguir por el usuario a fin de comenzar una simulación a partir de una configuración previamente guardada. Dicho archivo de configuración puede haber sido escrito por medio de la opción *Guardar* de la herramienta, o bien manualmente por el usuario siguiendo el formato descripto en el apéndice.

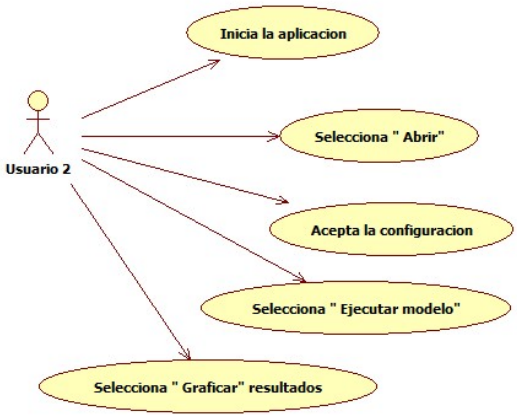

Figura 3.4: Caso 3

Capítulo 4

Modelo de difusión

El Dengue y la fiebre amarilla son problemas importantes en nuestro país y el desarrollo de plataformas que permitan usar la información satelital ayudan la detección temprana de posibles focos de epidemias.
El *aedes aegypti* es un mosquito que pude ser portador del virus del dengue y fiebre amarilla. Puede reconocerse por sus distintivas marcas blancas, aunque sus diferencias en aspecto con respecto a otros mosquitos pueden ser ligeras. Se encuentra más frecuentemente en los trópicos, pero está presente en los estados del sur de los Estados Unidos de América (como por ejemplo Florida)[23].

Figura 4.1: Aedes aegypti
[32]

Dados los requerimientos descriptos en el capítulo anterior, presentaremos el modelo o algoritmo de difusión de insectos [1], en nuestro caso *Aedes aegypti* que se implementará en esta herramienta.

Como lo establecido en [1], dependiendo de la especie, ciertos parámetros determinan el flujo de insectos $\vec{\jmath}$en un punto P y un instante t. Entre estos parámetros, tales como el viento, son considerados generales. Otros, como la temperatura, humedad y proximidad de mamíferos, pueden influir en el flujo de los vectores, dependiendo de la especie.
Existen 3 acciones que influyen sobre el flujo de insectos:

- Fenómeno de transporte

- Fenómeno de fuerzas atractivas y repulsivas
- Fenómeno de fricción ocasionado por la rugosidad del paisaje

4.1. Fenómeno de transporte

Para este modelo, consideraremos al viento un *efecto de transporte*. Si tomamos una superficie v alrededor de un punto P, donde la densidad de insectos está dada por $\rho(P,t)$, el balance del flujo de insectos a través del límite de una superficie σ de v será nulo si la velocidad del viento $\vec{W}(P,t)$ es ortogonal al vector $\nabla\rho(P,t)$ que mide el incremento de ρ. El flujo orientado hacia afuera aumenta σ debido a la velocidad del viento $\vec{W}(P,t)$, es proporcional a $\vec{W}(P,t).\nabla_\rho(P,t)$ y es máxima si el vector $\vec{W}(P,t)$ y $\nabla\rho(P,t)$ tienen la misma dirección. Por ejemplo, si los valores de $\vec{W}$ provienen de una estación meteorológica, el vector local $\vec{W}(P,t)$ en el punto P está sujeto a deformaciones debido a la rugosidad del paisaje (bosque, carreteras, viviendas). De esta forma, el incremento del flujo de insectos $\jmath$ a través de σ está denotado por el término:

$$-div[D_W(P,t).\vec{W}(P,t).\nabla\rho(P,t)] \tag{4.1}$$

En este término, $D_W(P,t)$ representa un tensor debido a la influencia de la rugosidad del paisaje sobre el flujo del viento. En adelante lo llamaremos **tensor de viento**.

4.2. Fenómeno de fuerzas atractivas y repulsivas

Las fuerzas atractivas pueden ser producidas por la presencia de mamíferos, distintas especies de vegetales, temperatura o humedad. A dicha fuerza la representaremos como $H(P,t)$.
Las fuerzas repulsivas, actúan en dirección opuesta sobre el flujo de insectos. Estas fuerzas pueden ser causadas por insecticidas o rangos inapropiados de temperatura y humedad. Estas fuerzas modifican el balance de insectos para el punto p sobre una superficie σ. Para este modelo, supondremos que la distribución de humanos crea una fuerza atractiva sobre los insectos, la cual orientará el flujo siguiendo la siguiente término: $-\nabla_H(P,t)$. Luego, el flujo de insectos debido a la distribución de humanos será proporcional a $\rho(P,t)$ y $-\nabla H(P,t)$. Finalmente, el papel del campo de fuerzas atractivas creado por la distribución H, contribuirá al balance del flujo a través de σ siguiendo el siguiente término:

$$-div[K_H.\rho(P,t).\nabla H(P,t)] \tag{4.2}$$

donde K_H es una constante positiva.

4.3. Rugosidad del paisaje

La rugosidad de la superficie es otra clase de parámetro que influye en el flujo de los insectos. De hecho, la diferencia de rugosidad entre un bosque y una carretera o dentro de una ciudad, ralentiza y orienta el flujo de insectos. Este efecto es independiente del efecto del viento. Por ejemplo, este efecto de rozamiento en los bosques es más alto que en las carreteras. Dicho fenómeno, opera sobre el vector de insectos $\vec{j}$ como un tensor de difusión. Luego la contribución sobre el balance del flujo sobre σ está dado por el siguiente término:

$$div[D_R(P,t).\nabla\rho(P,t)] \tag{4.3}$$

donde $D_R(P,t)$ es un tensor definido por el efecto directo de la superficie en la dinámica de los insectos.

4.4. Nacimientos y muertes

La taza de nacimientos y muertes es otro factor que influye en el flujo de los insectos. Están representados por:

$$\alpha(P,t)] \tag{4.4}$$

$$\beta(P,t)] \tag{4.5}$$

4.5. El modelo final

En las secciones anteriores calculamos por separado las ecuaciones correspondientes para el *fenómeno de transporte*, *fenómeno de fuerzas atractivas y repulsivas*, *fenómeno de rozamiento* y mostramos como modifican el flujo de insectos. Luego, la variación en el tiempo de ρ sobre un punto P representa flujo a través de una pequeña superficie alrededor del punto P sumado a la producción de insectos, con lo cual queda la siguiente ecuación:

$$density(P,t) = \alpha(P,t) - \beta(P,t) \tag{4.6}$$

donde **density(P,t)** está dado por:

$$\frac{\partial\rho}{\partial t}(P,t) - div[D_R(P,t).\nabla\rho(P,t)] + div[D_W(P,t).\vec{W}(P,t).\nabla\rho(P,t)] + div[K_H.\rho(P,t).\nabla H(P,t)] \tag{4.7}$$

para cada $P \in A$ y $t > 0$, donde A representa la superficie.

4.6. Implementación

Con el fin de poder representar las derivadas parciales presentes en la ecuación, utilizaremos el método de *Diferencias Finitas*.
Una diferencia finita es una expresión matemática de la forma $f(x+b) - f(x+a)$. Sólo se consideran normalmente tres formas: la anterior, la posterior y la central.

- **Diferencia finita posterior**: Una diferencia posterior es una expresión de la forma:

$$\Delta[f](x) = f(x) - f(x-h) \tag{4.8}$$

A continuación, veremos como queda el término correspondiente al fenómeno de fuerzas atractivas y repulsivas

$$-div[K_H.\rho(P,t).\nabla H(P,t)]$$

Sabemos que:

$$\nabla H(P,t) = (\tfrac{\partial H}{\partial x}(P,t), \tfrac{\partial H}{\partial y}(P,t), \tfrac{\partial H}{\partial z}(P,t))$$

Luego,

$$\rho(P,t).\nabla H(P,t) = (\rho(P,t).\tfrac{\partial H}{\partial x}(P,t), \rho(P,t).\tfrac{\partial H}{\partial y}(P,t), \rho(P,t).\tfrac{\partial H}{\partial z}(P,t))\ ^{1}.$$

Dado que $\frac{\partial f}{\partial x} = \frac{f(x+1)-f(x)}{\Delta x}$, tenemos:

$$\rho(P,t).\nabla H(P,t) = (\rho(P,t).\tfrac{H(t,x+1,y)-H(t,x,y)}{\Delta x}, \rho(P,t).\tfrac{H(t,x,y+1)-H(t,x,y)}{\Delta y})$$

Luego

$$K_H.\rho(P,t).\nabla H(P,t) = (K_H.\rho(P,t).\tfrac{H(t,x+1,y)-H(t,x,y)}{\Delta x}, K_H.\rho(P,t).\tfrac{H(t,x,y+1)-H(t,x,y)}{\Delta y}).$$

Por definición, $divF = \frac{\partial F}{\partial x} + \frac{\partial F}{\partial y} + \frac{\partial F}{\partial z}$.
Tomemos $A = K_H.\rho(P,t).\frac{H(t,x+1,y)-H(t,x,y)}{\Delta x}$ y sea $B = K_H.\rho(P,t).\frac{H(t,x,y+1)-H(t,x,y)}{\Delta y}$, luego:

