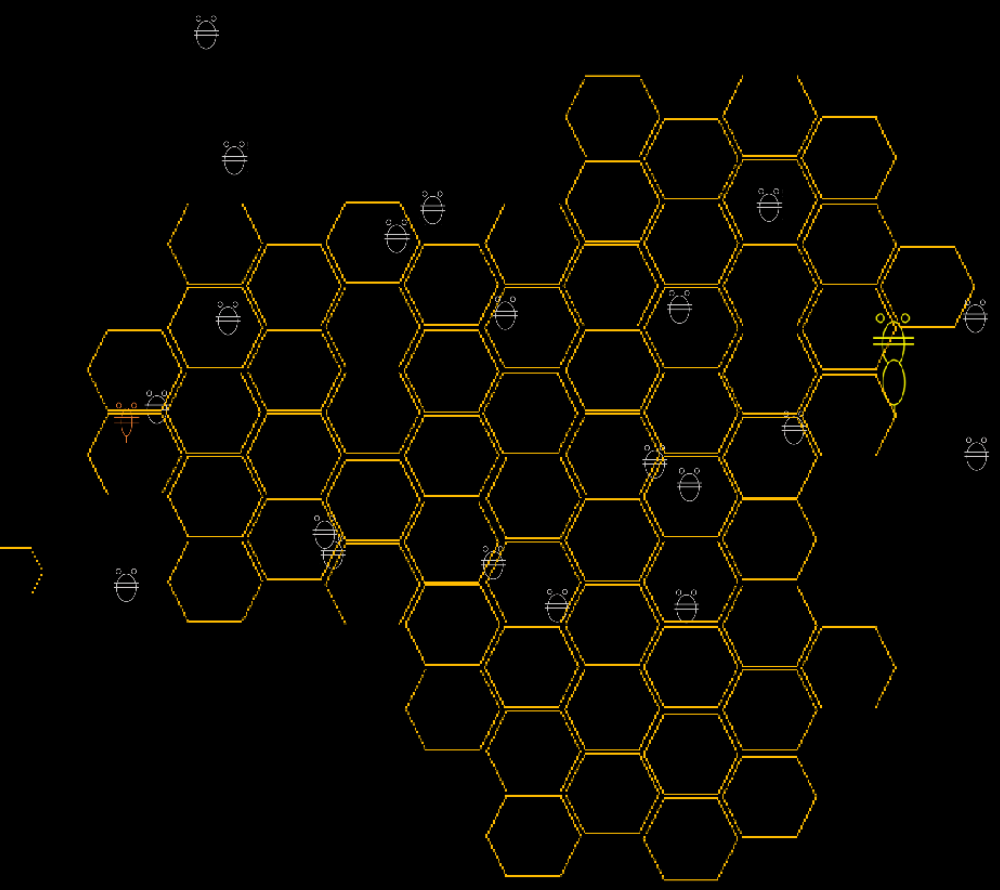
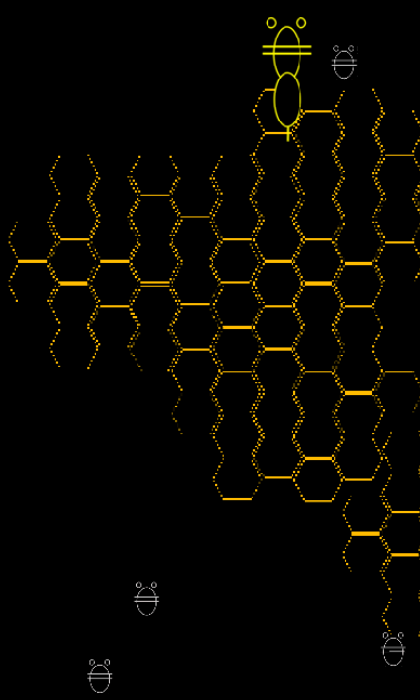
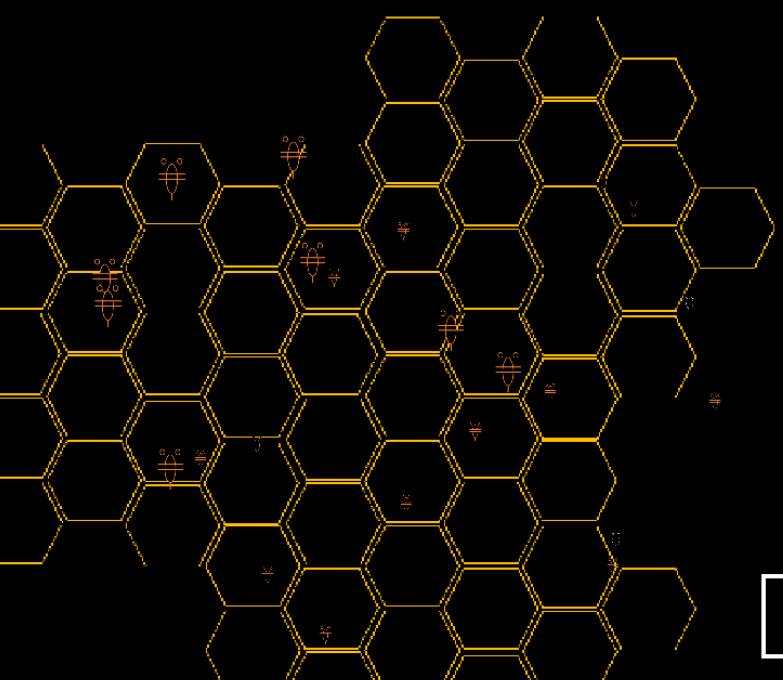
Sección central







mouse to pan, wheel to zoom

ESC F C S O U D

Zumbido Digital. Explorando el bio arte con un ecosistema de abejas virtuales

Artículo de investigación

Recibido: 03 de junio de 2024
Aprobado: 26 de junio de 2024

Eddie Jonathan García Borbón
Universidad Federal del Extremo Oriente (DVFU)
Rusia
garsiyaborbon.ed@dvfu.ru

—

Cómo citar este artículo: García Borbón, E.J. (2024). Zumbido Digital. Explorando el bio arte con un ecosistema de abejas virtuales. *Estudios Artísticos: revista de investigación creadora, 10*(17), pp. 102-127
DOI: https://doi.org/10.14483/25009311.22346

Resumen

El proyecto *Zumbido Digital*: Explorando el bio arte *con un ecosistema* de abejas virtuales." combina el arte, la entomología y la tecnología para crear un ecosistema digital que simula el comportamiento de las abejas. Desarrollado en la plataforma nota.space y galardonado con el NOTAnEAR AWARDS en el International Digitalkunst Festival de Stuttgart 2022, el proyecto de investigación – creación se basa en estudios entomológicos detallados de la morfología, locomoción y hábitat de las abejas. Utilizando algoritmos de caminata aleatoria y técnicas de programación orientada a objetos, se simularon los roles de abeja reina, obreras y zánganos. El prototipo de programación está orientado hacia una futura investigación sobre la simulación de ecosistemas digitales y sistemas autoorganizados, destacando la capacidad del bio arte para explorar la interactividad digital y las propiedades emergentes.

Palabras clave

Bio Arte, Arte y Vida Artificial, Ecosistema Digital, Simulación Biológica, Nota.Space

Digital Buzz. Exploring bio art with a virtual bee ecosystem

~~Abstract~~

The study "Digital Buzz: Exploring Bio Art with a Virtual Bee Ecosystem" combines art, entomology, and technology to create a digital ecosystem that simulates bee behavior. This research was developed on the nota.space platform and received the NOTAnEAR AWARDS at the International Digitalkunst Festival in Stuttgart 2022. Based on thorough entomological research, this study focuses on bee morphology, movement, and natural habitat. It utilizes random walking algorithms and object-oriented programming techniques, along with simulations of the

<

Ilustración de una parte de la colmena digital en la plataforma Nota.Space. Las abejas en amarillo son abejas reina y las de color blanco son abejas zangano. Elaboración propia.

behaviors of queen bees, worker bees, and drones. The programming prototype is oriented towards future research on simulating digital ecosystems and self-organized systems, highlighting the capacity of bio art to explore digital interactivity and emergent properties.

Palabras clave

Bio Art, Art and Artificial Life, Digital Ecosystem, Biological Simulation, Nota.Space

Le buzz numérique. Explorer l'art biologique avec un écosystème d'abeilles virtuelles

Résumé

Le projet Digital *Buzz* : "*explorer le* bio-art *avec* un écosystème virtuel d'abeilles" combine l'art, l'entomologie et la technologie pour créer un écosystème numérique qui simule le comportement des abeilles. Développé sur la plateforme nota.space et récompensé par les NOTAnEAR AWARDS au Festival international Digitalkunst de Stuttgart en 2022, le projet de recherche-création s'appuie sur des études entomologiques détaillées de la morphologie, de la locomotion et de l'habitat des abeilles. À l'aide d'algorithmes de marche aléatoire et de techniques de programmation orientées objet, les rôles des reines d'abeilles, des ouvrières et des drones ont été simulés. Le prototype de programmation est orienté vers de futures recherches sur la simulation d'écosystèmes numériques et de systèmes auto-organisés, mettant en évidence la capacité du bio-art à explorer l'interactivité numérique et les propriétés émergentes.

