ALGORITMOS GENÉTICOS APLICADOS AL PLANEAMIENTO DE TRAYECTORIAS DE UN ROBOT MÓVIL

Autores:

OSCAR DARÍO NAVAS GÓMEZ

JOSÉ NIKOLAI ORTIZ ORTEGA

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA

MAYO DE 2006

ALGORITMOS GENÉTICOS APLICADOS AL PLANEAMIENTO DE TRAYECTORIAS DE UN ROBOT MÓVIL

Autores:

OSCAR DARÍO NAVAS GÓMEZ

JOSÉ NIKOLAI ORTIZ ORTEGA

TRABAJO DE GRADO

Director:

Dr. Techn. ROBERTO MARTÍNEZ ÁNGEL

Codirector:

Mag. Ing. DIEGO ALEXANDER TIBADUIZA BURGOS

UNIVERSIDAD INDUSTRIAL DE SANTANDER

FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES

BUCARAMANGA

MAYO DE 2006

TABLA DE CONTENIDO

ÍNDICE DE FIGURAS

Pág.

ÍNDICE DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN

TÍTULO: ALGORITMOS GENETICOS APLICADOS AL PLANEAMIENTO DE TRAYECTORIAS DE UN ROBOT MOVIL*

AUTORES: NAVAS GOMEZ, OSCAR DARIO; ORTIZ ORTEGA, JOSE NIKOLAI**

PALABRAS CLAVES: Planeamiento de Trayectorias, Inteligencia Artificial, Algoritmos Genéticos, Algoritmo Genético Simple, Operación Genética, Elitismo, Extinción, Sobreviviente, Control, Robótica Móvil, Simulación de Trayectorias, Desarrollo de Software.

DESCRIPCIÓN:
En este documento se describe como se desarrolla una herramienta computacional basada en la técnica de inteligencia artificial denominada algoritmos genéticos y codificada en el lenguaje C# de la suite Visual Studio.NET, teniendo en cuenta el paradigma de la programación orientada a objetos. Esta herramienta permite profundizar en el tema del planeamiento de trayectorias en ambientes controlados tanto en modo online como de simulación, esto se demostró adaptándola a un sistema de visión y a un modulo de radiofrecuencia vinculado a un robot móvil pudiendo así probar la técnica desarrollada en una aplicación real. Además del algoritmo genético simple se incorporaron 2 nuevas técnicas basadas en el elitismo denominadas “Extinción” y “Sobreviviente” que constituyen un aporte al crecimiento de la computación evolutiva.
Inicialmente se exponen los conceptos básicos de la inteligencia artificial y del planeamiento de trayectorias. El siguiente capitulo describe la codificación utilizada en el algoritmo genético implementado así como los diagramas de flujo que representan las operaciones genéticas, también se presenta la herramienta desarrollada y sus características de funcionamiento la cual se somete a pruebas de aplicación donde se extraen una serie de resultados que permiten concluir acerca del algoritmo implementado y sus alcances, estas se exponen en el capitulo de conclusiones y recomendaciones.
Se incluyen una serie de anexos que incluyen el manual del usuario final y la descripción del marco de acción de la aplicación a la robótica móvil.

---

* Trabajo de Grado.

** Facultad de Ingenierías Físico-mecánicas. Ingeniería Electrónica. Roberto Martínez Ángel; Diego Alexander Tibaduiza Burgos

ABSTRACT

TÍTLE: GENETIC ALGORITHMS APPLIED TO TRAJECTORIES PLANNING OF A MOBILE ROBOT*

AUTHORS: NAVAS GOMEZ, OSCAR DARIO; ORTIZ ORTEGA, JOSE NIKOLAI**

KEYWORDS: Trajectories planning, Artificial Intelligence, Genetic Algorithms, Simple Genetic Algorithm, Genetic Operation, Elitism, Extinction, Survivor, Control, Mobile Robotics, Trajectories Simulation, Software Development.