[1] Para nuestro caso, tomaremos solo un espacio bidimensional.

$$div[K_H.\rho(P,t).\nabla H(P,t)] = \frac{\partial A}{\partial x} + \frac{\partial B}{\partial y}$$

Luego, aplicamos nuevamente diferencia finita posterior a los términos que contienen ∂, y queda:

$$div[K_H.\rho(P,t).\nabla H(P,t)] = K_H.\rho(P,t).\frac{H(t,x+2,y) - H(t,x+1,y)}{\Delta x} + K_H.\rho(P,t).\frac{H(t,x,y+2) - H(t,x,y+1)}{\Delta y} \tag{4.9}$$

- **Diferencia finita anterior**: Una diferencia anterior se obtiene de la posterior reemplazando $-h$ por h de la siguiente manera:

$$\Delta[f](x) = f(x+h) - f(x) \tag{4.10}$$

Dependiendo de la aplicación, el espaciado h se mantiene constante o se toma el limite $h \to 0$ A continuación, veremos como queda el término correspondiente al fenómeno de fuerzas atractivas y repulsivas como en la sección anterior:

$$div[K_H.\rho(P,t).\nabla H(P,t)] = K_H.\rho(P,t).\frac{H(t,x-2,y) - H(t,x-1,y)}{\Delta x} + K_H.\rho(P,t).\frac{H(t,x,y-2) - H(t,x,y-1)}{\Delta y} \tag{4.11}$$

- **Diferencia finita central**: es la media de las diferencias anteriores y posteriores. Viene dada por:

$$\Delta[f](x) = \frac{f(x+h) - f(x-h)}{2} \tag{4.12}$$

A continuación, mostraremos como queda el término correspondiente al fenómenos de fuerzas atractivas y repulsivas como en la sección anterior:

$$div[K_H.\rho(P,t).\nabla H(P,t)] = K_H.\rho(P,t).\frac{H(t,x+2,y) - H(t,x-2,y)}{2} + K_H.\rho(P,t).\frac{H(t,x,y+2) - H(t,x,y-2)}{2} \tag{4.13}$$

Para la implementación de este software el usuario podrá seleccionar que método de diferencia finita desea utilizar para la simulación.

Capítulo 5

Diseño del software

En este capítulo presentaremos la base teórica usada en el diseño del software y la descripción del mismo.

5.1. Fundamentos del diseño

El diseño del software de este trabajo de grado está basado en los patrones de diseño[1] **Strategy** (Estrategia) y **Template**. El patrón **Strategy** permite mantener un conjunto de algoritmos de los que el objeto cliente puede elegir aquél que le conviene e intercambiarlo según sus necesidades. Los distintos algoritmos se encapsulan y el cliente trabaja contra un objeto *Context* (Contexto). El cliente puede elegir el algoritmo que prefiera de entre los disponibles o puede ser el mismo objeto Context el que elija el más apropiado para cada situación. Cualquier programa que ofrezca un servicio o función determinada, que pueda ser realizada de varias maneras, es candidato a utilizar el patrón Strategy. Puede haber cualquier número de estrategias y cualquiera de ellas podrá ser intercambiada por otra en cualquier momento, incluso en tiempo de ejecución.

[1] Un patrón de diseño es una solución a un problema de diseño. Para que una solución sea considerada un patrón debe poseer ciertas características. Una de ellas es que debe haber comprobado su efectividad resolviendo problemas similares en ocasiones anteriores. Otra es que debe ser reusable, lo que significa que es aplicable a diferentes problemas de diseño en distintas circunstancias.

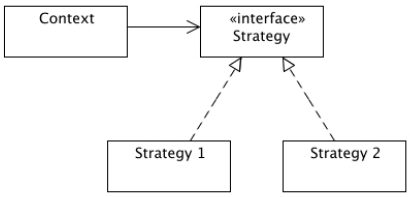

Figura 5.1: Strategy Pattern

El patrón **Template** define el esqueleto de un algoritmo dejando los detalles a las subclases. Este patrón permite que las subclases redefinan ciertos pasos de un algoritmo dado sin cambiar la estructura inicial del algoritmo. De esta manera, dicho algoritmo se comportará de acuerdo a la clase que lo reimplemente. Una forma de implementar este tipo de patrón es por medio de sobreescritura de métodos.

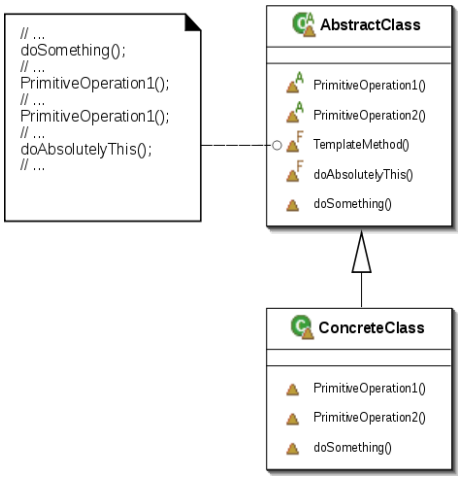

Figura 5.2: Template Pattern

5.2. Diagrama de componentes

En esta sección describiremos cada uno de los componentes que forman parte de la herramienta. La misma está formada por dos módulos principales:

1. Interfaz de usuario
2. Modelo

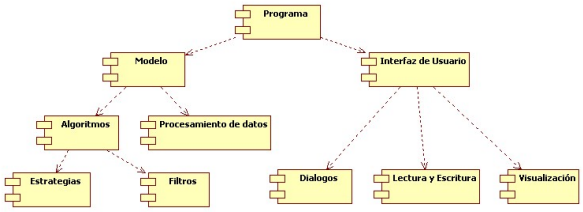

Figura 5.3: Diagrama de componentes

A continuación, describiremos cada uno de los submódulos presentes en el diagrama.

5.2.1. Modelo

Este módulo consta de dos submódulos:

1. **Algoritmos**: este submódulo consta de dos partes, **Estrategias** y **Filtros**. En Estrategias se implementan los algoritmos correspondientes al método de simulación que se va a utilizar. En nuestro caso, en este módulo encontraremos las implementaciones de la **diferencia finita anterior**, **posterior** y **central**. En **Filtros** se implementarán cada uno de los tipos de análisis que se le realizará a cada píxel. En nuestro caso, en este módulo encontraremos las implementaciones de cada uno de los términos de la ecuación 6.1 y también los métodos de normalización de los datos.

2. **Procesamiento de datos de entrada**: en este submódulo se encuentran las implementaciones de las clases que se encargaran de la carga de la información proveniente de los archivos de condiciones iniciales. Para ello, se implementaron las clases **ImageData**, **EnviDatFile**, **Point**, **Pair** y **OMatrix**.

5.2.2. Interfaz de usuario

Este módulo consta de 3 submódulos:

1. **Diálogos**: en este módulo se encuentran las implementaciones de todos los diálogos de esta herramienta. Aquí podemos encontrar **MainWindow**, que es el diálogo principal; **DialogModelConfigTool**, que es el diálogo de configuración del modelo; **Ayuda**, que es un diálogo con información sobre la herramienta.

2. **Lectura y escritura**: en este módulo se implementan los mecanismos de lectura y escritura de imágenes. En nuestro caso, se realizará la lectura y escritura de archivos PNG usando las funciones provistas por libpng[6].

3. **Visualización**: en este módulo se implementa la ventana de visualización de los resultados.

5.3. Diagrama de clases

En la sección 3.1.1 presentamos las características generales de este tipo de diagrama. En esta sección se presentará la definición de cada una de las partes que forman un diagrama de clases [34].

- **Propiedades**: también llamados atributos o características, son valores que corresponden a un objeto, como color, material, cantidad, ubicación. Generalmente se conoce como la información detallada del objeto. En la siguiente figura mostraremos como se representan en StarUML las propiedades de una clase.

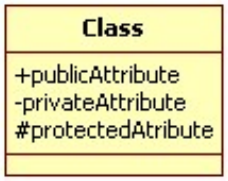

Figura 5.4: Ejemplo de propiedades de una clase

- **Operaciones**: son aquellas actividades o verbos que se pueden realizar con o para este objeto, como por ejemplo abrir, cerrar, buscar, cancelar, acreditar, cargar. De la misma manera que el nombre de un atributo, el nombre de una operación se escribe con minúsculas si consta de una sola palabra. Si el nombre contiene más de una palabra, cada palabra será unida a la anterior y comenzará con una letra mayúscula, a excepción de la primera palabra que comenzará en minúscula.

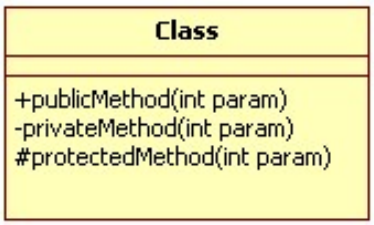

Figura 5.5: Ejemplo de operaciones de una clase

- **Herencia**: se define como la reutilización de un objeto padre ya definido para poder extender la funcionalidad en un objeto hijo. Los objetos hijos heredan todas las operaciones y/o propiedades de un objeto padre.