Mots-clés

Bio art, Art et vie artificielle, Écosystème numérique, Simulation biologique, Nota.Space

Buzz digital. Explorando a bioarte com um ecossistema de abelhas virtuais

Resumo

O projeto "*Digital Buzz: Explorando a* bioarte *com um* ecossistema de abelhas virtuais" combina arte, entomologia e tecnologia para criar um ecossistema digital que simula o comportamento das abelhas. Desenvolvido na plataforma nota.space e premiado com os PRÊMIOS NOTAnEAR no Festival Internacional Digitalkunst em Stuttgart 2022, o projeto de pesquisa-criação é baseado em estudos entomológicos detalhados da morfologia, locomoção e habitat das abelhas. Usando algoritmos de caminhada aleatória e técnicas de programação orientada a objetos, os papéis das abelhas rainhas, operárias e drones foram simulados. O protótipo de programação é voltado para pesquisas futuras sobre a simulação de ecossistemas digitais e sistemas auto-organizados, destacando a capacidade da bioarte de explorar a interatividade digital e propriedades emergentes.

Palavras chave

Bio Arte, Arte e Vida Artificial, Ecossistema Digital, Simulação Biológica, Nota.Space

Uiariikuna ialichispa iachaikuspa kuspa ima nukanchi kausaskapi tiaskawa kawachiku abejakuna

Maillallachiska

Kaipi tandariskakuna kawachingapa " Uiariikuna ialichispa iachaikuspa kuspa ima nukanchi kausaskapi tiaskawa kawachiku abejakuna" chapunaku imasam kai abejakuna nukanchita aidachinkuna i chasallata kawachiikuna tiami katichingapa imasa; kai abejakuna kausankuna, churainakumi sug kawachidurupi NOTA EART AWARDS kai kanchanimanda kunapas Digitalkunst festival Suttgart 2022, kai kilkaiwa munanakumi parlanga imasam kai abejakuna, kankuna tandarispa kai ukupi maki kuarinkuna, paipura chasallata munaku runakunata ningapa tandarisunchi Nukanchipa manda, llakichinakuspalla chasa sumaglla kai wachukuna llugsisunchi.

Rimangapa Ministidukuna

Bio arte, ima tiaskata kausachii, iuiariskata kawachii tukuikunamanda, kausaskatasina kawachii, kilkaska sakii kawangapa

La vida como la conocemos no tiene por qué ser la única forma de vida.
(Capitán Kirk, Star Trek)

Introducción

En el vasto universo de la imaginación humana, una pregunta resuena con particular fascinación: ¿Es posible la vida en formas no basadas en carbono? Esta interrogante, que ha cautivado a científicos, escritores y soñadores por igual, es el motor impulsor detrás del innovador proyecto *Zumbido Digital: Explorando el* bio a*rte con un ecosistema de* abejas virtuales". Inspirado por las conjeturas de la ciencia ficción y los ensayos visionarios sobre la naturaleza de la vida, este proyecto multidisciplinario fusiona arte, entomología y tecnología en un viaje de exploración hacia los límites de lo imaginable.

Tomando como referencia obras emblemáticas del cine como *Star Trek* y *Solaris* (Lem, 2002), que desafían la idea de vida extraterrestre en formas desconocidas y exploran la comunicación inter-especies en entornos alienígenas, respectivamente, *Zumbido Digital* se inspira en la exploración de lo desconocido y en las posibilidades de encuentro con formas de vida inspiradas o imaginadas a partir de las abejas terrestres.

Además, el proyecto se nutre del trabajo pionero de Erwin Schrödinger (1944) especialmente su ensayo *¿Qué es la vida?*, él se preguntaba cosas muy interesantes sobre cómo funciona la vida. Imaginaba que, al igual que los juguetes pueden moverse gracias a las pilas, nuestros cuerpos también funcionan gracias a una especie de energía muy pequeña. Pero lo más sorprendente es que Schrödinger pensaba que tal vez no sólo nosotros, sino otras cosas en el universo, podrían estar vivas de una manera diferente, no necesariamente igual que nosotros. Esto nos hace pensar que la vida podría ser mucho más variada de lo que imaginamos. Las ideas de Schrödinger nos hacen reflexionar sobre lo que realmente significa estar vivo y nos inspiran a explorar nuevas posibilidades, incluso formas de vida que no están hechas de los mismos materiales que nosotros.

Al fusionar estas influencias literarias con la investigación científica y el arte experimental, *Zumbido Digital* se erige como un puente entre la especulación y la realidad, desafiando nuestras percepciones sobre la vida, el arte y la ciencia, y abriendo nuevas fronteras en la exploración del concepto de vida en todas sus formas y manifestaciones.

Marco Teórico

En este capítulo se exploran conceptos fundamentales que abarcan desde la intersección entre el arte y la ciencia hasta la aplicación de técnicas computacionales avanzadas. En primer lugar, se aborda el campo del A-Life Art, donde se fusionan los principios de la vida artificial con expresiones artísticas para crear obras que exploran la naturaleza de la vida y sus manifestaciones en entornos digitales. Además, se profundiza en el estudio de ecosistemas digitales, que son simulaciones computacionales de sistemas naturales que permiten comprender mejor su funcionamiento y comportamiento emergente. Se examina también el concepto de caminata aleatoria, una técnica utilizada para modelar el movimiento de individuos en entornos sin un patrón predefinido, como las abejas en su búsqueda de néctar. Asimismo, se incorpora el campo de la entomología y la morfología, que estudian respectivamente la biología de los insectos y su estructura física, aspectos fundamentales para comprender y recrear digitalmente el comportamiento de las abejas en un ecosistema virtual. Estos conceptos proporcionan el fundamento teórico necesario para el desarrollo y la comprensión del proyecto *Zumbido Digital*, que busca explorar la complejidad y la interacción de la vida en un entorno digitalmente creado.

A-Life Art

El término *A-Life Art* se refiere a una interdisciplinariedad que combina los principios y métodos de la vida artificial con la expresión artística, con el fin de explorar y representar procesos y fenómenos relacionados con la vida en entornos virtuales o generados por computadora. Esta práctica artística se basa en la idea de crear simulaciones computacionales o sistemas autónomos que exhiben comportamientos emergentes, imitan procesos biológicos o generan formas de vida virtual (Penny, 2010). El

arte de la vida artificial busca no sólo entender mejor los fundamentos de la vida, sino también cuestionar y reflexionar sobre la naturaleza de la existencia, la evolución y la interacción entre lo orgánico y lo sintético. Este enfoque artístico abre un espacio para la experimentación creativa y la exploración de conceptos complejos relacionados con la vida y la evolución en un contexto digital. Las obras de *a-life art* pueden manifestarse en una variedad de formas, como instalaciones interactivas, visualizaciones computacionales, esculturas generativas y experiencias multimedia, ofreciendo nuevas perspectivas sobre la relación entre el arte, la ciencia y la tecnología. (Greenfield, 2021).

En el artículo *A Survey of Recent Practice of Artificial Life in Visual Art* de Wu, Zi-Wei, Huamin Qu, y Kang Zhang (2024) se examinan diversas prácticas contemporáneas de vida artificial en el arte visual. Un ejemplo que se podría mencionar es la obra *Bloom* del artista Jon McCormack (1993). *Bloom* es una instalación interactiva que emplea algoritmos inspirados en procesos biológicos para generar formas de vida digital en tiempo real. Los espectadores pueden interactuar con la obra a través de pantallas táctiles, observando cómo las formas de vida virtuales crecen, evolucionan y se transforman en respuesta a sus acciones. Esta obra combina la ciencia de la vida artificial con la expresión artística para ofrecer una experiencia inmersiva que invita a reflexionar sobre la naturaleza de la vida, la evolución y la interacción entre lo orgánico y lo sintético (McCormack, 2008).

A diferencia del clásico *Juego de la Vida* de John Conway (1970), que es un autómata celular donde las células en una cuadrícula viven, mueren o se reproducen según reglas matemáticas simples (Izhikevich, Conway, & Seth, 2015), Bloom utiliza algoritmos más complejos inspirados en procesos biológicos reales. Mientras que el *Juego de la Vida* de Conway demuestra cómo reglas simples pueden generar patrones complejos y autoorganizados, *Bloom* ofrece un nivel de interactividad y evolución en tiempo real, permitiendo a los espectadores influir directamente en el desarrollo de las formas de vida digitales. Las acciones de los espectadores en Bloom no sólo afectan la existencia inmediata de los organismos virtuales, sino que también pueden provocar cambios evolutivos a largo plazo, creando una experiencia dinámica y reflexiva sobre la naturaleza de la vida y la evolución. Esta capacidad de interactuar y observar las consecuencias de sus acciones en tiempo real fomenta una reflexión profunda sobre la naturaleza de la vida, su evolución y la relación entre lo orgánico y lo sintético.

Ecosistemas digitales

En el contexto del *A Life-Art*, un ecosistema digital se refiere a un entorno simulado por computadora que imita las interacciones y relaciones entre organismos virtuales o agentes autónomos. Estos ecosistemas digitales son diseñados por artistas y científicos para reflejar aspectos específicos de la biología, la ecología o la evolución, y son utilizados como medio de exploración creativa y expresión artística (Bartlem, 2005).