DESCRIPTION:
In this document it is described how to develop a computacional tool based on the artificial intelligence technique denominated genetic algorithms and codified on the language C# that belongs to the Visual Studio.NET suite, all of this using the object-oriented programming paradigm. This tool allows to deepen in the subject of the planning of trajectories in controlled environments so much in way online as of simulation, this was shown adapting it to a vision system and to a radio frequency module attached to a mobile robot thus being able to prove the developed technique in a real application. In addition to the simple genetic algorithm two new techniques based on the elitism were developed and denominated “Extinction” and “Survivor” that constitute a contribution to the growth of the evolutionary computation.
Initially the basic concepts of the artificial intelligence and the planning of trajectories are exposed. The following chapter describes the codification used in the genetic algorithm implemented as well as the flow charts that represent the genetic operations, it is also introduced the developed tool and their operation characteristics which is tested for extracting a series of results that allow to conclude about the implemented algorithm and its scopes, this can be found on the conclusions and recommendations chapter.
It is included on the annexed documents the manual of the final user and the description of the action-frame of the mobile robotics application.

---

* Work of Grade.

** Faculty of Engineering Phisical-Mechanics. Electronics Engineering. Roberto Martínez Ángel; Diego Alexander Tibaduiza Burgos.

INTRODUCCIÓN

El planeamiento de trayectorias busca dar solución al problema de trasladar un ente de un lugar a otro, describiendo en su respuesta solo la ruta a seguir, sin tener en cuenta la dinámica requerida para llevar a cabo esta trayectoria. Existen distintas formas de solucionar un problema de planeamiento de trayectorias, entre ellas se cuentan los algoritmos matemáticos, y los métodos de inteligencia artificial. Esta ultima en su búsqueda por emular la inteligencia humana y posteriormente superarla, ha visto en el planeamiento de trayectorias un reto muy interesante.

¿Qué hace llamativo este problema para la inteligencia artificial? El hecho de que el planeamiento de trayectorias es solucionado constantemente por la inteligencia humana. Cada vez que una persona intenta llevar algo de un lugar a otro su mente genera miles de posibles trayectorias a seguir y escoge la mejor, igual a cuando se toma un objeto con la mano o cuando hay que desplazarse en una ciudad.

Problemas de planeamiento de trayectorias más complejos como el enfrentarse a un laberinto o a un entorno variable ponen a prueba los métodos matemáticos convencionales y muestran que un sistema que emule el comportamiento humano representa una opción interesante, más aun si al planeamiento de la trayectoria se le agregan elementos socio-humanos como lo son el cooperativismo.

Entre las distintas técnicas de inteligencia artificial que intentan resolver el problema del planeamiento de trayectorias se encuentra los algoritmos genéticos, una técnica enfocada a la optimización que encuentra máximos globales (soluciones) sin la restricción de poseer una base de datos inicial. Esta característica la hace interesante si se trata de resolver problemas variables donde las soluciones parecen ser siempre únicas e independientes de cualquier otra solución. Basados en esta característica de los algoritmos genéticos, se ha planteado solucionar el problema de un planeamiento de trayectorias en un

entorno controlado, con dos obstáculos fijos y un obstáculo móvil, lo cual representa una tarea suficientemente compleja en la vasta área de aplicación de la inteligencia artificial. El desarrollo de una aplicación en un sistema computacional orientado al usuario final y el análisis de las características que conforman el algoritmo y su incidencia en las respuestas son entonces los objetivos primordiales de este proyecto.

1. CONCEPTOS BÁSICOS

En este capitulo se introducen los conceptos fundamentales que se deben conocer para entender los alcances del proyecto. Se abordan en el, tópicos tales como el planeamiento de trayectorias, la inteligencia artificial y su rama denominada algoritmos genéticos.

1.1 PLANEAMIENTO DE TRAYECTORIAS

El planeamiento de trayectorias consiste en definir la forma adecuada de llevar de un punto a otro un objeto. Esto aplicado en la robótica móvil presenta diversas variables, las cuales debemos cuantificar para poder escoger una ruta adecuada. El planeamiento de trayectorias no se ocupa de ejecutarlas, solo de presentar la mejor ruta posible a seguir. Teniendo en cuenta esto, las condiciones mínimas que debe cumplir una trayectoria son:

- Que logre llegar al punto marcado como final, pues es este el objetivo de una trayectoria.
- Realizar el desplazamiento en el menor tiempo posible, esto en cuanto a que el factor tiempo se relaciona directamente con los diversos procesos que dependan de la acción del robot que cubre la trayectoria.