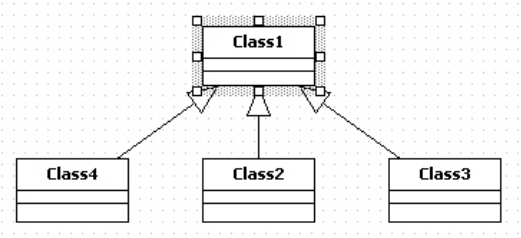

Figura 5.6: Ejemplo de herencia

5.3.1. Clase Model

En esta clase se implementará el algoritmo completo que se utilizará para la simulación. Cada uno de los términos de la ecuación descripta en 6.1 será implementado como un tipo de *Filtro*.

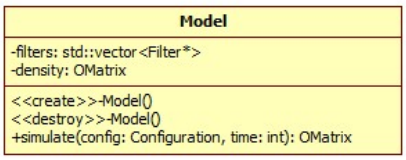

Figura 5.7: Clase Model

5.3.2. Clase Filter

Esta clase es la clase base para la implementación de cada uno de los términos de la ecuación 6.1. Uno de sus métodos *apply* es virtual debido al patrón **Template** definido anteriormente, con lo cual, cada una de las clases que herede de Filter puede sobreescribir este método para que se comporte de acuerdo al filtro que se desea aplicar. Para este trabajo final, se decidió aplicar este enfoque para permitir mayor flexibilidad en la modularización, así si en un futuro se decide agregar más términos, se puede crear un nuevo filtro sin realizar mayores modificaciones en el resto del programa.

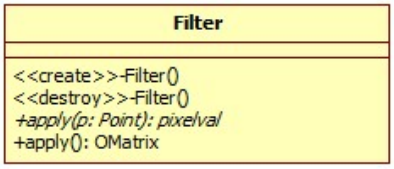

Figura 5.8: Clase Filter. NOTA: con cursiva se denotan en StarUML los métodos virtuales de una clase.

Las clases que heredan de Filter son:

- **Clase FilterTransport:** esta clase se utiliza para representar el término correspondiente al **Fenómeno de transporte** presente en la ecuación. Los datos a partir de los cuales se creará un objeto de esta clase son los nombres de los archivos de condiciones iniciales provistos por el usuario. Dichos archivos son: tensor de viento, viento y densidad inicial de mosquitos.

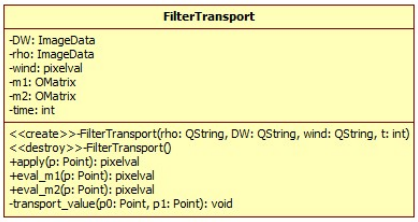

Figura 5.9: Clase FilterTransport

- **Clase FilterRoughness:** esta clase se utiliza para representar el término correspondiente al **Fenómeno de rozamiento** presente en la ecuación. Los datos a partir de los cuales se creará un objeto de esta clase son los nombres de los archivos de condiciones iniciales provistos por el usuario. Dichos archivos son: tensor de rozamiento y densidad inicial de mosquitos.

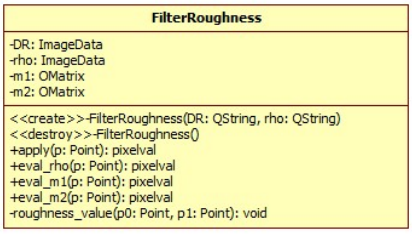

Figura 5.10: Clase FilterRoughness

- **Clase FilterAttraction:** esta clase se utilizará para representar el término correspondiente al **Fenómeno de atracción** presente en la ecuación. Los datos a partir de los cuales se creará un objeto de esta clase son los nombres de los archivos de condiciones iniciales provistos por el usuario. Dichos archivos son: atractores y densidad inicial de mosquitos.

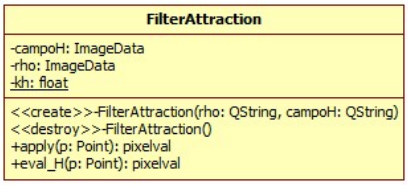

Figura 5.11: Clase FilterAttraction

- **Clase FilterDeaths y FilterBirths:** estas dos clases se utilizaran para representar el factor de nacimientos y muertes presentes en la ecuación. Los datos a partir de los cuales se creará un objeto de esta clase son los nombres de los archivos de condiciones iniciales provistos por el usuario. Dichos archivos son: nacimientos y muertes respectivamente.

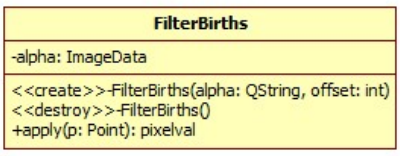

Figura 5.12: Clase FilterBirths

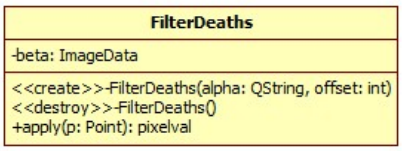

Figura 5.13: Clase FilterDeaths

- **Clase HighPassFilter y LowPassFilter:** estas dos clases se utilizaran normalizar los resultados debido a la discretización de la ecuación. Para ello se utiliza el concepto de *Filtro de pasa bajo* y *Filtro de pasa algo* En estos dos filtros, la función *apply* se implementó de manera tal que restablezca los valores que están por debajo de un *umbral*, en el caso de HighPassFilter. Para el caso de LowPassFilter, dado un punto p, el valor de ese punto se normalizará tomando el promedio de los 4 vecinos, es decir, si p es el punto cuyas coordenadas son (x, y), entonces sus vecinos serán los puntos dados por las siguientes coordenadas $(x+1, y)$, $(x, y+1)$, $(x-1, y)$, $(x, y-1)$.

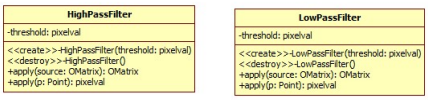

Figura 5.14: Clase HighPassFilter y LowPassFilter

5.3.3. Clase StrategyFiniteDifference

La clase StrategyFiniteDifference, es la clase base de cada uno de los métodos de diferencia finita que se pueden seleccionar en la herramienta. Todos los métodos de esta clase son virtuales, dado que cada una de las clases que herede de StrategyFiniteDifference reimplementará los que se consideren necesarios.

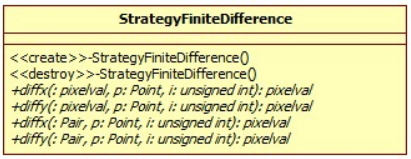

Figura 5.15: Clase StrategyFiniteDifference NOTA: con cursiva se denotan en StarUML los métodos virtuales de una clase.

Heredan de esta clase:

- **StrategyForwardDifference**: en esta clase se implementa la diferencia finita posterior.

- **StrategyBackwardDifference**: en esta clase se implementa la diferencia finita anterior.
- **StrategyCentralDifference**: en esta clase se implementa la diferencia finita central.

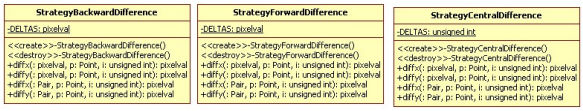

Figura 5.16: Clase StrategyForwardDifference, StrategyBackwardDifference y StrategyCentralDifference

5.3.4. Clase EnviFile

En la clase **EnviFile** se implementarán las rutinas encargadas de la carga de datos. Se define esta clase base para considerar los distintos tipos de archivos generados por Envi, como los archivos de clasificación, sean procesados como sea necesario. En este caso, se declaró una subclase, EnviDatFile, la cual se encargará de la carga de datos correspondiente al archivo de condiciones iniciales provistos por el usuario.

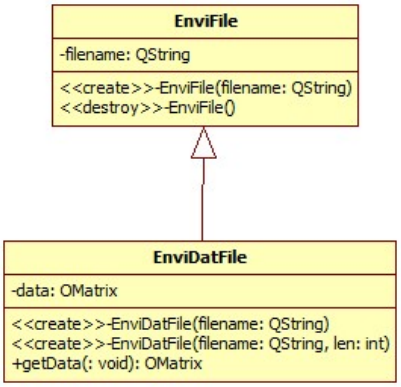

Figura 5.17: Clase EnviFile

5.3.5. Clase ImageData

En la clase **ImageData** se implementa la carga de la información contenida en un *EnviFile* para su posterior procesamiento. El metodo principal de esta clase es **evaluación** indexada por puntos.

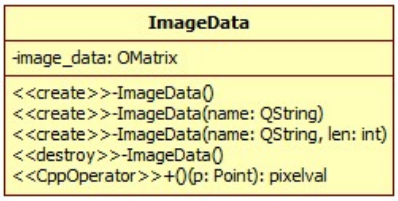

Figura 5.18: Clase ImageData

5.3.6. Clase ImageViewer e ImageWriter

Estas dos clases se crearon con la finalidad de que *ImageViewer* se encargue de la visualización de los resultados e *ImageWriter* se encargue de la escritura de los archivos con formato **png**, tarea que se llevó a cabo con llamadas a funciones de la librería *libpng*.