Según *Estética, vida artificial y biopolítica*, un ecosistema digital se define como un entorno donde convergen arte, tecnología y biología para explorar nuevas formas de expresión y evolución cultural. En este contexto, el arte del A-Life (vida artificial) se presenta como una disciplina que utiliza algoritmos inspirados en procesos biológicos para crear obras interactivas y dinámicas. Estas obras no solo desafían las convenciones artísticas tradicionales, sino que también simulan dinámicas emergentes similares a las observadas en la naturaleza, explorando así los límites y las posibilidades de la interacción entre lo orgánico y lo sintético (Hernández & Niño, 2010).

Por ejemplo, algunos proyectos de a-life art se centran en la producción de paisajes abstractos o morfologías en constante evolución utilizando técnicas como algoritmos genéticos o geometría fractal (Padua, 2021). Estos paisajes digitales evocan imágenes de la naturaleza y exhiben comportamientos emergentes similares a los observados en los ecosistemas reales.

Otros proyectos de a-life art hacen referencia a sistemas naturales más reconocibles, como células, organismos o criaturas extrañas, habitando un entorno virtual. Estos ecosistemas digitales suelen ser interactivos, permitiendo al usuario interactuar con el sistema y afectar su evolución. Por ejemplo,

un proyecto llamado *Diseased Squares* presenta un modelo autónomo donde los organismos virtuales nacen, se desarrollan, se reproducen, mueren y se descomponen con el tiempo, todo mientras interactúan con el usuario a través de cambios cromáticos y sonidos (Dorin, 2008).

La interactividad suele ser un aspecto central en el diseño de estos ecosistemas artificiales. Algunos proyectos, como *A-volve* de Christa Sommerer y Laurent Mignonneau, 2023, permiten a los usuarios crear y modificar criaturas virtuales, mientras que otros, como *EIDEA* de John D. Mitchell y Robb E. Lovelles (García Oliver, 2012), conectan el ecosistema virtual con el entorno físico del usuario, utilizando datos meteorológicos reales para influir en el comportamiento de los organismos virtuales.

Finalmente, en el contexto latinoamericano, proyectos como *Simbiosis* de Eddie Jonathan García Borbón (2018) y *Del suspiro en el alba hasta el abrazo en el ocaso: una composición evolutiva* de Luis Fernando Sánchez Gooding (2018) representan una evolución fascinante en la intersección entre el arte, la tecnología y la biología. En estas obras, la relación simbiótica entre el ser humano y la máquina se convierte en el núcleo de una experiencia creativa innovadora. Ambas composiciones han concebido un entorno musical donde un director/intérprete (mentor) ejerce control en tiempo real sobre la interacción entre un intérprete humano y una población de entidades sonoras autónomas (Gooding, Borbón, & Sánchez, 2018).

Estas obras trascienden los límites convencionales de la música al integrar elementos de inteligencia artificial y computación evolutiva en su proceso creativo. El director/intérprete actúa como mediador entre el mundo humano y el digital, coordinando una sinfonía de sonidos generados por las entidades sonoras autónomas. Estas entidades, influenciadas por algoritmos de aprendizaje automático y estrategias bio-inspiradas, interactúan dinámicamente con el intérprete humano y su entorno sonoro. La indeterminación no sólo redefine las estructuras musicales convencionales, sino que también simula dinámicas biológicas emergentes, donde la interacción entre lo humano y lo digital crea un espacio sonoro fluido y participativo, explorando nuevos horizontes de expresión creativa en el arte del A-life.

La integración de la simbiosis en este contexto ofrece un nuevo horizonte de posibilidades creativas y expresivas dentro del campo del a-life art. Al permitir que el intérprete humano colabore y se relacione directamente con entidades sonoras autónomas, *Simbiosis* proporciona una experiencia inmersiva y participativa única. Los límites entre lo orgánico y lo artificial se desdibujan, creando un espacio donde la creatividad humana y la inteligencia digital convergen para dar forma a una experiencia sonora sin precedentes.

Caminata aleatoria

La caminata aleatoria es un concepto fundamental en la teoría de procesos estocásticos y se utiliza para modelar el movimiento aleatorio de partículas u organismos en diversas disciplinas, incluyendo la biología, la física, la economía y la informática. Este proceso describe una trayectoria que consiste en una serie de pasos aleatorios, donde la dirección y, en algunos casos, la longitud de cada paso se determina de manera aleatoria (Lawler, 2010).

Definición y conceptos básicos

- *Pasos aleatorios*: cada paso en una caminata aleatoria es independiente y puede tener igual probabilidad de ocurrir en cualquier dirección. En un espacio bidimensional, esto generalmente significa que, en cada punto, el agente puede moverse hacia arriba, abajo, izquierda o derecha con igual probabilidad (Grinstead, 2012).

- *Independencia de pasos:* la independencia de cada paso significa que la trayectoria futura del agente no depende de su trayectoria pasada. Este carácter independiente produce rutas impredecibles y aparentemente erráticas (Feller, 1968).

Aplicaciones y relevancia

- *Modelado de comportamientos naturales*: La caminata aleatoria se utiliza para modelar una variedad de comportamientos naturales, como el movimiento de animales en busca de alimento, la difusión de partículas en un fluido y el crecimiento

de estructuras biológicas. En biología, por ejemplo, puede modelar cómo las bacterias exploran su entorno en busca de nutrientes (Codling, Plank, & Benhamou, 2008; Berg, 1983).

- *Simulación de procesos evolutivos*: En el estudio de la evolución y la ecología, la caminata aleatoria puede representar las trayectorias genéticas de poblaciones a través del tiempo, mostrando cómo las variaciones aleatorias pueden llevar a la diversidad genética (Kimura, 1983; Ewens, 2004).

- *Análisis financiero:* En economía y finanzas, se utiliza para modelar las fluctuaciones de los precios de las acciones y otros activos financieros, bajo la suposición de que los cambios en los precios son aleatorios e independientes (Malkiel, 1973; Fama, 1965).

Entomología

La entomología es la rama de la biología que se dedica al estudio de los insectos. Como una subdisciplina dentro de la zoología, la entomología abarca una amplia gama de aspectos relacionados con los insectos, incluyendo su clasificación, morfología, comportamiento, ecología, evolución y relación con los humanos (Gullan & Cranston, 2014). A continuación, se enumeran algunas áreas de estudio:

- Taxonomía y clasificación: La entomología incluye la identificación y clasificación de los insectos en órdenes, familias, géneros y especies. Este proceso se basa en características morfológicas, genéticas y de comportamiento (Triplehorn & Johnson, 2005).
- Morfología: La morfología se centra en la estructura física de los insectos, incluyendo su anatomía interna y externa. Los entomólogos estudian las partes del cuerpo de los insectos, como las alas, antenas, patas y sistemas internos, para comprender sus funciones y adaptaciones (Snodgrass, 1935).
- Comportamiento: El estudio del comportamiento de los insectos abarca aspectos como la alimentación, el apareamiento, la comunicación, la migración y las estrategias de defensa. Este conocimiento es crucial para entender cómo los insectos interactúan con su entorno y otras especies (Chapman, 2013).
- Ecología: La ecología entomológica investiga las interacciones de los insectos con su entorno, incluyendo sus roles en los ecosistemas, sus relaciones con las plantas y otros animales, y su impacto en los ciclos de nutrientes y la polinización (Speight, Hunter, & Watt, 2008).
- Evolución: La entomología también explora la evolución de los insectos, analizando cómo han cambiado y diversificado a lo largo del tiempo. Esto incluye el estudio de fósiles de insectos y la utilización de técnicas de biología molecular para entender su historia evolutiva (Grimaldi & Engel, 2005).
- Entomología médica y veterinaria: Esta subdisciplina se ocupa de los insectos que afectan la salud humana y animal. Incluye el estudio de insectos vectores de enfermedades, como mosquitos que transmiten malaria, dengue y zika, así como piojos, pulgas y garrapatas (Service, 2012).
- Entomología agrícola: Se centra en los insectos que afectan la agricultura, ya sea como plagas que dañan los cultivos o como agentes de control biológico que ayudan a manejar las poblaciones de plagas (Pedigo & Rice, 2014).

Plataforma Nota.Space

Para la realización del proyecto *Zumbido* digital", se ha utilizado la plataforma Nota.Space, una herramienta colaborativa en línea diseñada para facilitar la creación y gestión de proyectos artísticos y científicos. Nota.Space ofrece un entorno virtual que permite a los usuarios desarrollar y experimentar con entornos interactivos y simulaciones complejas.