- Mantener el buen estado del robot. Una trayectoria que genere daños al robot no es viable, ya sean estos daños debidos a que el grado de dificultad que posee la misma implique daños mecánicos en el robot, o por que esta lo traslada a través de zonas no aptas para la circulación del robot.

Pensando en estos puntos a la hora de planear trayectorias, es indispensable cuantificar estos problemas y sopesar las posibles rutas a seguir. Los diversos métodos de solución incluyen el suavizado de los caminos utilizando

interpolaciones de distintos órdenes o el uso de inteligencia artificial para la obtención de la ruta óptima.

1.2 INTELIGENCIA ARTIFICIAL

La inteligencia artificial consiste de un compendio de técnicas y modelamientos, que pretenden simular un proceso autónomo. La idea fundamental plantea que un sistema pueda generar ("pensar") una solución a un problema en particular sin la necesidad de una respuesta pre-establecida por un operador humano. El termino "Inteligencia Artificial" fue aceptado como tal por primera vez en 1956 en el marco de la Conferencia de Dartmouth. En este evento John McCarthy ubicó a la inteligencia artificial como un campo individual aparte de las ciencias de la computación; sin embargo existen otros hitos históricos anteriores de esta área como lo son los aportes de Warren McCulloch y Walter Pitts en 1943 quienes desarrollaron las teorías de las redes neuronales artificiales y el trabajo de Alan M Turing en 1950 quien con su test de turing (El test consiste en que no se pudiese diferenciar un humano de un ordenador en una charla normal) definió inicialmente como podría un sistema tener o no inteligencia artificial.

En la década de los 60 el principal avance de la inteligencia artificial se dio en el campo de los llamados "Sistemas Expertos". Estos, basados en máquinas secuénciales, encontraron como su mejor aliado a los lenguajes de programación PROLOG y LISP, ambos basados en matemática discreta. Con ellos se podía modelar bucles y procesos repetitivos optimizados para la toma de decisiones; sin embargo esto no era una aplicación rigurosa de lo que se pretendía por inteligencia artificial.

Para esta misma década se produjo el nacimiento de ELIZA, el sistema desarrollado por Joseph Weizenbaum que imitaba a un psicoterapeuta en una charla habitual con un paciente, llegando a engañar a varios de ellos, ¿cumplió con el test planteado por turing? esa pregunta es aún controversial pero lo que si

es seguro es que terminó convirtiéndose en un hito de la inteligencia artificial y al mismo tiempo complicó aún más la definición de la misma, ya que ELIZA funcionaba con base en trucos de lenguaje donde se intercambiaban los pronombres haciendo pensar que se llevaba una conversación, algo lejano a lo que se esperaba fuese el mecanismo de funcionamiento de la inteligencia artificial.

Para principios de los 70 John McCarthy agregaba una nueva variable al problema de la inteligencia artificial. La cual fue bautizada como “el problema del marco”. McCarthy demostraba que mientras la aplicación de la inteligencia artificial fuese particular no iba nunca a poder tener un nivel de percepción global que sí tiene el humano y no iba a poder contar con las alternativas de solución que no siempre están implícitas en el problema a resolver. A partir de esta situación los mecanismos de percepción sensorial (que presuponen el conocimiento humano) se han hecho más populares en la inteligencia artificial, de esta manera ramas de estudio como el control, la instrumentación y la visión artificial se unieron en sistemas globales logrando una percepción del ambiente que “alimentaba” de información al “cerebro” artificial que ejecutaba la técnica.

Fue así como 1980 se convirtió en la época del boom de la inteligencia artificial donde las herramientas ya recopiladas permitieron que saliera de los laboratorios hacia la industria, creando una rama laboral con sobre-demanda, y una curiosidad social que aún en estas fechas sufre de confusión al respecto, y es que la combinación de la robótica móvil y la inteligencia artificial sumada a las expresiones culturales de cine y televisión que entienden al robot como una analogía humana mecanizada agitó el problema filosófico que auguraba el hecho de que una maquina pudiese pensar. Una muestra de ello es el libro del autor de ELIZA “Computer Power And Human Reason” donde se reconocía la posibilidad de desarrollar verdadera inteligencia artificial pero se criticaba el entorno ético y moral que pudiese acompañarla.