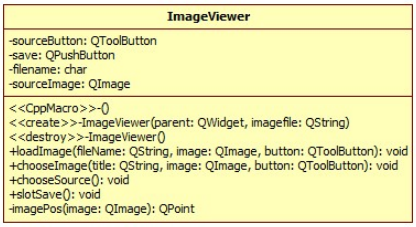

Figura 5.19: Clase ImageViewer

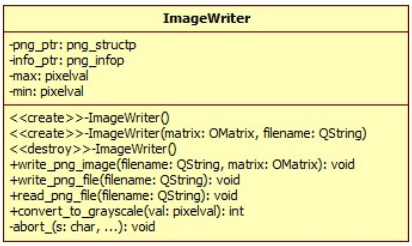

Figura 5.20: Clase ImageWriter

5.3.7. Clase Configuration

Dado que la herramienta tiene dos formas de configuración: por medio de un archivo de configuración o por medio de la interfaz gráfica, se creó esta clase la cual se encarga de cargar la información requerida en cada uno de sus atributos por medio de la función *set_member*.

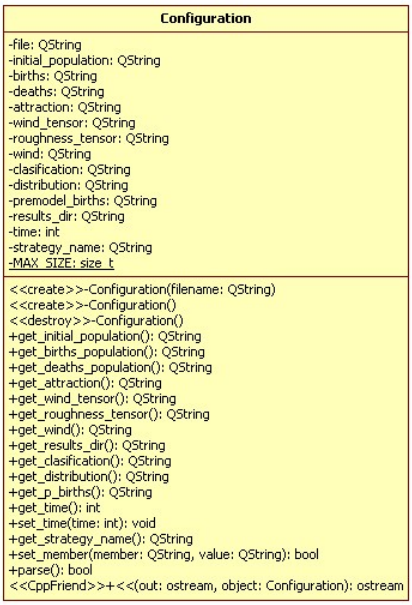

Figura 5.21: Clase Configuration

5.3.8. Clase MainWindow, DialogModelConfigTool y Ayuda

Estas tres clases son las que implementan toda la estructura de la interfaz de usuario.

- **MainWindow:** en esta clase se implementa toda la estructura, conectores y funciones del diálogo principal, es decir el primer diálogo que se despliega cuando se inicia la ejecución de la herramienta.

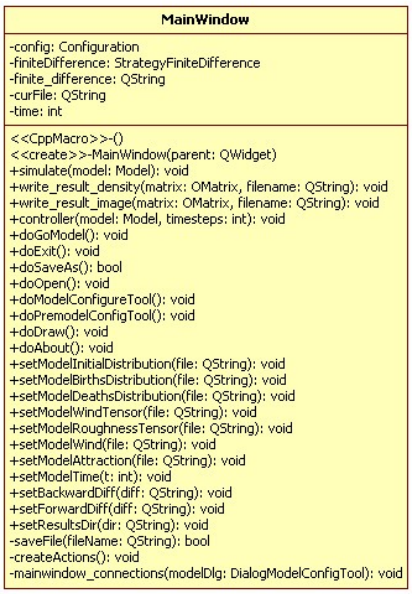

Figura 5.22: Clase MainWindow

- **DialogModelConfigTool:** en esta clase se implementa la estructura, conectores y funciones del diálogo de configuración del modelo. A este diálogo se puede acceder seleccionando del menú principal la opción *Configurar Modelo*

Figura 5.23: Clase DialogModelConfigTool

- **Ayuda:** en esta clase se implementa un diálogo con información al usuario sobre versión de la herramienta, autor, etc.

5.3.9. Clase OMatrix, Point y Pair

Estas tres clases se definieron con la finalidad de independizar el desarrollo de la herramienta de los tipos de datos usados. La clase **OMatrix** se definió para el almacenamiento de datos sobre la cual luego se agregaron métodos como el de evaluación de una omatrix por medio de un punto. La clase **Point** se definió con la finalidad de *indexar* la información proveniente de una imágen satelital.

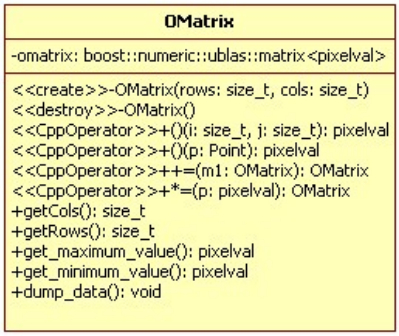

Figura 5.24: Clase OMatrix

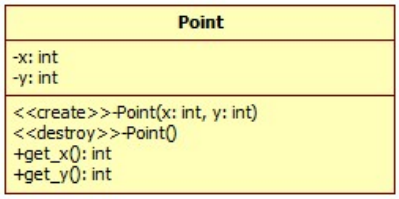

Figura 5.25: Clase Point

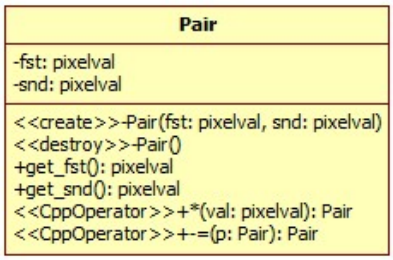

Figura 5.26: Clase Pair

5.4. Diagrama de secuencia

Un diagrama de secuencia muestra los objetos que intervienen en el escenario con líneas discontinuas verticales, y los mensajes pasados entre los objetos como flechas horizontales.
Existen dos tipos de mensajes: síncronos y asíncronos. Los mensajes síncronos se corresponden con llamadas a métodos del objeto que recibe el mensaje. El objeto que envía el mensaje queda bloqueado hasta que termina la llamada. Este tipo de mensajes se representan con flechas con la cabeza llena. Los mensajes asíncronos terminan inmediatamente, y crean un nuevo hilo de ejecución dentro de la secuencia. Se representan con flechas con la cabeza abierta. También se representa la respuesta a un mensaje con una flecha discontinua.
Los mensajes se dibujan cronológicamente desde la parte superior del diagrama a la parte inferior; la distribución horizontal de los objetos es arbitraria.
El método llamado, o invocado, pertenece a la definición de la clase instanciada por el objeto en la recepción final del mensaje [35].

En esta sección, presentaremos el diagrama de secuencia de mensajes correspondiente al algoritmo utilizado. En primer lugar, se describen las acciones a tomar para la inicialización de los datos correspondientes a la ventana de configuración de la simulación.

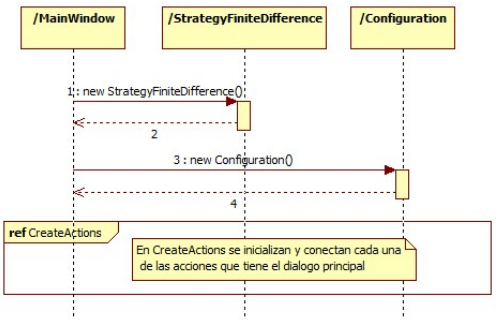

Figura 5.27: Inicialización de la ventana principal

1. Se crea un objeto del tipo StrategyFiniteDifference
2. Este mensaje indica el retorno a la llamada anterior.
3. Se crea un objeto del tipo Configuration, el cual se crea a partir de un archivo de configuración por defecto.
4. Este mensaje indica el retorno a la llamada anterior.
5. A continuación se presenta una referencia a un diagrama de estado en el cual se crean y conectan las acciones del menú y los botones utilizando el concepto de *Slots and Signals* de Qt [5].

Luego, una vez que los datos fueron cargados por el usuario en la interfaz, se procede con la ejecución del algoritmo de simualción.
Esto se puede ver accediendo a la opción *Ejecutar modelo* de la barra de herramientas del programa.
A continuación, describiremos el diagrama de secuencia para dicho algoritmo. Se crearon referencias, representadas con **ref** en la esquina superior izquierda de cada diagrama para facilitar la legibilidad.

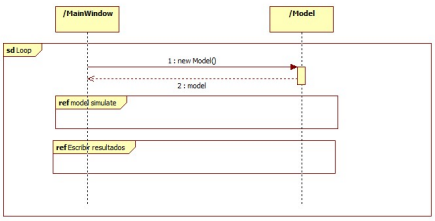

Figura 5.28: Algoritmo de simulación

1. Se crea un objeto del tipo Model
2. Este mensaje indica el retorno a la llamada anterior.

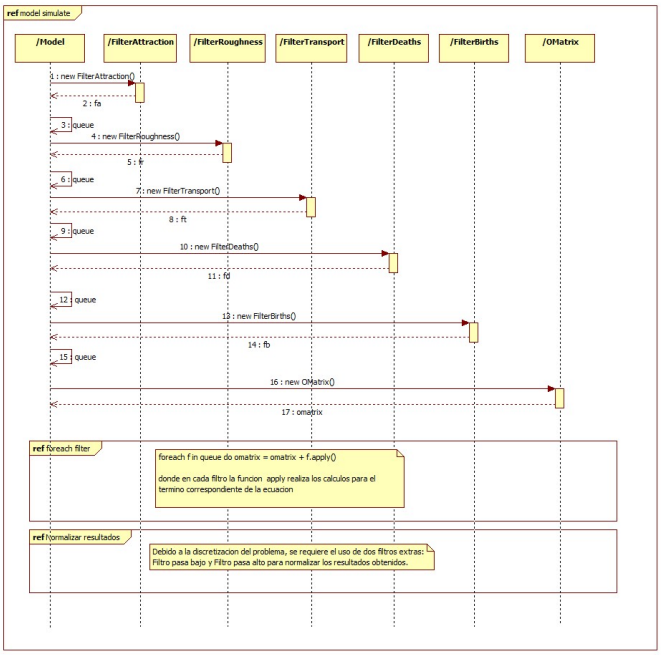

Figura 5.29: Método simulate

1. Se crea un objeto del tipo FilterAttraction. Este objeto se utilizará para invocar a la función que calcula el valor de las fuerzas atractivas y repulsivas para un punto dado.