- Entorno de desarrollo: Nota.Space proporciona un entorno de desarrollo integrado (IDE) que soporta múltiples lenguajes de programación, incluidos Python, JavaScript y *Processing*[1]. Esto permite una gran flexibilidad a la hora de

1 *Processing* y p5.js son entornos de programación y bibliotecas de código abierto utilizados para la creación de proyectos interactivos en el ámbito del arte digital y la educación creativa. Processing se centra en el desarrollo de aplicaciones visuales y gráficas, mientras que p5.js, basado en JavaScript, extiende estos conceptos para la web y permite crear obras interactivas directamente en el navegador. «https://editor.p5js.org/»

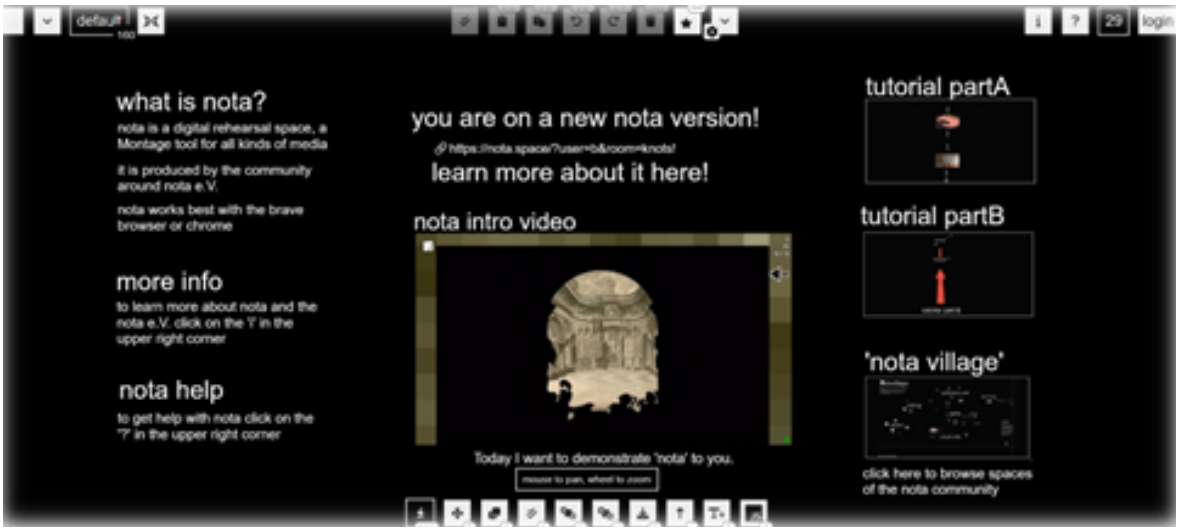


Imagen 1. Interfaz de usuario de la plataforma Nota.Space, una herramienta colaborativa en línea diseñada para la creación y gestión de proyectos artísticos y científicos. «https://nota.space/»

programar las interacciones y comportamientos de las abejas virtuales.

- Colaboración en tiempo real: La plataforma permite la colaboración en tiempo real entre múltiples usuarios. Los artistas y científicos pueden trabajar conjuntamente en el mismo proyecto, compartiendo ideas, código y recursos de manera eficiente.

- Simulación y visualización: la creación de modelos dinámicos y la observación de comportamientos emergentes en el ecosistema de abejas virtuales.

- Interactividad: Los usuarios pueden interactuar directamente con las simulaciones, ajustando parámetros y observando los efectos en tiempo real. Esta interactividad es crucial para la experimentación y ajuste fino de los comportamientos de las abejas virtuales.

- Integración de datos: La plataforma permite la integración de datos externos, como información meteorológica real, que puede influir en el comportamiento del ecosistema digital, proporcionando una capa adicional de realismo y complejidad.

Marco metodológico

Estudio morfológico y abstracción geométrica de las abejas

Para realizar una abstracción geométrica de las abejas en JavaScript, se utilizó un estudio morfológico básico de las abejas para identificar y simplificar las características esenciales que definen su forma y apariencia. Aquí está el análisis de cómo se realizó este proceso en el código proporcionado:

Identificación de partes clave de la abeja

Las abejas tienen varias partes clave que son representativas de su morfología:

- *Cabeza*: La parte frontal donde se encuentran los ojos y las antenas.
- *Tórax*: La sección media a la que están unidas las alas y las patas.
- *Abdomen*: La sección trasera del cuerpo.
- Alas: Dos pares de alas que se extienden desde el tórax.
- *Patas*: Seis patas que también están unidas al tórax.

Simplificación geométrica

El siguiente paso es simplificar estas partes en formas geométricas básicas que pueden ser fácilmente dibujadas usando los métodos de dibujo de JavaScript (en este caso, la biblioteca p5.js):

- Cabeza y abdomen: Se representan como elipses.
- Tórax: Se representa también como una elipse, pero con dimensiones que sugieren su conexión entre la cabeza y el abdomen.
- Alas: Se representan como líneas para simplificar.
- Patas: Se representan como líneas.

Abstracción geométrica en el código

En el código, se utilizaron funciones para crear los diferentes tipos de abejas (reina, obrera y zángano) con estos principios de simplificación:

Abeja reina

```
// Definición de la clase AbejaReina
function AbejaReina(tempX, tempY) {
  // Propiedades de la abeja reina
  this.x = tempX; // Posición X
  this.y = tempY; // Posición Y

  // Método para mostrar la abeja reina
  this.display = function() {
    stroke(204, 102, 0); // Color del trazo (naranja)

    // Dibujar la cabeza de la abeja
    ellipse(this.x, this.y, a, b); // Cabeza (elipse)

    // Dibujar el cuerpo de la abeja
    ellipse(this.x, this.y + 10, a, b); // Cuerpo (elipse)

    // Dibujar las alas de la abeja
    line(this.x - 7, this.y - 3, this.x + 7, this.y - 3); // Alas (línea horizontal)
    line(this.x - 7, this.y, this.x + 7, this.y); // Alas (línea horizontal)

    // Dibujar el aguijón de la abeja
    line(this.x, this.y + 17, this.x, this.y + 21); // Aguijón (línea vertical)

    // Dibujar los ojos de la abeja
    point(this.x - 6, this.y - 9); // Ojo izquierdo (punto)
    point(this.x + 6, this.y - 9); // Ojo derecho (punto)
  };
}
```

Imagen 2. Abstracción geométrica en JavaScript (P5.js) de la morfología de la abeja reina. Implementación del código en JavaScript utilizando P5.js para abstraer geométricamente la morfología de la abeja reina en el proyecto Zumbido Digital. Elaboración propia.

Imagen 3. Simulación visual de la abstracción geométrica de la abeja reina generada por el código en Javascript. Elaboración propia.

```
function AbejaReina(tempX, tempY) { ... }:
```

Esto define el constructor de la clase AbejaReina, que acepta dos parámetros tempX y tempY para establecer las coordenadas iniciales (x e y) de la abeja.

```
this.x = tempX; y this.y = tempY;
```

Aquí se inicializan las propiedades x e y de la abeja con los valores proporcionados al crear una nueva instancia de AbejaReina.

```
this.display = function() { ... }:
```

Este es un método de la clase AbejaReina que se encarga de dibujar la representación gráfica de la abeja reina en el canvas.

```
stroke(204, 102, 0);
```

Establece el color del trazo de las formas que se dibujarán a continuación (en este caso, un naranja oscuro).

```
ellipse(this.x, this.y, a, b);
```

Dibuja la cabeza de la abeja como una elipse en la posición (this.x, this.y) con anchura a y altura b.

```
ellipse(this.x, this.y + 10, a, b);
```

Dibuja el cuerpo de la abeja como otra elipse, desplazada verticalmente para estar debajo de la cabeza.

```
line(this.x - 7, this.y - 3, this.x + 7, this.y - 3);
    line(this.x - 7, this.y, this.x + 7, this.y);
```

Dibuja las alas de la abeja como líneas horizontales.

```
line(this.x, this.y + 17, this.x, this.y + 21);
```

Dibuja el aguijón de la abeja como una línea vertical.

```
point(this.x - 6, this.y - 9); y point(this.x + 6, this.y - 9);
```

Dibuja los ojos de la abeja como puntos, uno a la izquierda y otro a la derecha de la cabeza.

Abeja obrera

```
// Definición de la clase AbejaObrera
function AbejaObrera(tempX, tempY) {
  // Propiedades de la abeja obrera
  this.x = tempX; // Posición X
  this.y = tempY; // Posición Y

  // Método para mostrar la abeja obrera
  this.display = function() {
    stroke(107, 80, 72); // Color del trazo (marrón)
    fill(253, 208, 137); // Color de relleno (crema)

    // Dibujar el cuerpo de la abeja
    ellipse(this.x, this.y, a, b); // Cuerpo (elipse)

    // Dibujar las alas de la abeja
    line(this.x - 7, this.y - 3, this.x + 7, this.y - 3); // Alas (línea horizontal)
    line(this.x - 7, this.y, this.x + 7, this.y); // Alas (línea horizontal)

    // Dibujar el aguijón de la abeja
    line(this.x, this.y + 7, this.x, this.y + 11); // Aguijón (línea vertical)

    // Dibujar los ojos de la abeja
    point(this.x - 6, this.y - 9); // Ojo izquierdo (punto)
    point(this.x + 6, this.y - 9); // Ojo derecho (punto)
  };
}
```

Imagen 4. Abstracción geométrica en JavaScript (P5.js) de la morfología de la abeja obrera. Implementación del código en JavaScript utilizando P5.js para abstraer geométricamente la morfología de la abeja obrera en el proyecto Zumbido Digital. Elaboración propia.

Imagen 5. Simulación visual de la abstracción geométrica de la abeja obrera generada por el código en JavaScript. Elaboración propia.