Buscando una definición más precisa, la propuesta de Martin Fischles y Oscar Firschein en 1987 permitió reglamentar la inteligencia artificial mediante el concepto de los “agentes inteligentes” una serie de normas que compendian el “creer”, “aprender”, “entender”, “planificar”, “conocer”, “distinguir”, “crear”, “percibir”, “expresarse” y finalmente “resolver”.

Actualmente existe un gran numero de técnicas de inteligencia artificial distintas, cada una con un contexto e idea diferente pero que al final parecen encontrarse en pos del objetivo común de lograr que un sistema actué independientemente de la intervención humana, entre ellas se encuentran las Redes Neuronales, la Lógica Difusa, Los Algoritmos Genéticos, el Aprendizaje Reforzado, y las Redes Bayesianas, por mencionar unas cuantas.

1.2.1 ALGORITMOS GENÉTICOS

Los algoritmos genéticos, son una técnica de inteligencia artificial que busca encontrar el valor máximo de una función; para lo cual genera de manera aleatoria un grupo de posibles respuestas y después aplicando operaciones sobre estas, encuentra el mejor resultado.

La característica aleatoria es la que hace de los algoritmos genéticos una técnica gustosa entre los usuarios de la inteligencia artificial, ya que dicha característica independiza el funcionamiento del algoritmo de la interacción humana con el mismo. Esto es posible dado que el programador del algoritmo indica las características que debe tener el sujeto ganador y se aísla en el resto del proceso permitiendo que sea el mismo programa el que genere las posibles soluciones y escoja la mejor. El algoritmo genético no necesita un entrenamiento previo, ni un fuerte conocimiento de la teoría del problema, lo que facilita su aplicación. Sin embargo esta carencia de soluciones anticipadas genera un tiempo de procesamiento mucho mayor.

Esta línea de inteligencia artificial nace en los años setentas, planteada por John Holland. Su origen está inspirado en la teoría de la evolución y su base Genético - molecular. Las posibles soluciones se codifican parametrizando las variables que intervienen en el problema. Los parámetros son llamados cromosomas y están formados de genes. Las soluciones son obtenidas de manera aleatoria y luego se les aplica una serie de operaciones genéticas, buscando guardar lo mejor de cada posible solución y eliminar lo peor de la misma, para al final ser evaluadas por la función a maximizar.

Los términos de gen y cromosoma son resultado de la analogía entre su significado biológico y el papel que desempeñan en el algoritmo genético. Un gen es la unidad básica estructural encargada de transmitir las características por medio de la herencia entre un grupo de individuos. Estos genes se agrupan formando cromosomas, los cuales a su vez se agrupan formando las cadenas de ADN que son las que caracterizan al individuo.

Como un ejemplo podemos ver el caso de los cromosomas "X" y "Y" de la especie humana. En estos cromosomas se encuentran los genes que dan las características sexuales del individuo; por ello son llamados cromosomas sexuales. También es de notar que la especie humana cuenta con 46 cromosomas en los cuales se encuentran los diferentes genes encargados de dar las características al hombre, lo que nos muestra de alguna manera el funcionamiento de un modelo genético; los genes guardan la información y los diferentes cromosomas guardan los genes que tiene por común denominador características relacionadas del individuo.

Se ilustrara esto con el ejemplo de la búsqueda de un sabueso perfecto. Si se deseara construir un sabueso perfecto su nariz, longitud de las piernas y contextura serian los cromosomas que definirían los problemas cuantificables como son capacidad olfativa, velocidad y resistencia. Los distintos genes que se encuentran en los cromosomas serian aquellos que darían las características de

los problemas cuantificables es decir si su capacidad olfativa es buena o mala, si es lento o rápido y si se cansa fácil o tiene mayor resistencia. Estas características se pueden cuantificar, luego ser evaluadas y de esta forma saber cual combinación de características es la que conlleva a un sabueso perfecto.

COMPONENTES

Los algoritmos genéticos poseen diversos componentes, necesarios para su correcta aplicación. Cada componente trata de reflejar de alguna manera las circunstancias biológicas en las que se desarrollan las teorías evolutivas, tales como el apareamiento, las mutaciones y las pruebas que el entorno les impone a los individuos; todo esto con el fin de mantenerse cercano a la naturaleza misma, y así replicar de forma aproximada los resultados que se ven en el entorno biológico.