2. Este mensaje indica el retorno a la llamada anterior.

3. Se encola este filtro.

4. Se crea un objeto del tipo FilterRoughness. Este objeto se utilizará para invocar a la función que calcula el valor del rozamiento del paisaje para un punto dado.

5. Este mensaje indica el retorno a la llamada anterior.

6. Se encola este filtro.

7. Se crea un objeto del tipo FilterTransport. Este objeto se utilizará para invocar a la función que calcula el valor del fenómeno de transporte para un punto dado.

8. Este mensaje indica el retorno a la llamada anterior.

9. Se encola este filtro.

10. Se crea un objeto del tipo FilterDeaths. Este objeto se utilizará para invocar a la función que calcula el valor de muertes para un punto dado.

11. Este mensaje indica el retorno a la llamada anterior.

12. Se encola este filtro.

13. Se crea un objeto del tipo FilterBirths. Este objeto se utilizará para invocar a la función que calcula el valor de nacimientos para un punto dado.

14. Este mensaje indica el retorno a la llamada anterior.

15. Se encola este filtro.

16. Se crea un objeto del tipo OMatrix, el cual se utilizará para almacenar el valor total de la ecuación para un punto dado.

17. Este mensaje indica el retorno a la llamada anterior.

18. El siguiente paso es ejecutar la funcion *apply()* para cada punto y cada uno de los filtros encolados.

19. A continuación, una vez obtenido el resultado, se invoca al filtro de Pasa alto que se encargará de eliminar aquellos valores inválidos.

A continuación, cada vez que se ejecuta el método *apply()* de cada filtro, se invoca el método *write_image* de la clase ImageWriter para la escritura de los resultados. Una vez finalizada la simulación para una cantidad dada de horas, se invoca al método *load_image* de la clase ImageViewer para la visualización de los resultados. Esto se puede ver accediendo a la opción *Graficar* presente en la barra de herramientas del programa.

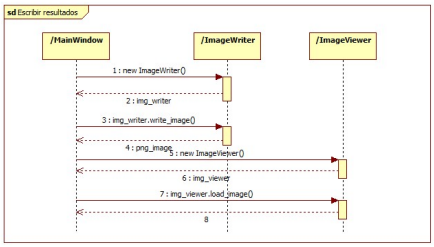

Figura 5.30: Escritura y visualización de resultados

1. Se crea un objeto ImageWriter.
2. Este mensaje indica el retorno a la llamada anterior.
3. Se invoca al método **write_image** en el cual se implementó el algoritmo de escritura de archivos png
4. Este mensaje indica el retorno a la llamada anterior.
5. Se crea un objeto ImageViewer
6. Este mensaje indica el retorno a la llamada anterior.
7. Se invoca al método **load_image** que se encargará de desplegar el diálogo mostrando la imágen resultado.

Capítulo 6

Caso de estudio

6.1. Generación de archivos de condiciones iniciales

Para probar el funcionamiento del software desarrollado en forma completa, se utilizará para la simulación los datos descriptos en [3], los cuales provienen del análisis de la ciudad **Hipólito Yrigoyen** y alrededores, Salta. Se consideró un solo paisajes de 5.7 x 5.7 km para ejecutar el modelo definido en 4.6.

Figura 6.1: Ciudad de Hipólito Yrigoyen, Salta. Imagen LandSat TM5

El area de estudio que se utilizará es un área rural, con casas aisladas rodeadas de ríos y bosques ($23\,°9'31{,}08''S, 64\,°14'52{,}58''E$)

Despejando de 4.6 el término correspondiente a $\frac{\partial \rho}{\partial t}(P,t)$, tenemos:

$$\frac{\partial \rho}{\partial t}(P,t) = Rozamiento(P,t) - Transporte(P,t) - Atraccion(P,t) + \alpha(P,t) - \beta(P,t) \tag{6.1}$$

en la cual:

$$Rozamiento(P,t) = div[D_R(P,t).\nabla\rho(P,t)] \tag{6.2}$$

$$Transporte(P,t) = [D_W(P,t).\vec{W}(P,t).\nabla\rho(P,t)] \tag{6.3}$$

$$Atraccion(P,t) = div[K_H.\rho(P,t).\nabla H(P,t)] \tag{6.4}$$

donde: ρ = densidad de mosquitos
D_R = tensor de rozamiento
D_W = tensor de viento
W = velocidad del viento
K_H = constante de atracción
H = campo de atracción
α =razón de nacimientos
β =razón de muertes

Para la generación de estas valores, utilizaremos una imágen LandSat 5TM tomada el 25 de Enero del 2004 y tomaremos 5 clases de rugosidad en el paisaje, el índice de vegetación de diferencia normalizada (NDVI) y factores de vegetación dados por la transformación *Tasseled Cap* El NDVI transforma datos multiespectrales (Rojo Cercano e infrarrojo Cercano) en una sola banda de la imágen que representa la distribución de la vegetación, donde en cada píxel se indicará la cantidad de vegetación verde presente en el mismo. En la transformación *Tasseled Cap* utilizaremos las capas: **Brillo**, **Verdor** y un **Tercero** que serán obtenidos por una combinación lineal de las bandas originales tales que:

- **Brillo**: será el índice de brillo del suelo
- **Verdor**: será el índice de vegetación verde
- **Tercer componente**: estará relacionado con ciertas propiedades del suelo, como por ejemplo la humedad.

También se aplicará un Filtro de textura a la banda del NDVI. Dentro de las 5 clases de rugosidad del paisaje tenemos:

1. **Agua**: que obtendremos por medio del método de clasificación supervisada usando la *máxima probabilidad* de 3 bandas Tasseled Cap y las bandas 1-2-3-4-5-7 del LandSat 5TM
2. **Suelo descubierto**: que obtendremos por la banda de NDVI cuyos valores sean menores que 0.25

3. **Vegetación baja**, **Bosque abierto** y **Selva**: que obtendremos a partir de Regiones de interés derivadas de 3 diferentes umbrales del NDVI.

Figura 6.2: Clasificaciones

6.2. Análisis de los resultados

Esta herramienta genera dos tipos de salidas:

- Imágenes formato PNG
- Archivos de dato sin formato

Los resultados se pueden visualizar desde la herramienta, seleccionando la opcion *Graficar* desde la barra de herramientas. Los resultados de los archivos sin formato se pueden utilizar para ser procesados por herramientas más poderosas como Envi [7]. A continuación, algunas imágenes de los resultados obtenidos usando los datos descriptos en la sección 6.1 y el método de diferencia finita posterior.

