Una Alternativa Artificial, Drones Polinizadores:

Manuel Alejandro Gutiérrez Mejía
Universidad EAFIT
Colombia
magutierrm@eafit.edu.co

Kevin Alexander Herrera Garcés
Universidad EAFIT
Colombia
kaherrera@eafit.edu.co

Jose Joab Romero Humba
Universidad EAFIT
Colombia
jjromeroh@eafit.edu.com

Mauricio Toro Bermúdez
Universidad Eafit
Colombia
mtorobe@eafit.edu.co

RESUMEN:

Se sabe que las abejas le aportan mucho al planeta tierra, estos insectos que son la evolución de las avispas son los mayores agentes polinizadores de la tierra pero la exterminación de las abejas va en aumento debido al avance de la industrialización, el uso de ciertas toxinas que afectan al ecosistema de estas o por la simple ignorancia del hombre, por esto mismo varios grupos de investigación se han tomado la tarea de indagar e intentar crear prototipos de drones o robots los cuales logren cumplir de manera similar la tarea de estos organismos, polinizar nuestro ecosistema, pero esto traería una problemática, algunas abejas podrían colisionar y causar un serio problema en su sistema, por lo tanto se debe implementar un algoritmo que determine por cual lugar deben volar estos drones y de qué forma deben estar posicionados para no colisionar entre ellos. Estos algoritmos son muy utilizados en la creación de videojuegos y procesamiento de imágenes.

Palabras Claves: Algoritmos, Estructuras de datos, Colisión, Lista, Abejas.

Palabras Claves de la Clasificación de la ACM: Theory of computation→Design and analysis of algorithms→Data structures and algorithms → Collision detection

1. INTRODUCCIÓN:

En palabras de Einstein “Al hombre sólo le quedarían cuatro años de vida. Sin abejas, no hay polinización, ni hierba, ni animales, ni hombres”

Según la organización para la agricultura hay 100 especies de cultivos que proporcionan el 90% de los alimentos del mundo y 70 de ellos se polinizan con las abejas.

Se han notado en diversas ocasiones el decrecimiento de la población de abejas en diversos lugares del mundo a través de los años, como en USA en 2012 donde la población de abejas se redujo en un 60%; se a determinado gran parte de las causas por las cuales la población de esta especie se reduce, entre ellas se encuentran los pesticidas, parásitos y calentamiento global. Estas estadísticas permiten realizar de manera más clara los siguientes interrogantes:

- ¿Qué sucedería si desaparecen las abejas?
- ¿Se encuentra el hombre preparado para enfrentar la desaparición de esta especie?
- ¿Estamos a tiempo para remediarlo?

Diversas entidades han planteado soluciones a este problema, el objetivo de este documento es dar a conocer una posible solución desde la perspectiva de la tecnología y la programación con el propósito que va desde reducir el impacto ocasionado por este fenómeno hasta resolver en gran parte la problemática.

2. PROBLEMA:

La reducción descontrolada de abejas tendría como consecuencia efectos catastróficos para la vida de gran cantidad de especies (incluído el hombre) .

Para contrarrestar los grandes efectos adversos que causaría el decrecimiento drástico de la población de esta especie es necesario (entre otras cosas) implementar un método o sistema que realice la polinización de las diferentes especies de cultivos que dependen de esta especie.

Se ha pensado en emplear drones que ayuden con la polinización, pero se necesita un algoritmo rápido y eficiente que regule el movimiento de los mismos y detecte las posibles colisiones entre ellos

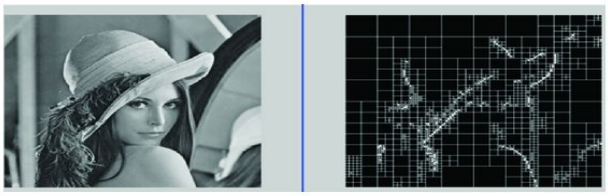
3.TRABAJOS RELACIONADOS:

3.1 Multi-focus image fusion based on edges and focused region extraction:

“Multi focus image” es un algoritmo propuesto por Juanxiu Tian, Guacoi Liu y Jingguang Liu en 2018, la idea del algoritmo es fusionar múltiples imágenes parcialmente enfocadas en una imagen más nítida. Una de las claves para la fusión de imágenes es cómo detectar las regiones enfocadas. Cuando se trata de fusionar imágenes se debe tener en cuenta los tipos de borde que presenta la imagen, cuando se trata de bordes enfocados salientes (Los cuales solo existen en regiones enfocadas) se debe tener en cuenta el tipo de umbral con el cual vamos a actuar y la región de la imagen en la cual utilizaremos nuestra estructura del quadtree. Para el segundo tipo de bordes es el sintetizado, el cual utiliza para refinar los límites de las regiones enfocadas, con las regiones fusionadas los experimentos mostrados por los autores muestran que la propuesta del algoritmo puede extraer regiones enfocadas con los límites adecuados. Por lo tanto, las imágenes fusionadas

pueden evitar los artefactos de distorsión y preserve la nitidez de los objetos enfocados.

Imagen 1: Algoritmo de fusión de imágenes



3.2 Use Quadtrees to Detect Likely Collisions in 2D Space:

Muchos juegos requieren el uso de algoritmos de detección de colisión para determinar cuándo dos objetos han colisionado, pero estos algoritmos son a menudo operaciones costosas y pueden ralentizar en gran medida un juego. La detección de colisiones es una parte esencial de la mayoría de los videojuegos. Tanto en los juegos en 2D como en 3D, es importante detectar cuándo chocaron dos objetos, ya que una detección de colisión deficiente puede dar lugar a algunos resultados muy interesantes, una solución a esta problemática es la implementación del quadtree. Los Quadtrees son una forma de ayudar a acelerar la detección de colisiones y mantener el juego funcionando a una velocidad adecuada.

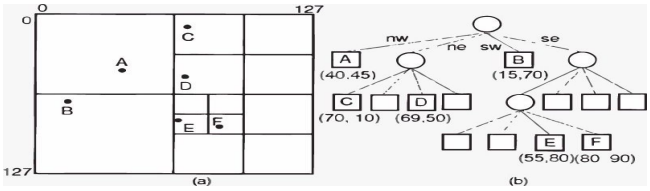


Imagen 2: Metodología de los quadtrees

3.3 TCAS (Traffic Collision Avoidance System):

Anteriormente era una labor complicada el tema de la prevención de colisiones entre aviones y cómo actuar en caso de que haya peligro de una. para ello se desarrolló el sistema TCAS que alerta sobre tráficos cercanos a los pilotos con el fin de evitar, como su propio nombre indica, colisiones y accidentes y además propone una maniobra evasiva en el plano vertical (ascienda o descienda) opuesta en cada avión

Se podría decir que el desarrollo de este sistema comienza en el año 1955 cuando el DR J. S Morell de Bendix Avionics publicó en su ensayo "La Física de las colisiones" un algoritmo computable que relacionaba y definía diferentes proporciones entre el avión que se acerca y el avión amenazado. Este trabajo fue la base para todas las investigaciones posteriores que pretenden diseñar un sistema para evitar colisiones aéreas

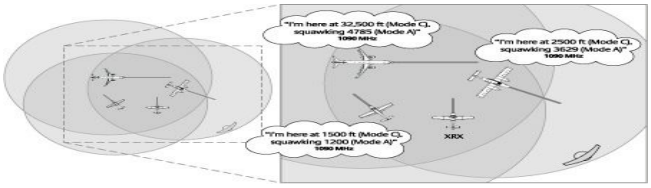


Imagen 3: TCAS, prevención de colisiones

3.4 UTM (Unmanned aircraft system Traffic Management):

UTM es un algoritmo que ayudará a solucionar el problema de la gestión de tráfico para drones. tiene la intención de cumplir con muchas de las funciones de control del tráfico aéreo, pero radicará en la nube y será en gran parte automatizado. este tendrá como objetivo la prevención de accidentes, alertar al dron cuando una colisión es posible y calcular las maniobras necesarias para evitarlas.