- **FUNCIÓN DE EVALUACIÓN**

La función de evaluación es la parte fundamental de un algoritmo genético, pues dicha función es la que estamos evaluando, tratando de encontrar el valor máximo. Esta función de evaluación puede tener varios máximos, el algoritmo genético encontrara alguno de ellos dependiendo de los intervalos en los que se encuentren las primeras soluciones aleatorias.

La función de evaluación no necesariamente debe ser una expresión matemática, también puede ser una expresión lógica que tenga por resultado una respuesta más optima que otras.

La función de evaluación es aplicada a los diferentes individuos generados por el algoritmo.

- **OPERACIONES GENÉTICAS**

Las operaciones genéticas son una serie de modificaciones que se le realizan a los individuos con el propósito de mejorarlos y producir nuevos individuos

Se diferencian dos operaciones genéticas fundamentales, operación genética de cruce y la operación genética de mutación. La primera es aplicada a grupos de individuos mientras la segunda es aplicada de manera individual. Existen otras operaciones que han complementado el algoritmo en la búsqueda de optimizar su funcionamiento y hacerlo más cercano a su equivalente biológico.

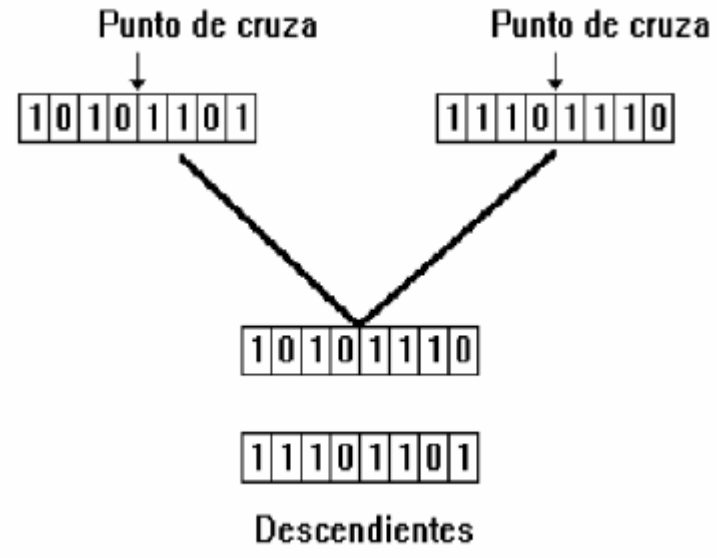
- **OPERACIÓN GENÉTICA CRUCE**

La operación genética cruce se encarga de tomar dos individuos escogidos de manera aleatoria, dando prioridad a los mejores individuos, combinar su material genético y de esta manera obtener dos individuos nuevos.

Existen diversos tipos de procedimientos y formas de realizar el cruce y escoger los individuos. Tantos como los que pueda diseñar el programador del algoritmo genético. Entre ellos tenemos, con uno o dos puntos de cruce. El punto de cruce es la ubicación de cromosomas desde donde se intercambia el material genético. La forma de escoger este punto de cruce es totalmente aleatoria. Otras maneras de realizar el cruce son de forma uniforme donde en cada bit se elige al azar un padre para que contribuya con su bit al del hijo, mientras que el segundo hijo recibe el bit del otro padre. PMX, SEX estos son operadores más sofisticados fruto de mezclar y aleatorizar los anteriores.

La manera de escoger la pareja de individuos debe cumplir con asignarles mayor prioridad a los mejores individuos del grupo generado

Figura 1. Método de cruce de un solo punto



Fuente. [4]

- **OPERACIÓN GENÉTICA MUTACIÓN**

La mutación es la operación genética encargada de simular los procesos aleatorios de mutación, donde un individuo sufre un cambio en algún gen de su cadena genética, lo que puede producir un mejoramiento o daño en el individuo.
El individuo o individuos y el gen o genes que sufren la mutación se escogen de manera aleatoria.

Figura 2. Ejemplo de mutación.

- **OPERACIÓN GENÉTICA EXTINCIÓN**

Esta nueva operación genética es desarrollada debido a la necesidad de incrementar la velocidad de respuesta del algoritmo.