Figura 6.3: Pantalla principal

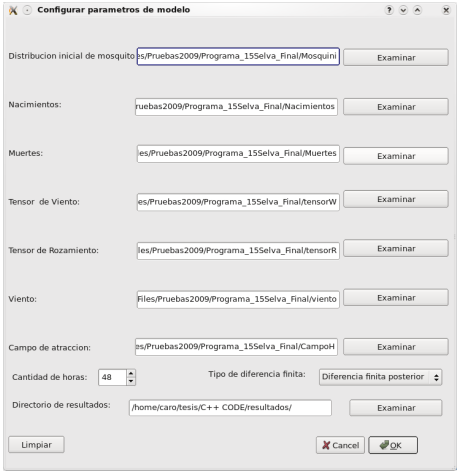

Figura 6.4: Pantalla de configuración del modelo

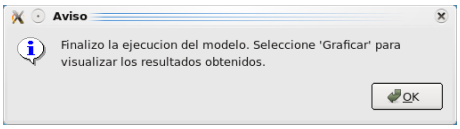

Figura 6.5: Finalizó ejecución

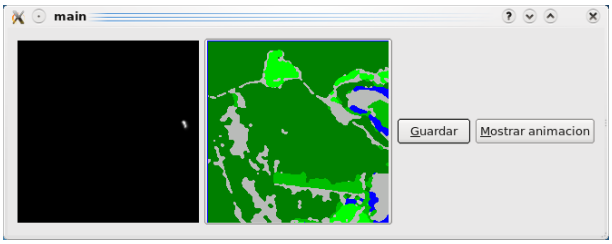

Figura 6.6: Visor de resultados

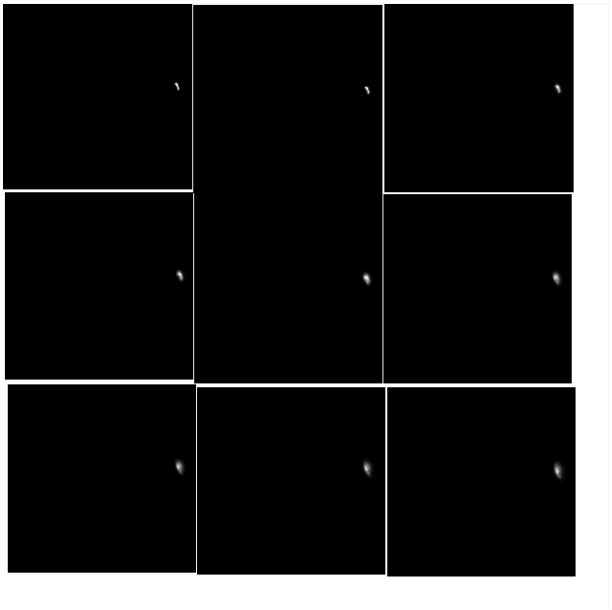

Figura 6.7: Hora 1 a hora 9

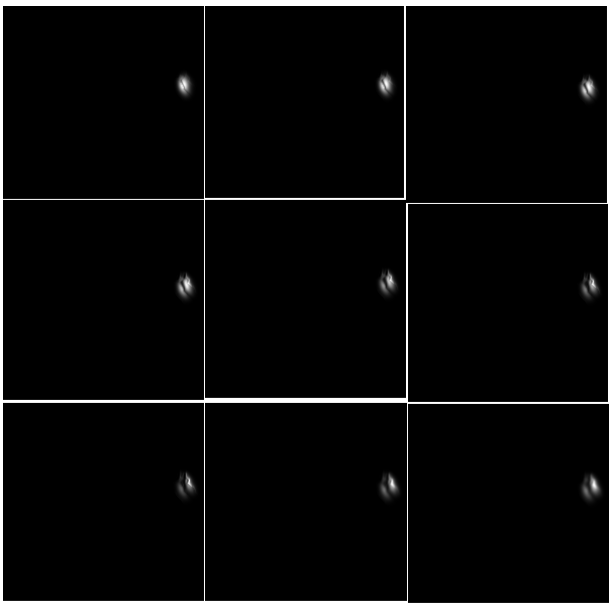

Figura 6.8: Hora 20 a hora 27

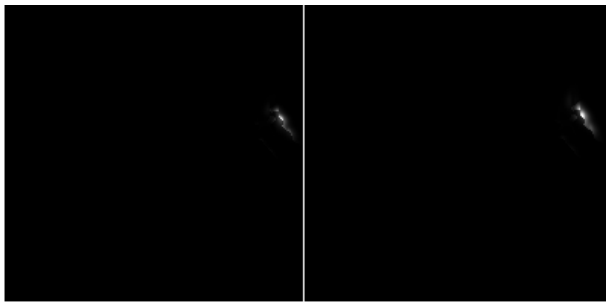

Figura 6.9: Hora 47 a hora 48

Gracias a la *teledetección ambiental* podemos recrear diferentes escenarios para simular la dinámica de algunos insectos. En esta herramienta implementamos el modelo de difusión propuesto por Annelise Tran en [1] en el cual influyen parámetros como la población inicial de mosquitos, el clima y la rugosidad del paisaje.

En la imágen 6.7, podemos apreciar como se incrementa la población de mosquitos cuyos nacimientos fueron especificados en esos puntos. En ese instante de tiempo, no hay influencia del viento, solo influyen la rugosidad del paisaje y los atractores (por ejemplo, la presencia de mamíferos).
En la imágen 6.9 podemos apreciar la densidad final de mosquitos.

Para poder analizar con más detalle estos resultados, utilizaremos ENVI cuyo procesamiento es más especializado.

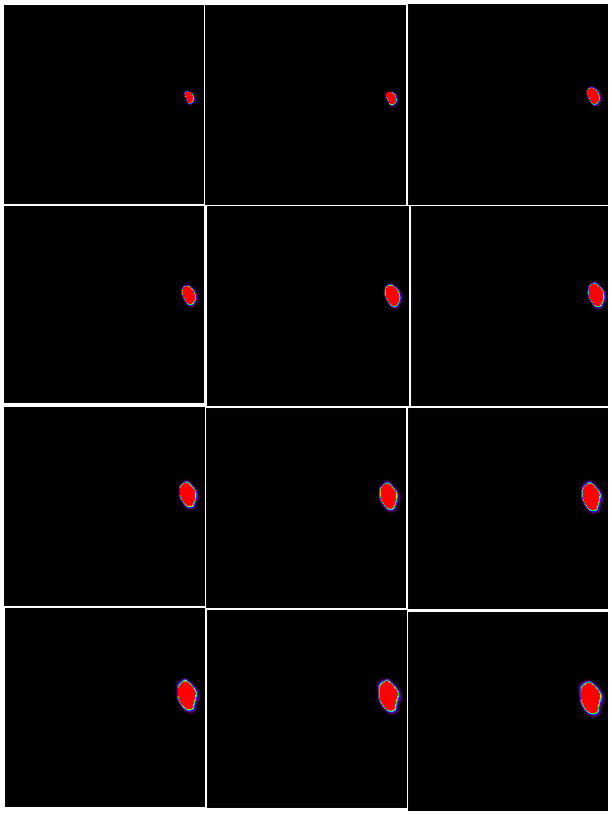

Figura 6.10: Hora 1 a hora 12

Se puede apreciar que en las primeras horas se observa una dispersión homogénea de la distribución de población de insectos. No se observan problemas con las condiciones de contorno.

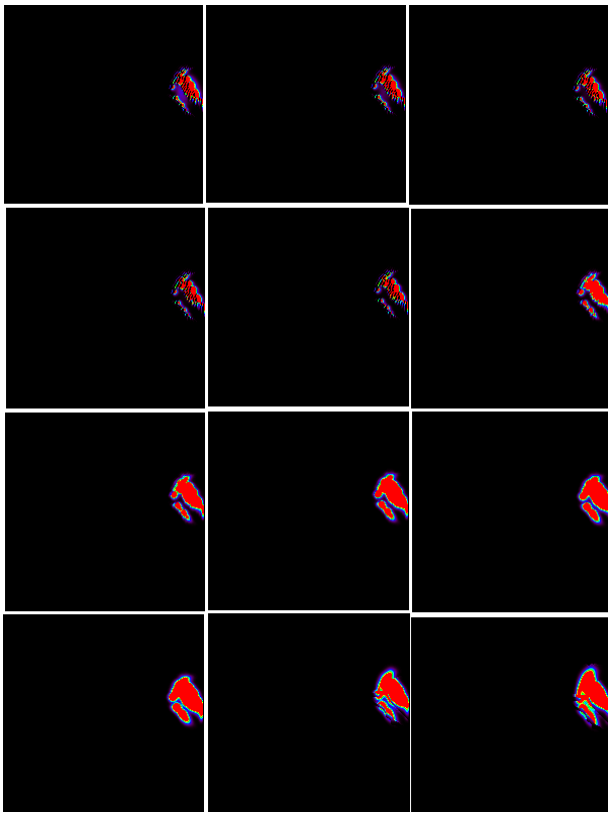

Figura 6.11: Hora 37 a hora 48

A medida que pasan las horas se observa una heterogeneidad espacial en la distribución, lo cual es producto de las irregularidades del paisaje y la velocidad del viento.

Además de estos resultados se realizaron diferentes ejecuciones del modelo modificando los parámetros de entrada, como el valor del viento, muertes, densidad inicial. Se realizaron pruebas anulando la influencia del vector del viento, esto es tomando el viento como cero para todas las horas; también se modificó el archivo de muertes suponiendo que no se produjeran muertes durante el periodo de la simulación. De esta manera pudimos observar como influye cada uno de los términos de la ecuación en la dispersión de los mosquitos. También, se varió el tipo de diferencia finita utilizado.
A modo de ejemplo, en esta sección solo se exponen las imágenes correspondientes a una ejecución cuyos archivos con datos de condiciones iniciales son los generados por [3] y utilizando el método de diferencia finita posterior.