La clase AbejaObrera se define como un constructor que acepta dos parámetros tempX y tempY, los cuales representan las coordenadas iniciales (x e y) donde se ubicará la abeja obrera en el lienzo (canvas).

```
function AbejaObrera(tempX, tempY) {
          this.x = tempX;
          this.y = tempY;
```

Al igual que en la clase AbejaReina, se inicializan las propiedades x e y de la abeja obrera con los valores proporcionados al crear una nueva instancia de AbejaObrera.

```
this.display = function() {
```

El método display es parte de la clase AbejaObrera y se encarga de dibujar la representación gráfica de la abeja obrera en el lienzo. A continuación, se describen los elementos gráficos que conforman esta representación:

`stroke(107, 80, 72);` // Establece el color del trazo para las formas que se dibujarán a continuación (marrón oscuro)

`fill(253, 208, 137);` // Establece el color de relleno para las formas que se dibujarán a continuación (tono crema)

En contraste con la AbejaReina, la abeja obrera utiliza un color de trazo más oscuro (107, 80, 72) y un color de relleno más claro (253, 208, 137), lo que refleja las diferencias morfológicas entre las dos castas de abejas.

`ellipse(this.x, this.y, a, b);` // Dibuja el cuerpo de la abeja obrera como una elipse en la posición (this.x, this.y) con anchura a y altura b

La elipse representa el cuerpo de la abeja obrera, posicionada en las coordenadas (this.x, this.y). Las variables a y b representan las dimensiones de la elipse, las cuales pueden ser ajustadas según las necesidades del diseño.

```
line(this.x - 7, this.y - 3, this.x + 7, this.y - 3);
```

// Dibuja las alas de la abeja obrera como líneas horizontales

```
line(this.x - 7, this.y, this.x + 7, this.y);
line(this.x, this.y + 7, this.x, this.y + 11);
```

// Dibuja el aguijón de la abeja obrera como una línea vertical

```
point(this.x - 6, this.y - 9);
```

// Dibuja el ojo izquierdo de la abeja obrera como punto

```
point(this.x + 6, this.y - 9);
```

// Dibuja el ojo derecho de la abeja obrera como punto

Estas líneas y puntos representan características específicas de la abeja obrera:

Las alas se dibujan como dos líneas horizontales ubicadas encima del cuerpo de la abeja.
El aguijón se representa como una línea vertical debajo del cuerpo.
Los ojos se dibujan como puntos, uno a la izquierda (this.x - 6, this.y - 9) y otro a la derecha (this.x + 6, this.y - 9) del cuerpo de la abeja obrera.

En conjunto, esta clase AbejaObrera en JavaScript con P5.js proporciona una representación gráfica simplificada pero distintiva de la morfología y características visuales de una abeja obrera. Comparada con la AbejaReina, ajusta los colores, formas y detalles para reflejar con precisión las diferencias entre estas dos castas dentro de una colonia de abejas.

Zángano

```
function AbejaZangano(tempX, tempY, tempSpeed) {
  this.x = tempX; // Establece la coordenada x inicial del zángano
  this.y = tempY; // Establece la coordenada y inicial del zángano

  this.display = function () {
    fill(237, 244, 255); // Establece el color de relleno para el cuerpo del zángano (blanco azulado)
    stroke(107, 80, 72); // Establece el color del trazo para las formas que se dibujarán a continuación (marrón oscuro)

    // Dibuja el cuerpo del zángano como una elipse en la posición (this.x, this.y) con anchura 7 y altura 15
    ellipse(this.x, this.y, 7, 15);

    // Dibuja las alas del zángano como líneas horizontales encima del cuerpo
    line(this.x - 7, this.y - 3, this.x + 7, this.y - 3);
    line(this.x - 7, this.y, this.x + 7, this.y);

    // Dibuja los ojos del zángano como puntos, uno a la izquierda y otro a la derecha del cuerpo
    point(this.x - 6, this.y - 9);
    point(this.x + 6, this.y - 9);
  };
}
```

Imagen 6. Abstracción geométrica en JavaScript (P5.js) de la morfología de la abeja zangano. Implementación del código en JavaScript utilizando P5.js para abstraer geométricamente la morfología de la abeja obrera en el proyecto Zumbido Digital. Elaboración propia.

Imagen 7. Simulación visual de la abstracción geométrica de la abeja zángano generada por el código en JavaScript. Elaboración propia.

Componentes específicos:

- Abeja zángano: Representa las alas como líneas horizontales y los ojos como puntos a los lados del cuerpo.
- Abeja reina y abeja obrera: Pueden tener características adicionales específicas, como estructuras de antenas, patrones de coloración o elementos adaptados a sus funciones dentro de la colmena.

Comparación general:

- Función principal: Las tres clases (`AbejaReina`, `AbejaObrera`, `AbejaZangano`) tienen un método `display()` que dibuja la representación gráfica de la abeja correspondiente en el lienzo.
- Parámetros de inicialización: Todas aceptan coordenadas `tempX` y `tempY` para establecer la posición inicial de la abeja en el lienzo.
- Adaptación a oles: Cada clase puede tener adaptaciones específicas en su código para representar visualmente las características únicas de cada tipo de abeja (reina, obrera, zángano), reflejando diferencias en tamaño, color, estructuras corporales y comportamientos (diferente locomoción).

Movimiento y desplazamiento

Además de la representación visual, se añadió movimiento y desplazamiento a las abejas para simular su comportamiento natural:

Movimiento aleatorio: simula el vuelo errático de las abejas.

```
// Método para simular el movimiento aleatorio de las abejas
this.move = function () {
  // Incrementar la posición X de la abeja de manera aleatoria dentro del rango [-this.speed, this.speed]
  this.x += random(-this.speed, this.speed);

  // Incrementar la posición Y de la abeja de manera aleatoria dentro del rango [-this.speed, this.speed]
  this.y += random(-this.speed, this.speed);
};
```

Imagen 8. Código en Javascript (P5.Js) que permite la simulación del vuelo errático de las abejas. Elaboración propia

Explicación:

`this.move = function () { ... };`

Define un método move en el contexto de un objeto o clase (la estructura completa no se proporciona aquí). Este método se llama move y se asume que pertenece a una abeja específica (como una abeja reina o una abeja obrera).

`this.x += random(-this.speed, this.speed);`

Aquí, this.x representa la posición actual en el eje X de la abeja. El operador += se utiliza para incrementar this.x con un valor aleatorio generado por la función random. random(-this.speed, this.speed) devuelve un número aleatorio dentro del rango

-this.speed a this.speed. Esto provoca que la abeja se mueva de manera aleatoria hacia la izquierda o la derecha en el canvas.

this.y += random(-this.speed, this.speed);
Similar al punto anterior, this.y representa la posición actual en el eje Y de la abeja. También se incrementa this.y con un valor aleatorio generado por random(-this.speed, this.speed), lo cual hace que la abeja se mueva de manera aleatoria hacia arriba o hacia abajo en el canvas.

Uso de random

`random(-this.speed, this.speed)`
genera un número aleatorio en cada llamada dentro del rango especificado. La función random en p5.js devuelve un número decimal aleatorio entre el primer y el segundo argumento.
this.speed

`this.speed`
es un parámetro que controla la velocidad máxima a la que puede moverse la abeja en cada dirección. Si this.speed es pequeño, el movimiento será más suave y controlado; si es grande, el movimiento será más errático y rápido.

Finalidad:

- Este método move se utiliza para simular el vuelo errático de las abejas en un entorno digital. El movimiento aleatorio es característico del comportamiento natural de las abejas mientras buscan flores y polinizan, lo cual aporta realismo y dinamismo a las representaciones gráficas de abejas en aplicaciones o simulaciones visuales.

- Desplazamiento controlado: Define rutas de desplazamiento para simular patrones de vuelo más dirigidos.

Desplazamiento de un punto a otro

```
this.displacement = function () {
  // Verificar si el desplazamiento ha alcanzado su punto final
  if (this.pct < 1.0) {
    // Calcular la posición actual en función del porcentaje completado
    this.displacementX = this.startX + (this.stopX - this.startX) * this.pct;
    this.displacementY = this.startY + (this.stopY - this.startY) * this.pct;

    // Incrementar el porcentaje completado
    this.pct += this.step;

    // Verificar si se completó el desplazamiento
    if (this.pct > 1.0) {
      // Actualizar los puntos de inicio y fin para el próximo desplazamiento
      this.startX = this.stopX;
      this.startY = this.stopY;

      // Generar nuevos puntos finales aleatorios para el próximo desplazamiento
      this.stopX = random(10, 350);
      this.stopY = random(10, 350);

      // Reiniciar el porcentaje completado y generar un nuevo paso aleatorio
      this.pct = 0.0;
      this.step = random(0.0, 0.1);
    }
  }
};
```

Imagen 9. Código en Javascript que genera un movimiento o desplazamiento de punto A a punto B. Elaboración propia.

Explicación:

- Verificación del progreso del desplazamiento:

```
if (this.pct < 1.0) { ... }:
```

Esta condición verifica si el porcentaje completado (pct) es menor que 1.0, lo cual indica que el desplazamiento aún no ha alcanzado su destino final.