Una raza es una subdivisión de las especies. Las razas poseen características comunes entre si, pero se diferencian en el grado de especialización que tengan en determinadas áreas, lo que les da características físicas diferentes; de esta forma si consideramos a las soluciones del problema planteado una especie, podemos también considerar a la primera generación de individuos una raza de esta especie, y al igual que en un proceso evolutivo si la descendencia de esta raza no prospera, es probable que no logre su supervivencia, sin tener en cuenta factores extremos, y la raza se extinga. Sin embargo esto no necesariamente significa el fin de la especie, pues la aparición de otras razas mejores puede darle continuidad a la misma.

De esta manera si una raza de soluciones no genera un vencedor, entonces esta raza es eliminada y se produce una nueva.

- **OPERACIÓN GENÉTICA SOBREVIVIENTE**

Dado el alto número de iteraciones que tiene un algoritmo genético y debido a que no posee una base de datos inicial, se hace necesario aprovechar los resultados, que más se aproximen a la solución final aún cuando estos no sean la respuesta mínima exigida. Observando esto, en Alemania se desarrolló este proceso, donde el más fuerte de los individuos lograba superar las operaciones de cruce y mutación, llegando intacto a la siguiente generación; de esta forma se creaba una especie de base de datos donde las mejores respuestas ofrecían su material genético para la siguiente generación. Dicha técnica se denominó elitismo.

Esta estrategia se modificó para complementar el funcionamiento de la operación genética extinción. En este caso el mejor individuo de cada raza pasa intacto a la primera generación de la raza siguiente. Dadas estas modificaciones se le denominó operación genética sobreviviente.

2. HERRAMIENTA SOFTWARE PARA LA PLANEACIÓN DE TRAYECTORIAS

En este capitulo se describe el desarrollo de la herramienta basada en algoritmos genéticos que se utilizó para solucionar el problema del planeamiento de trayectorias en un ambiente controlado (ver Anexo D), igualmente se especifican las características principales del algoritmo genético como lo son su codificación, su función de evaluación y la metodología utilizada para implementar las distintas operaciones genéticas involucradas, junto con los diagramas de flujo que representan estas rutinas. Finalmente se presenta y describe la herramienta FREYJA como producto final del desarrollo del algoritmo utilizado haciendo enfasis en los aspectos inherentes a la programación, que se vieron implicados en la construcción de la herramienta.

2.1 DEFINICIÓN DEL ALGORITMO GENÉTICO

2.1.1 CODIFICACIÓN

Una vez planteado un problema que se desea solucionar con algoritmos genéticos, se realiza como primer paso la escogencia de una codificación de las características básicas que pueden dar solución al problema. En el caso del planeamiento de trayectorias se trata pues de las direcciones a seguir por el robot móvil, ya que son estas las que describen una trayectoria. A partir de ellas se definen entonces las características que serán llamadas cromosomas. La codificación se hace en binario, dado que estos estados son excluyentes entre si, o vale 1 o vale 0, de ninguna forma aparecen estados intermedios.

Si volvemos al ejemplo del sabueso perfecto se tiene que:

La codificación de estos genes debe ser exclusiva, el cromosoma nariz tendría los siguientes genes como unos y ceros 00-> Olfato Malo, 01->Olfato Bueno, 10->Olfato Medio, 11-> Olfato Nulo. Según esta representación, los genes son los números binarios y las parejas que representan una cualidad son los cromosomas.

Se pueden codificar entonces, el largo de las patas para poder medir el desarrollo de la velocidad, al igual que la resistencia muscular etc..., hasta al fin obtener un genoma del sabueso que nos describa aquello en lo que estamos interesados.

Si se piensa en una trayectoria se debe hacer énfasis en como definirla en términos de sus componentes fundamentales (genes) para esto se trabajó en la codificación binaria que permite ordenar como cromosomas cualidades como su ubicación en el eje cartesiano y la longitud de la trayectoria. Los cromosomas escogidos representan:

- Cromosoma α : Monotonía en el eje X o Y de la trayectoria.
- Cromosoma ß: Dirección.
- Cromosoma δ : Distancia.

La estrategia de codificación de los genes para los cromosomas es la que se observa en la Tabla 1, ver también figura 3.