6.3. Análisis de rendimiento

En este capítulo analizaremos el rendimiento de la herramienta mediante *profiling*. Por medio de la herramienta **gprof**[33] podemos observar en que llamadas a funciones se consume la mayor parte del tiempo de procesamiento.
Tabla de referencias: [1]

- **% time**: representa el porcentaje del tiempo total que se utilizó en una determinada función, incluyendo el tiempo utilizado en las llamadas a otras funciones desde esta función.
- **self**: representa la cantidad de tiempo utilizado en la función.
- **called**: representa el número de veces que se invocó a la función
- **name**: representa el nombre de la función.

```
 %     cumulative   self
 time   seconds   seconds    calls   name
 58.85     10.04    10.04 436703592  boost::numeric::ublas::
basic_row_major<unsigned int, int>::element(unsigned int, unsigned int,
unsigned int, unsigned int)
 13.42     12.33     2.29 336285320  boost::numeric::ublas::
matrix<float, boost::numeric::ublas::basic_row_major<unsigned int, int>,
boost::numeric::ublas::unbounded_array<float, std::allocator<float> > >::
operator()(unsigned int, unsigned int)
  4.10     13.03     0.70  2400000   FilterTransport::apply(Point)
  3.05     13.55     0.52 41583840   StrategyFiniteDifference::
diff(float(*)(Point const&), Point const&, Point const&, float)
  3.05     14.07     0.52  2400000   FilterRoughness::apply(Point)
  2.52     14.50     0.43     2462   OMatrix::OMatrix(unsigned int,
unsigned int)
  2.29     14.89     0.39 46099200   _eval_rho(Point const&)
  1.70     15.18     0.29 28486280   Filter::can_apply(Point const&, int)
  1.52     15.44     0.26  9219840   _eval_H_x(Point const&)
  1.41     15.68     0.24      900   EnviDatFile::EnviDatFile(QString&, int)
  1.11     15.87     0.19  9219840   _eval_H_y(Point const&)
  0.76     16.18     0.13 20791920   StrategyForwardDifference::
diffx(float(*)(Point const&), Point const&, unsigned int)
  0.70     16.30     0.12 20791920   StrategyForwardDifference::
diffy(float(*)(Point const&), Point const&, unsigned int)
  0.64     16.41     0.11      300   Filter::apply()
  0.59     16.51     0.10  4704480   _eval_m1(Point const&)
  0.47     16.59     0.08 16134720   gradient(float (*)(Point const&),
Point const&, int)
```

```
  0.35     16.65     0.06       60   ImageWriter::write_png_image(QString,
OMatrix&)
  0.29     16.70     0.05  4657200   divergence(float (*)(Point const&),
float (*)(Point const&), Point const&)
  0.29     16.75     0.05  2400000   FilterAttraction::apply(Point)
  0.23     16.79     0.04  4609920   _eval_m1(Point const&)
  0.23     16.83     0.04  4609920   _eval_m2(Point const&)
  0.23     16.87     0.04       60   MainWindow::write_result_density(OMatrix&,
QString)
  0.18     16.90     0.03  2400000   FilterBirths::apply(Point)
  0.18     16.93     0.03  2400000   FilterDeaths::apply(Point)
  0.18     16.96     0.03       60   OMatrix::get_maximum_value()
  0.12     16.98     0.02  4704480   _eval_m2(Point const&)
  0.12     17.00     0.02  2304960   ImageData::operator()(Point const&)
  0.12     17.02     0.02       60   HighPassFilter::apply(OMatrix&)
  0.12     17.04     0.02       60   OMatrix::operator*=(float)
  0.06     17.05     0.01  2304960   divergence(Pair&)
  0.06     17.06     0.01       60   OMatrix::get_minimum_value()
```

Los valores que se muestran arriba, corresponden a una hora de ejecución del modelo y las funciones que consumen el 85 % del tiempo son:

- $boost :: numeric :: ublas :: basic\_row\_major$
- $boost :: numeric :: ublas :: unbounded\_array < float, std :: allocator < float >>:: operator()(unsignedint, unsignedint)$
- $FilterTransport :: apply(Point)$
- $StrategyFiniteDifference :: diff(float(*)(Pointconst\&), Pointconst\&, Pointconst\&, float)$
- $FilterRoughness :: apply(Point)$

Se puede apreciar que la mayor parte del tiempo y las mayor cantidad de llamadas a función se consumen al utilizar el tipo de datos **matrix** provisto por las librerías Boost[20].
El tipo **matrix** se eligió para el almacenamiento de la información proveniente de los archivos de entrada. Una mejora en el rendimiento de esta herramienta sería utilizar otro tipo de estructura de dato para almacenar esa información.

Capítulo 7

Conclusión

Con este trabajo final hemos alcanzado una herramienta que incorpora resultados del procesamiento de imágenes satelitales para el estudio de la dispersión del Aedes Aegypti. El software desarrollado implementa el modelo propuesto por Annelise Trans en [1]. El programa genera dos tipos de resultados, uno visual y el otro de datos. El archivo de datos se genera para brindar la posibilidad de un análisis más profundo con herramientas mas especializadas como ENVI [7]. Por medio de este análisis se permite ver el valor de la densidad en cada pixel, mapas de colores, referencias, etc. El archivo de extensión png es el que se utilizará para brindar al usuario la posibilidad de apreciar la dispersión de los insectos mediante una animación. Dado el diseño de este software se puede modificar y ampliar fácilmente permitiendo su evolución junto al modelo propuesto. Si bien este trabajo final ha alcanzado su etapa final en el rol de trabajo de grado, aún quedan muchos aspectos por mejorar y ampliar. Una mejora posible sería agregar la creación de los archivos de condiciones iniciales utilizados para la simulación. Otra mejora sería poder visualizar los resultados del mismo algoritmo implementado en IDL y también mejorar el rendimiento del programa mediante la información provista en el capítulo *Análisis del rendimiento*.

Bibliografía

[1] M. Raffy and A.L. Tran. 'On the dynamics of flying insects populations controlled by large scale information.' Theoretical Population Biology, vol. 68. p.p. 91–104, 2005.

[2] Requerimientos funcionales y no funcionales - http://es.wikipedia.org/wiki/Requerimiento_funcional , http://es.wikipedia.org/wiki/Requerimiento_no_funcional

[3] Aedes aegypti spatio-temporal modelling based on non homogeneous environment derived from remote sensors information. S. Massuelli, C.H. Rotela, M. Lamfri, C.M Scavuzzo - 2007

[4] StarUML - http://staruml.sourceforge.net/docs/user-guide(en)/toc.html

[5] Qt 4.3 Trolltech - http://doc.trolltech.com/4.3/index.html

[6] Libpng - http://www.libpng.org/pub/png/libpng.html

[7] ENVI Version 3.5 The Environment for Visualizing Images. Copyright (C) 2001, Research Systems, Inc. http://www.ResearchSystems.com/envi

[8] Space-time analysis of the dengue spreading dynamics in the 2004 Tartagal outbreak, Northern Argentina. Camilo Rotela, Florence Fouque, Mario Lamfri, Phillipe Sabatier, Virginia Introini, Mario Zaidenberg, Carlos Scavuzzo.

[9] Teledetección ambiental - Emilio Chuvieco - Editorial Ariel, Mayo 2002.

[10] IDL Version 5.5 (C) Copyright 2001, Research Systems, Inc.

[11] Porcasi X., D.E. Gorla, C.M. Scavuzzo, M Lamfri, Remote sensing in the study of vectors of Chagas Disease: the case of Triatoma infestans"; Revista SELPER, "Sociedad Especialistas Latinoamericana en Percepción Remota", ISSN 0717-2915; V20 N1, pp 62-66 2005.

[12] Tran, A., 2004. Télédétection et Épidémiologie: Modélisation de la dynamique de populations dínsectes et application au contrôle de maladies à transmission vectorielle. Université Louis Paster Strasbourge France, pp200. http://eprints-scd-ulp.u-strasbg.fr:8080/249/

[13] Van Dyk, J.K., 2003. Mosquito Host-Seeking: a partial review Iowa State University Department of Entomology. http://www.ent.iastate.edu/dept/research/vandyk/hostseek.html

[14] B.L. Wood, L.R. Beck, R.K. Washino, S. Palchnick and P. Sebesta. "Spectral and spatial characterization of rice field mosquito habitat." Int J. Remote Sens, vol. 2, p.p. 621-6. 1991.

[15] J. Murray. "Spatial models and biomedical applications." Mathematical Biology II. 3rd Edition. Springer-Verlag, Berlin 2003.

[16] K.J. Linthicum, A. Anyamba, C.J. Tucker, P.W. Kelley, M.F. Myers and C.J. Peters. "Climate and satellite indicators to forecast Rift Valley fever epidemics in Kenya." Science, vol. 285, p.p. 397-400, 1999.

[17] U. Kitron. "Landscape ecology and epidemiology of vector-borne diseases: tools for spatial analysis." J Med Entomol, vol. 35, p.p. 435-45. 1998.

[18] J.R. Jensen. "Introductory Digital Image Processing." Prentice-Hall, New Jersey, p. 379. 1986.

[19] R.O. Hayes, E.L. Maxwell, C.J. Mitchell and T.L. Woodzick.. "Detection, identification and classification of mosquito larval habitats using remote sensing scanners in earth-orbiting satellites." Bull World Health Org; vol. 63, p.p. 361-74, 1985.

[20] Boost C++ Libraries - http://www.boost.org/ - http://www.boost.org/users/news/version_1_37_0

[21] C++ reference - http://www.cppreference.com/wiki/

[22] Brote de dengue en Argentina 2009 - http://es.wikipedia.org/wiki/Brote_de_dengue_en_Argentina_de_2009#Brote

[23] Aedes Aegypti - http://es.wikipedia.org/wiki/Aedes_aegypti#Riesgo_para_la_salud

[24] UML - http://es.wikipedia.org/wiki/UML

[25] Casos de uso - http://es.wikipedia.org/wiki/Casos_de_uso

[26] Espectro electromagnético - http://www.semar.gob.mx/ermexs/Temas %20percepcion.html.

[27] LandSat 5 - http://www.fas.org/irp/imint/docs/rst/Intro/Part2_20.html

[28] LandSat 7 - http://landsathandbook.gsfc.nasa.gov/handbook/handbook_htmls/chapter2/chapter2.html.