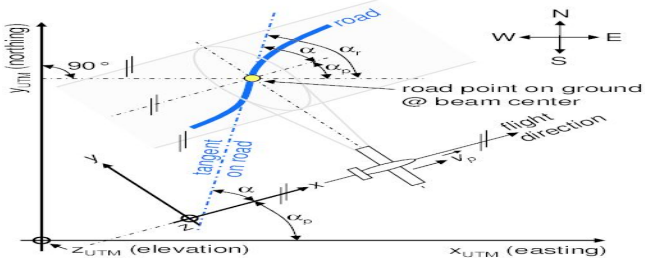


Imagen 4: UTM, sistema de tráfico de drones

4. Estructura de Datos (primer diseño): ArrayList

Gráfica 1: ArrayList implementado

4.1 Criterios de Diseño de la estructura de datos ArrayList:

La razón por la cual se eligió el arrayList, fue debido a las siguientes razones:

Inicialmente al tener una capacidad dinámica, no se es necesario tener que conocer exactamente la cantidad de abejas que hay para poder almacenarlas en el mismo. De la misma manera, debido a que siempre los datos se están almacenando en la última posición, la complejidad que requiere esto será de O(1) por lo que no hay problema en su rendimiento y no afecta a la complejidad del programa.

Además, también facilita acceder a una ubicación de la lista, teniendo este procedimiento una complejidad de O(1).

La complejidad para el peor de los casos de los métodos que se implementaron es O(n^2)

4.2 Tiempos de ejecución:

Tabla 1: Tiempos de Ejecución

4.3 Memoria:

Operación	Hash map
10 abejas.	10.4ms
100 abejas.	23ms
1000 abejas.	78.8ms
10000 abejas.	214.4ms
100000 abejas.	611 ms

Tabla 2: Consumo de memoria

Operación	Hash map
10 abejas.	0 MB
100 abejas.	1 MB
1000 abejas.	2 MB
10000 abejas.	12 MB
100000 abejas.	16.8 MB

4.4 Análisis de los Resultados:

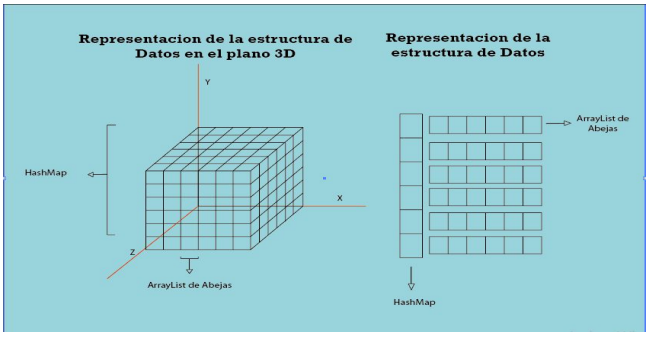
Tabla 3: Resultados de la implementación; promedios, máximos y mínimo del tiempo.

Cantidad	Mejor Tiempo	Peor Tiempo	Tiempo Promedio
10 Abejas	7 ms	15 ms	10.4 ms
100Abejas	17 ms	30 ms	23 ms
1000 Abejas	63 ms	95 ms	78.8 ms
10000 Abejas	174 ms	253 ms	214.ms
100000 Abejas	533 ms	709 ms	612 ms

Tabla 4: Resultados de la implementación; promedios, máximos y mínimos de la memoria.