- Cálculo de la posición actual:

```
this.displacementX = this.startX + (this.stopX - this.startX) * this.pct; y this.displacementY = this.startY + (this.stopY - this.startY) * this.pct;:
```

Estas líneas calculan las coordenadas actuales (displacementX y displacementY) basadas en el porcentaje completado (pct). Utilizan una interpolación lineal entre los puntos de inicio (startX, startY) y los puntos finales (stopX, stopY) para determinar la posición actual durante el desplazamiento.

- Avance en el desplazamiento:

```
this.pct += this.step;:
```

Incrementa el porcentaje completado (pct) por el valor de step. Esto avanza gradualmente el desplazamiento desde el punto de inicio hacia el punto final.

- Finalización del movimiento:

```
if (this.pct > 1.0) { ... }:
```

Esta condición se ejecuta cuando el desplazamiento ha alcanzado su destino final (cuando pct es mayor que 1.0). En este caso:

- Los puntos de inicio (startX, startY) se actualizan a los últimos puntos finales alcanzados (stopX, stopY).
- Se generan nuevos puntos finales aleatorios (stopX, stopY) para el próximo movimiento.
- pct se reinicia a 0.0 para iniciar un nuevo desplazamiento desde los nuevos puntos de inicio.
- step se establece en un nuevo valor aleatorio (random(0.0, 0.1)), controlando la velocidad o la rapidez del próximo movimiento.

Este método ofrece una manera efectiva de simular el movimiento dinámico y natural de las abejas en un entorno virtual. Inspirado en conceptos como los planteados por Xenakis en *Formalized Music* 1992, el proyecto se basa en modelado de patrones y adaptaciones de programación para crear un comportamiento dinámico similar al observado en las abejas reales dentro de sus colonias.

Implementación en Nota.Space

A continuación, se explicará la implementación del código en la plataforma Nota.Space:

```
audioContext = modules.audio.getContext();
```

```
//////// Código de inicialización - solo al cargar el espacio o cerrar el editor de scripts
if (!fragment.tmpMem.init) {
    // Configuración inicial del fragmento
    fragment.text = '';
    fragment.tmpMem.amp = 0.5;

    // Tipo de onda para el oscilador
    let wave = 'sine';

    // Inicialización de variables de estado
    fragment.tmpMem.init = true;
    fragment.tmpMem.random1 = Math.random() * 77777777777;
    fragment.tmpMem.random2 = Math.random() * 77777777777 * 10;

    // Creación de nodos para la generación de sonido
    const gainNode = audioContext.createGain();
    gainNode.gain.value = 0.0;
    fragment.tmpMem.freq = 200 + Math.random() * 150;
    const oscillator = new OscillatorNode(audioContext, {
        frequency: fragment.tmpMem.freq,
        type: wave
    });
    const pan = new StereoPannerNode(audioContext, { pan: 0 });

    // Conectar el oscilador con el nodo de ganancia y el panoramizador al destino de audio
    oscillator.connect(gainNode).connect(pan).connect(audioContext.destination);
    oscillator.start(0);

    // Almacenar los nodos creados en la memoria temporal del fragmento
    fragment.tmpMem.gain = gainNode;
    fragment.tmpMem.osc = oscillator;
    fragment.tmpMem.pan = pan;
}
```

Imagen 10. Código completo de una abeja en la plataforma Nota.Space. Elaboración propia.

audioContext: Aquí se obtiene el contexto de audio (audioContext) desde el objeto modules.audio. Este contexto de audio es esencial para trabajar con sonido en JavaScript, permitiendo la creación de nodos de audio y la gestión del flujo de audio.

Inicialización: Esta sección del código se ejecuta una sola vez cuando fragment.tmpMem.init es false (o no está definido). Esto asegura que las configuraciones iniciales del fragmento se realicen solo una vez.

```
fragment.text = '';
```

Establece el texto del fragmento como ''.

```
fragment.tmpMem.amp = 0.5;
```

Inicializa la amplitud del sonido en 0.5.

```
let wave = 'sine';
```

Define el tipo de onda del oscilador como 'sine', que es una onda sinusoidal.

```
fragment.tmpMem.init = true;
```

Marca la inicialización como completada para evitar ejecuciones repetidas.

`fragment.tmpMem.random1 = Math.random();` y `fragment.tmpMem.random2 = Math.random() * 10;`
Genera dos números aleatorios y los guarda para usar en la caminata aleatoria y otras variaciones.

```
const gainNode = audioContext.createGain();
```
Crea un nodo de ganancia (gainNode) en el contexto de audio.

```
gainNode.gain.value = 0.0;
```
Inicializa el valor de ganancia del nodo a 0.0, lo que significa que inicialmente no se escuchará ningún sonido.

```
fragment.tmpMem.freq = 2000 + Math.random() * 150;
```
Define la frecuencia inicial del oscilador como un valor entre 2000 y 2150 aproximadamente.

```
const oscillator = new OscillatorNode(audioContext, {
  frequency: fragment.tmpMem.freq, type: wave });
```
Crea un nodo de oscilador (oscillator) en el contexto de audio, configurando su frecuencia y tipo de onda.

```
const pan = new StereoPannerNode(audioContext, { pan: 0 });
```
Crea un nodo de panoramización estéreo (pan) en el contexto de audio, inicialmente centrado.

```
oscillator.connect(gainNode).connect(pan).connect(audioContext.destination);
```
Conecta el oscilador primero al nodo de ganancia (gainNode), luego al nodo de panoramización (pan), y finalmente al destino de audio del contexto (audioContext.destination), que generalmente es la salida de audio del sistema.

```
oscillator.start(0);
```
Inicia la generación de sonido desde el oscilador en el tiempo 0 del contexto de audio.

```
38  ////////////////////////////////////// En cada fotograma
39
40  // Movimiento aleatorio
41  this.x_drift = sketch.noise(new Date().getTime() / 5000 + this.tmpMem.random1) * 30;
42  this.y_drift = sketch.noise(new Date().getTime() / 5000 + this.tmpMem.random2) * 30;
```

Imagen11. Fragmento del código en Nota.space que ilustra el movimiento aleatorio o caminata aleatoria. Elaboración propia.

Movimiento de las abejas

En el código proporcionado, la parte donde se aplica la caminata aleatoria para el movimiento del fragmento está en la sección que calcula this.x_drift y this.y_drift en cada fotograma:

```
sketch.noise():
```
La función noise() genera valores de ruido pseudoaleatorios. Toma como entrada una sola dimensión y devuelve valores entre 0 y 1 de manera suave y continua.

```
new Date().getTime() / 70000:
```
Aquí se usa el tiempo actual dividido por 70000 para obtener una variación lenta del tiempo. Esta técnica se utiliza para generar variaciones de ruido que cambian gradualmente con el tiempo.

this.tmpMem.random1 y this.tmpMem.random2:

Son números aleatorios precalculados que se suman a la entrada de noise(). Esto introduce variabilidad adicional en el patrón de ruido, asegurando que el movimiento no sea completamente predecible.

* 30:

Escala los valores generados por noise() para controlar la magnitud del desplazamiento. En este caso, multiplica por 30, lo que determina la amplitud máxima del movimiento en las direcciones x (this.x_drift) y y (this.y_drift).

En conjunto, esta sección del código asegura que el fragmento experimente un movimiento aleatorio y continuo en el espacio de Nota.Space, creando una sensación de dinamismo y vida en la representación visual y sonora del fragmento en el contexto del proyecto.

Modificación de la frecuencia del oscilador:

const f = fragment.tmpMem.freq + Math.random() * 10;:

```
// SONIDO
// FRECUENCIA
const f = fragment.tmpMem.freq + Math.random() * 15;
fragment.tmpMem.osc.frequency.exponentialRampToValueAtTime(
    f,
    audioContext.currentTime + 0.5 + Math.random() * 0.3
);

// AMPLITUD
if (sketch.frameCount % 10 === 0) {
    const mid_a = 0.5;
    fragment.tmpMem.amp = (mid_a + Math.random() * mid_a / 10);
}

// Ajustar la amplitud del sonido de manera lineal
fragment.tmpMem.gain.gain.linearRampToValueAtTime(
    fragment.tmpMem.amp * fragment.getRelativeOnScreenArea(),
    audioContext.currentTime + 0.1
);
```

Imagen 12. Fragmento del código en Nota.Space que ilustra la generación de sonido y sus características para una abeja. . Elaboración propia.

Aquí se genera un nuevo valor de frecuencia f sumando un valor aleatorio entre 0 y 10 a la frecuencia base fragment.tmpMem.freq. Esto introduce variabilidad en la frecuencia del oscilador, permitiendo

```
64  // Panning (posicionamiento espacial)
65  const panner = fragment.tmpMem.pan;
66  // Calcular el valor de posición del fragmento de -1 a 1
67  let fragMid = fragment.screenX() + fragment.screenW() / 2;
68  let pos = fragMid / modules.main.getSketch().width;
69  pos = -1 + 2 * pos;
70  pos = Math.min(1, Math.max(-1, pos));
71  // Aplicar el panoramizado con rampa lineal dependiente del tiempo
72  panner.pan.linearRampToValueAtTime(
73      Math.max(0, Math.min(1, pos)),
74      audioContext.currentTime + 0.2
75  );
```

Imagen 13. Fragmento del código en Nota.Space que permite controlar la posición del sonido para una abeja. Elaboración propia.

cambios sutiles y aleatorios en el tono del sonido generado.