[29] SPOT 5 - http://www.planet-action.org/web/18graphicalchart.php

[30] EnviSAT - http://mtp.jpl.nasa.gov/missions/envisat/envisat.html

[31] SAC-C - www.conae.gov.ar

[32] Aedes Aegypti - http://www.imsc.res.in/~indu/JM/2009/JanFeb/SNews/Aedes_aegypti_CDC-Gathany.jpg.

[33] GNU gprof - http://sourceware.org/binutils/docs-2.16/gprof/

[34] Diagrama de clases: http://es.wikipedia.org/wiki/Diagrama_de_clases

[35] Diagrama de secuencia: http://es.wikipedia.org/wiki/Diagrama_de_secuencia

Apéndice

Especificaciones del sistema operativo

La herramienta desarrollada es multiplataforma, por lo tanto puede ser ejecutada en Linux o Windows. A continuación daremos las especificaciones que deberá tener el sistema para poder ejecutar esta herramienta.

Para poder ejecutar esta herramienta en un sistema operativo tipo Linux o Windows, se deberán instalar los siguientes componentes:

- Qt Open source Edition 4.4.3
- Librerías Boost C++ Version 1.37.0
- libpng - Portable Network Graphics (PNG) Reference Library 1.2.27

Una vez finalizada la instalación, se procederá a descomprimir el archivo **sddv_app.zip**. Al descomprimirlo, se creará una carpeta llamada **SDDV**, dentro de la cual se encontrará:

- El archivo ejecutable, llamado **sddv**
- El archivo de configuración **simulacion.config**
- Un directorio llamado **Escenarios**, en el cual estarán los archivos para la ejecución del modelo.
- Un directorio llamado **Resultados**, en el cual se almacenarán los archivos de salida del programa

Para proceder con la ejecución del programa desde un sistema operativo tipo Linux, se deberá acceder desde una consola hasta el directorio SDDV y ejecutar:

```
caro@victoria:~/SDDV$./sddv
```

Archivo de configuración: simulacion.config

Esta herramienta cuenta con un archivo de configuración, el cual se tomará para cargar los datos iniciales que luego podrán ser modificados por el usuario si fuera necesario. Este archivo de configuración por defecto es *simulacion.config*. Cualquier nombre de archivo con extensión .config y con el formato detallado más abajo, podrá utilizarse para la ejecución de la herramienta. El contenido del archivo de configuración deberá seguir el siguiente formato:

```
POBLACION_INICIAL=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/Mosquini
NACIMIENTOS=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/Nacimientos
MUERTES=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/Muertes
TENSOR_VIENTO=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/tensorW
TENSOR_RUGOSIDAD=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/tensorR
VIENTO=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/viento
ATRACCION=/home/caro/tesis/C++ CODE/Files/Pruebas2009/
Programa_15Selva_Final/CampoH
TIEMPO=48
DIFERENCIAFINITA=Diferencia finita posterior
DIRECTORIO_RESULTADOS=/home/caro/tesis/C++ CODE/resultados/
```

En POBLACION_INICIAL se deberá colocar la ruta del archivo que contiene la densidad inicial de mosquitos. En NACIMIENTOS se deberá colocar la ruta del archivo que contiene la densidad de nacimientos. En MUERTES se deberá colocar la ruta del archivo que contiene la densidad de muertes. En TENSOR_VIENTO se deberá colocar la ruta del archivo que contiene los valores del tensor de viento. En TENSOR_RUGOSIDAD se deberá colocar la ruta del archivo que contiene los valores del tensor de rugosidad del paisaje. En VIENTO se deberá colocar la ruta del archivo que contiene los valores de viento. En ATRACCION se deberá colocar la ruta del archivo que contiene los valores de las fuerzas atractivas. En TIEMPO, DIFERENCIAFINITA y DIRECTORIO_RESULTADOS se deberá colocar la cantidad de horas que se desea hacer la simualción, el tipo de diferencia finita que se utilizará y el directorio en el cual se almacenarán los resultados.

Nota: se debe tener en cuenta siempre que de acuerdo a la cantidad de horas que se desee hacer la simulación será la cantidad de información sobre nacimientos, muertes y viento que deben estar disponibles debido a que estos datos varían en el tiempo. Por ejemplo, en nuestro caso, los archivos de nacimientos y muertes son 48 bloques de 200 x 200 valores cada uno, mientras que el viento es un vector con 48 valores.

Configuración de datos de entrada

Para la configuración de esta herramienta, el usuario debe proveer un conjunto de archivos, llamados *condiciones iniciales* en los cuales están todos los valores necesarios para la realizar la simulación. Dichos archivos son:

- **Tensor de viento:** en este archivo, estarán los datos correspondientes al tensor de viento para una imágen sobre la cual se desee hacer la simulación.

- **Tensor de rozamiento**: en este archivo, estarán los datos correspondientes al tensor de rozamiento para la imágen sobre la cual se desee hacer la simulación. Dicho archivo es generado a partir de un archivo de clasificación.
- **Viento:** este archivo contiene los valores de la velocidad del viento correspondiente a la cantidad de horas que se desee hacer la simulación. Por ejemplo, si se desea hacer una simulación durante 48 hs, entonces en este archivo deberán estar todos los valores del viento para cada una de las horas de la simulación.
- **Nacimientos:** en este archivo estarán los datos correspondientes a los nacimientos de insectos para la imágen dada. Al igual que para el archivo del viento, en este caso, deberán estar presentes todos los valores para toda la cantidad de horas que se desee realizar la simulación. Esto también se debe a que la taza de nacimientos puede variar de acuerdo al tiempo.
- **Muertes:** este archivo contiene los valores correspondientes a las muertes de los insectos para la imágen dada. Este archivo tiene el mismo formato que el archivo de nacimientos.
- **Atractores:** en este archivo estarán los datos correspondientes a los **atractores**[1].
- **Densidad inicial:** en este archivo se encontrarán los datos correspondientes a la densidad de insectos inicial.

Para proceder con la configuración de la herramienta, el usuario debe seleccionar del menú principal: **Configuración** $->$ **Configurar Modelo**, y se le mostrará la siguiente pantalla:

Figura 1: Pantalla principal

En esta imágen, se puede apreciar que hay un campo configurable para cada nombre de archivo, la cantidad de horas que se desea ejecutar el modelo, el tipo de diferencia finita que se va a utilizar y el directorio donde se almacenarán los resultados de la simulación.

Ejecución

Una vez que hayan sido completados todos los campos de esta pantalla, el usuario deberá presionar **OK** para aceptar o **Cancel** para cancelar la operación.

[1]Llamaremos atractores a la presencia de humanos, animales, etc que representen una fuerza de atracción para los insectos.

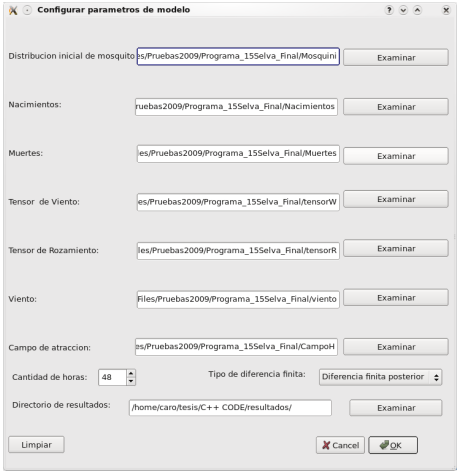

Figura 2: Pantalla de configuración del modelo

Si el usuario presionó **OK**, se volverá a la pantalla principal y presionará el botón **Ejecutar modelo.**

Figura 3: Ejecutar modelo

Visualización de los resultados.

Al finalizar la ejecución del model, se le notificará al usuario para que proceda a la visualización de los resultados:

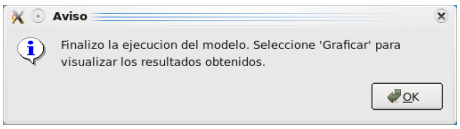

Figura 4: Finalizó ejecución

Para visualizar los resultados, el usuario deberá seleccionar la opción **Graficar** de la barra de herramientas

Figura 5: Graficar

A continuación, el usuario podrá apreciar la siguiente pantalla:

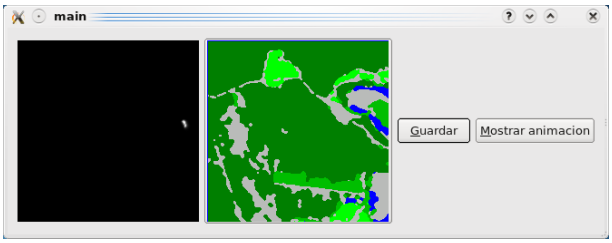

Figura 6: Visor de resultados

En esta pantalla, se muestra a la izquierda el resultado final de la ejecución y a la derecha el archivo de clasificaciones para esa ejecución. Este archivo puede ser seleccionado por el usuario presionando sobre la imágen.
Luego, si el usuario desea guardar el resultado obtenido, presionará el botón **Guardar**. Si el usuario desea ver una animación de los resultados obtenidos, desde la hora 1 hasta el final, deberá presionar **Visualizar animación**.