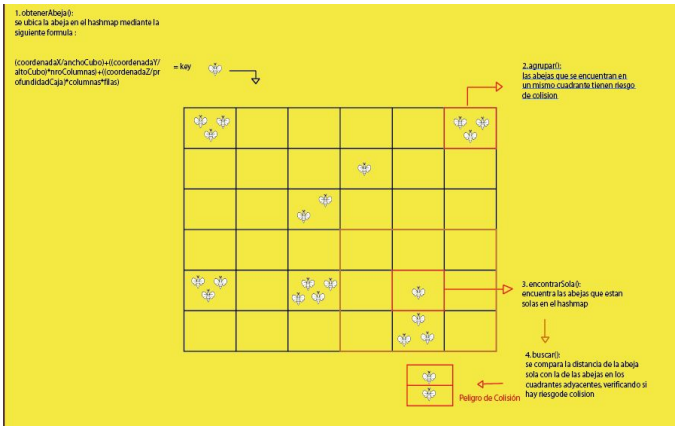
Cantidad	Mejor Memoria	Peor Memoria	Memoria Promedio
10 Abejas	0MB	0MB	0MB
100Abejas	1MB	1MB	1MB
1000 Abejas	2MB	2MB	2MB
10000 Abejas	12MB	12MB	12MB
100000 Abejas	15MB	17MB	16.8MB

5 Hash Map y ArrayList:



Grafica 2: representacion HashTable

5.1 Operaciones:



Grafica 3: métodos implementados

5.2 Criterios de Diseño:

Sabiendo que el objetivo consiste en averiguar de manera rápida y eficiente qué abejas están en riesgo de colisionar entre ellas se decidió emplear una HashTable (Hash map en java) puesto que sus aplicaciones y funciones encajaban perfectamente con lo necesitado, sus métodos eficientes y de bajo consumo de capacidad posibilitaron un análisis más dinámico, dado que:

Cuando se divide el mapa de acuerdo al rango de choque es posible ubicar a las abejas (con ayuda de una fórmula que analiza su posición) en uno de los sectores y de esta manera las abejas que compartan sector permiten concluir que se encuentra en riesgo de choque.

El anterior proceso cuenta con gran eficiencia puesto que el HashTable permite realizarlo fácilmente y finalmente para el almacenamiento de las abejas se emplea un ArrayList dado que almacenar al final y acceder a un elemento no conllevan complejidad alguna.

5.3. Análisis de Complejidad:

Metodo	Complejidad
obtenerAbeja()	O(n)
agrupar()	O(h*n)
encontrarSola()	O(n)
buscar()	O(n)

Tabla 4: complejidad de los métodos, donde H es el tamaño del hashTable y N el tamaño del ArrayList en esta posición.

6. Conclusiones:

- Es indispensable elegir una estructura de datos correcta para cada problema, dado que la eficiencia que pueden proporcionar es notable en los resultados.
- Con respecto al mundo de la algoritmia, existe una gran cantidad de problemas con relación a la detección de colisiones, el enfoque principal de este documento fue como realizar un algoritmo de manera eficiente para un problema de esta índole
- Al implementar una mejor estructura de datos se han obtenidos resultados con una mejor significativa en cuanto a consumo de memoria, tiempo de ejecución y eficiencia

Agradecimientos:

-Nosotros agradecemos especialmente a Ricardo Rafael Azopardo Cardenas por la ayuda en el diseño de las clases y la estructura de datos para nuestro código.

-Nosotros agradecemos a las entidades que pagan nuestros estudios por permitirnos realizar esta investigación.

REFERENCIAS:

1. Tian, J., Liu, G. and Liu, J., 2018. *Multi-focus image fusion based on edges and focused region extraction* 1st ed.

2 Lambert, S., 2018. Quick Tip: Use Quadtrees to Detect Likely Collisions in 2D Space. *Game Development Envato Tuts+*.

3.cienciaplus, 2018. Un algoritmo regulará el vuelo de drones sin controladores humanos. *europapress.es*.

4 etsist, 2018. TCAS (Traffic alert and Collision Avoidance System). *ingeniatic*.