`fragment.tmpMem.osc.frequency.exponentialRampToValueAtTime():`

Esta línea indica al oscilador (fragment.tmpMem.osc) que cambie su frecuencia de manera exponencial hasta alcanzar el valor f en un tiempo determinado. La función exponentialRampToValueAtTime() es útil para transiciones suaves en la frecuencia, creando cambios de tono gradual en el sonido.

`audioContext.currentTime + 0.5 + Math.random() * 0.3:`

Define el momento exacto en el tiempo en el que la frecuencia del oscilador debe alcanzar el valor f. audioContext.currentTime representa el tiempo actual del contexto de audio. Sumando 0.5 y añadiendo un pequeño valor aleatorio (Math.random() * 0.3), se introduce una pequeña variación en el tiempo de la transición de la frecuencia, lo que contribuye a la sensación de dinamismo y naturalidad en el cambio de tono.

Creación del nodo de panorámico (panner):

`const panner = fragment.tmpMem.pan;`

Aquí se obtiene el nodo de panorámico (pan) almacenado en la memoria temporal del fragmento (fragment.tmpMem.pan). Este nodo se encarga de controlar la posición del sonido en el espacio estéreo.

Cálculo de la posición en el espacio estéreo:

```
let fragMid = fragment.screenX() + fragment.screenW() / 2;
    let pos = fragMid / modules.main.getSketch().width;
```

- fragMid calcula la posición horizontal media del fragmento en la pantalla sumando su posición fragment.screenX() con la mitad de su ancho fragment.screenW() / 2.
- pos calcula la posición relativa del fragmento en la pantalla dividiendo fragMid por el ancho total del sketch (modules.main.getSketch().width). Esto da como resultado un valor entre 0 y 1 que representa la posición relativa del fragmento en la pantalla.

Normalización del rango de posición:

```
pos = -1 + 2 * pos;
pos = Math.min(1, Math.max(-1, pos));
```

- pos = -1 + 2 * pos;: Transforma el rango de pos de [0, 1] a [-1, 1]. Ahora pos representa la posición panorámica del sonido, donde -1 es totalmente a la izquierda, 0 es el centro y 1 es totalmente a la derecha.
- Math.min(1, Math.max(-1, pos));: Asegura que pos esté dentro del rango permitido [-1, 1]. Esto es importante para evitar valores fuera del rango que podrían causar distorsiones en el panoramizado.

Aplicación del panorámico con rampa lineal dependiente del tiempo:

```
panner.pan.linearRampToValueAtTime(Math.max(0,Math.min(1,pos)), audioContext.current-
Time + 0.2);
```

- panner.pan.linearRampToValueAtTime(): Este método del nodo de panorámico (panner.pan) aplica una transición lineal de la posición panorámica del sonido (pan) a Math.max(0, Math.min(1, pos)) en un tiempo específico.
- Math.max(0, Math.min(1, pos)): Asegura que pos esté dentro del rango [0, 1], ya que el valor del panoramizado debe estar entre 0 (totalmente a la izquierda) y 1 (totalmente a la derecha).
- audioContext.currentTime + 0.2: Especifica el momento en el tiempo (en segundos) en el cual la transición lineal debe completarse. Esto asegura que los cambios en el panoramizado sean suaves y controlados.

En resumen, esta sección del código calcula y aplica un efecto de panoramizado que posiciona el sonido generado por el oscilador en el espacio estéreo, basado en la posición relativa del fragmento en la pantalla. Esto contribuye a una experiencia auditiva envolvente y dinámica en la plataforma Nota.Space.

Resultados

Se presentan capturas de pantalla de la colonia digital de abejas en la plataforma Nota.Nota.Space. Para explorar la simulación completa de la colonia de abejas, visite el siguiente enlace: «https://nota.space/?user=eddiefloat&room=bees» o escanee el siguiente código QR.

Imagen 14. Código QR para acceder a la obra en la plataforma Nota.Space. Elaboración propia.

Para navegar por el entorno y escuchar el sonido de las abejas, utilice el scroll del mouse para desplazarse.

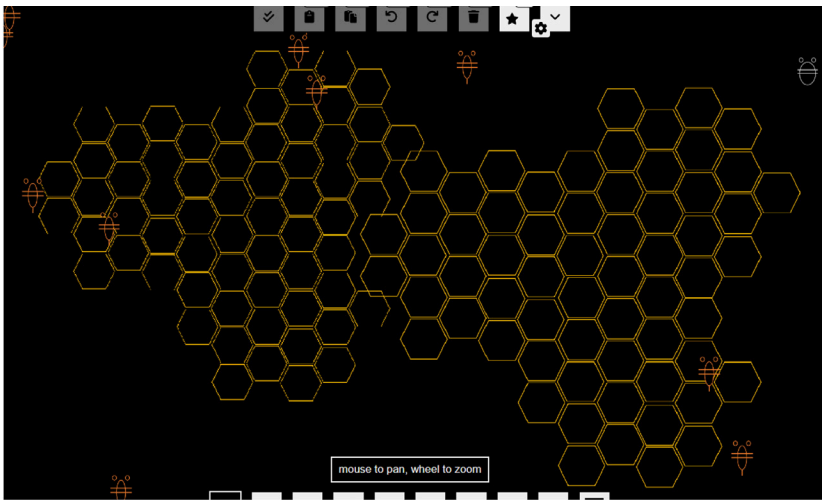


Imagen 15. Ilustración de una parte de la colmena digital en la plataforma Nota.Space. Las abejas en amarillo son abejas reina y las de color blanco son abejas zangano. Elaboración propia.

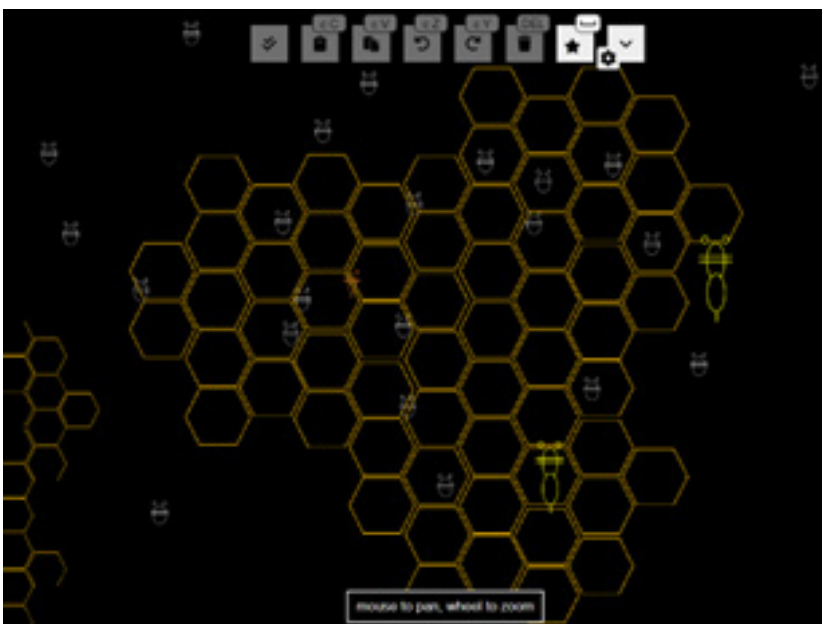


Imagen16. Ilustración de una parte de la colmena digital en la plataforma Nota.Space. Las abejas de color marrón son abejas obreras.laboración propia.

Conclusiones

El proyecto *Zumbido Digital: Explorando el* bio *arte con un ecosistema de abejas irtuales* demuestra cómo la combinación de arte, entomología y tecnología puede crear experiencias inmersivas y educativas que no sólo reflejan la vida real, sino que también exploran conceptos abstractos sobre la vida y la evolución en entornos digitales. A través del uso de algoritmos de caminata aleatoria y técnicas de programación orientada a objetos, se ha logrado simular de manera efectiva los comportamientos de diferentes roles dentro de una colonia de abejas.

Interactividad y emergencia en el ecosistema de abejas: Implicaciones bioéticas

La simulación del ecosistema de abejas permite observar cómo comportamientos complejos pueden surgir de interacciones simples entre agentes autónomos. Este aspecto no sólo resalta la capacidad del bio arte para explorar y representar fenómenos biológicos de manera interactiva y accesible, sino que también plantea importantes consideraciones bioéticas. Desde una perspectiva ética, la capacidad de simular y manipular entornos biológicos digitales plantea preguntas sobre el manejo responsable de la tecnología en la replicación y alteración de la naturaleza.

Las implicaciones bioéticas de estas simulaciones incluyen la necesidad de evaluar cómo estas tecnologías podrían influir en el comportamiento y la salud de las especies reales, así como en la conservación y la biodiversidad. Además, la representación digital de la naturaleza plantea cuestiones sobre la ética de la intervención humana en ecosistemas naturales y la responsabilidad en la gestión de estas representaciones.

En conclusión, mientras que la interactividad y la emergencia en la simulación del ecosistema de abejas son fascinantes desde un punto de vista artístico y científico, también demandan una reflexión crítica sobre sus implicaciones éticas y bioéticas en relación con el manejo y la representación de la vida natural y artificial.

Innovación en el arte digital y horizontes futuros

Al integrar principios científicos con la expresión artística, *Zumbido Digital* no sólo se posiciona como una obra visualmente atractiva, sino también como un proyecto conceptualmente profundo. Este enfoque interdisciplinario no marca un antes y un después absoluto en la relación entre arte, ciencia y tecnología, pero sí abre nuevos horizontes significativos para futuros proyectos de arte digital.

Desde esta perspectiva, el arte siempre ha estado influenciado por la ciencia y la tecnología de su tiempo. Sin embargo, la innovación en el arte digital contemporáneo radica en su capacidad para explorar y traducir los avances científicos y tecnológicos más recientes en nuevas formas de expresión artística. Esto no sólo amplía el espectro estético, sino que también permite una reflexión profunda sobre las intersecciones entre la cultura humana y el progreso tecnológico.

Mirando hacia el futuro, la integración de la ciencia y la tecnología en el arte digital podría llevar a la creación de obras que no sólo sean estéticamente innovadoras, sino también críticamente reflexivas sobre temas como la inteligencia artificial, la biotecnología, y la sostenibilidad ambiental. Estos horizontes emergentes no sólo prometen expandir el campo del arte digital, sino también generar discusiones más profundas y multidimensionales sobre el papel del arte en la sociedad contemporánea.

Además, la percepción de profundidad y atractivo en el arte digital puede ser subjetiva y contextual, requiriendo un análisis desde diversos marcos teóricos y culturales para una comprensión completa de su impacto y significado en la actualidad y en el futuro.

Potencial para investigaciones futuras

El prototipo desarrollado puede servir como base para futuras investigaciones en simulación de ecosistemas digitales y sistemas autoorganizados. Las técnicas y algoritmos empleados pueden adaptarse y expandirse para explorar otros fenómenos biológicos y ecológicos.

Reflexión profunda sobre la vida artificial

Zumbido Digital no sólo presenta un ecosistema digital que simula la vida real de las abejas, sino que también invita a una reflexión profunda sobre la naturaleza de la vida y la existencia, especialmente en el contexto de la vida artificial. Este proyecto desafía nuestras percepciones al imitar mecánicamente comportamientos biológicos complejos, pero también abre un espacio para explorar filosóficamente qué significa estar vivo en un entorno digital.

La concepción de la vida en *Zumbido Digital* puede parecer ambigua a lo largo del documento, reflejando una dualidad entre el impulso mecánico y el continuum naturaleza-cultura. Sin embargo, más allá de simplemente imitar comportamientos biológicos, la obra-investigación-dispositivo podría profundizar al considerar observaciones empíricas sobre el comportamiento real de las abejas. Esta perspectiva podría enriquecer la interpretación de la vida artificial no sólo como una simulación técnica, sino como un estudio continuo de la interacción compleja entre lo biológico y lo artificial.

Para darle mayor profundidad a esta exploración, sería relevante integrar análisis críticos que aborden cómo *Zumbido Digital* no solamente replica mecánicamente la vida de las abejas, sino que también plantea preguntas sobre la autonomía, la adaptabilidad y la evolución en los sistemas biológicos simulados. Además, podríamos explorar cómo estas simulaciones podrían influir en nuestra comprensión de la vida misma, desafiando las fronteras tradicionales entre lo natural y lo creado por el ser humano.

Finalmente, al considerar estas observaciones y enriquecer la investigación con un enfoque interdisciplinario que incluya tanto la ciencia como la filosofía, Zumbido Digital puede ofrecer una plataforma única para explorar y debatir las complejidades éticas, científicas y culturales de la vida artificial en el arte contemporáneo.

En conclusión, Zumbido Digital no sólo es una obra de arte innovadora, sino también una plataforma de experimentación para el estudio de la vida artificial y la interacción digital. La combinación de tecnología, ciencia y arte en este proyecto demuestra el potencial del bio arte para crear experiencias significativas y transformadoras.

Agradecimientos

Agradezco sinceramente a la plataforma Nota. Space por su invaluable apoyo durante la realización de este proyecto, así como al International Digitalkunst Festival por la oportunidad de presentar mi trabajo. También expreso mi gratitud a las organizaciones NOTA e.V. y Stuttgarter Kollektiv für aktuelle Musik e.V. por su reconocimiento y respaldo continuo.

Además, quiero reconocer a la Universidad Federal del Extremo Oriente, donde tomé cursos relacionados con el arte digital durante mi maestría. Este conocimiento ha sido fundamental para el desarrollo de esta investigación.

Por último, agradezco a la Universidad Nacional Autónoma de México, donde actualmente estoy realizando mi programa de doctorado. El soporte académico y los recursos proporcionados por esta institución han sido fundamentales para profundizar en el estudio de la composición algorítmica y su aplicación en este proyecto de investigación.

Referencias

Bartlem, E. (2005). Immersive artificial life (A-life) art. *Journal of Australian Studies, 28*(84), pp. 95-107.

Berg, H. C. (1983). *Random Walks in Biology*. Princeton University Press.

Codling, E. A., Plank, M. J., & Benhamou, S. (2008). Random walk models in biology. *Journal of the Royal Society Interface, 5*(25), pp. 813-834. https://doi.org/10.1098/rsif.2008.0014

Conway, J. (1970). The game of life. *Scientific American, 223*(4), p. 4.

Dorin, A. (2008). A survey of virtual ecosystems in generative electronic art. In The art of artificial evolution: A handbook on evolutionary art and music (pp. 289-309). Berlin, Heidelberg: Springer Berlin Heidelberg.

Ewens, W. J. (2004). *Mathematical Population Genetics*. Springer.

Fama, E. F. (1965). The behavior of stock-market prices. *The Journal of Business, 38*(1), pp. 34-105. https://doi.org/10.1086/294743

Feller, W. (1968). *An Introduction to Probability Theory and Its Applications*, Vol. 1. Wiley.

García Oliver, M. Á. (2012). *Complejidad y Modelado Basado en Agentes*. Una aproximación a los ecosistemas virtuales en el A-Life Art.

Gooding Sánchez, L. F. (2018). Del suspiro en el alba hasta el abrazo en el ocaso: Una composición evolutiva. *Espacio Sonoro, 46*, pp. 1-7.

Gooding Sánchez, L. F. & Borbón, E. J. G. (2018). Simbiosis: Composición para bandola andina colombiana y entidades sónicas. *Estudios artísticos. Revista de investigación creadora 4*(4), pp. 88-109.

Grimaldi, D., & Engel, M. S. (2005). *Evolution of the Insects*. Cambridge University Press.

Grinstead, C. M., & Snell, J. L. (2012). *Introduction to Probability*. American Mathematical Society.

Gullan, P. J., & Cranston, P. S. (2014). *The Insects: An Outline of Entomology*. Wiley-Blackwell.

Hernández, I., & Niño, R. (2010). *Estética, vida artificial y biopolítica. Expansiones en la evolución cultural y biológica a través de la tecnología*.

Izhikevich, E. M., Conway, J. H., & Seth, A. (2015). Game of life. *Scholarpedia, 10*(6), p. 1816.

Kimura, M. (1983). *The Neutral Theory of Molecular Evolution*. Cambridge University Press.

Lawler, G. F., & Limic, V. (2010). Random walk: a modern introduction, Vol. 123. Cambridge University Press.

Lem, S. (2002). *Solaris*. United Kingdom: Harcourt Brace & Company.

Malkiel, B. G. (1973). *A Random Walk Down Wall Street*. W. W. Norton & Company.

McCormack, J. (1993). *Interactive evolution of L-system grammars for computer graphics modelling*. Complex Systems: from biology to computation, 2.

McCormack, J. (2008). Evolutionary L-systems. In *Design by evolution: Advances in evolutionary design* (pp. 169-196). Berlin, Heidelberg: Springer Berlin Heidelberg.

Nota.Space. (n.d.). *About*. Recuperado de «https://nota.space»

Ohlenschlager, K., Weibel, P., & Weidinger, A. (Eds.). (2023). Christa Sommerer & Laurent Mignonneau: *The Artwork as a Living System 1992-2022*. MIT Press.

Padua, D., & Padua, D. (2021). *The digital ecosystem. Digital Cultural Transformation: Building Strategic Mindsets via Digital Sociology*, pp. 89-136.

Penny, S. (2010). Twenty years of artificial life art. *Digital Creativity, 21*(3), pp. 197-204.

Schrödinger, E. (1944). *What is life? The physical aspect of the living cell*.

Service, M. W. (2012). *Medical Entomology for Students*. Cambridge University Press.

Snodgrass, R. E. (1935). *Principles of Insect Morphology*. McGraw-Hill.

Speight, M. R., Hunter, M. D., & Watt, A. D. (2008). *Ecology of Insects: Concepts and Applications. Wiley-Blackwell.*

Triplehorn, C. A., & Johnson, N. F. (2005). *Borror and DeLong's Introduction to the Study of Insects*. Cengage Learning.

Wu, Z. W., Qu, H., & Zhang, K. (2024). A survey of recent practice of Artificial Life in visual art. *Artificial Life, 30*(1), pp. 106-135.

Xenakis, I. (1992). *Formalized Music: Thought and Mathematics in Composition*. Hillsdale, NJ: Pendragon Press.