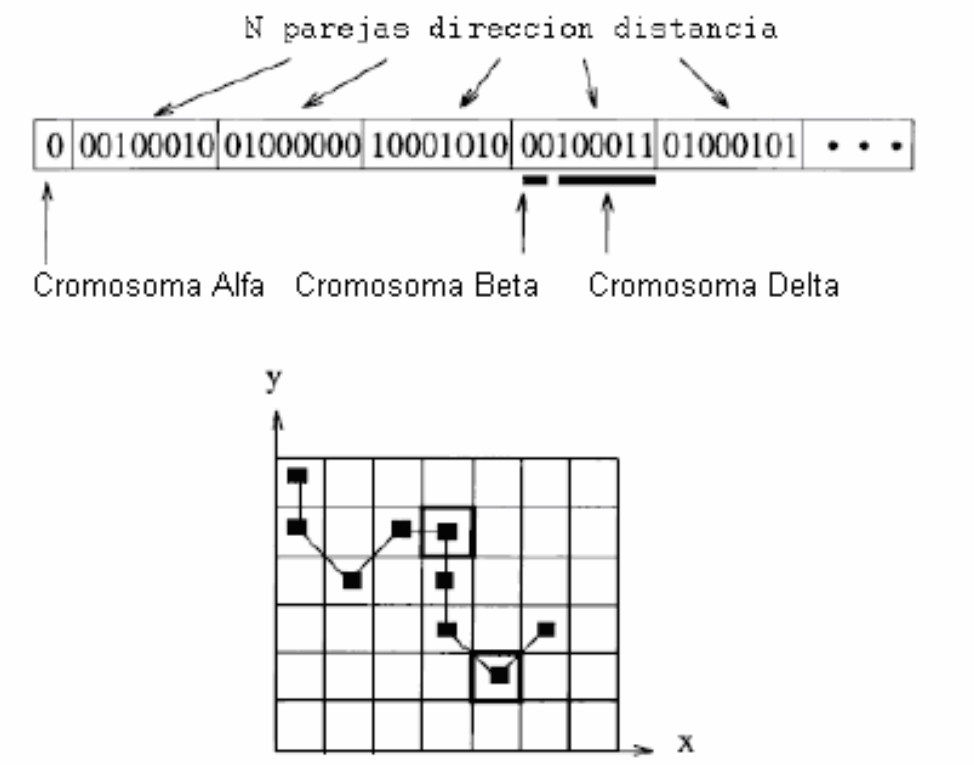
Tabla 1. Codificación y cromosomas

Cromosoma	Codificación	Característica que representa
α	0	MX (Monótono en X)
	1	MY (Monótono en Y)
ß	00	Vertical para MX y horizontal para MY
	01	Diagonal superior para MX Diagonal izquierda para MY
	10	Horizontal para MX Vertical para MY
	11	Diagonal Inferior para MX Diagonal derecha para MY
. δ		Según ecuación (1)

Debido a que el problema se desarrolla en un territorio delimitado, para encontrar el camino el territorio es divide en una cantidad de celdas N a lo largo y ancho del mismo. Los cromosomas dirección y distancia se agrupan (parejas ßδ) y se interpretan como la representación de los pasos de distancia variable. Cuando el cromosoma de dirección sea codificado en "00" el cromosoma δ puede ser positivo o negativo, es decir el primer gen de este cromosoma es 0 para positivo o 1 para negativo. Si es positivo y el individuo es MX representará una vertical positiva hacia arriba seguida de una diagonal superior derecha. Si es negativo representará una vertical hacia abajo seguida de una diagonal inferior derecha. Si es positivo y el individuo es MY representara una horizontal positiva hacia la derecha seguida de una diagonal inferior derecha, si es negativo representara una horizontal negativa hacia la izquierda seguida de una diagonal inferior izquierda. [3]

Ecuación 1. $1 + Log_2 N = genesCromosoma\delta$

Figura 3. Ejemplo de trayectoria codificada.



Fuente. [3]

2.1.2 FUNCIÓN DE EVALUACIÓN

Al momento de escoger la función de evaluación se tiene en cuenta el objetivo principal del algoritmo. Debido a que lo que se busca con un algoritmo genético es maximizar una función, entonces se debe plantear la función de evaluación de tal manera que un mayor resultado implique un mejor camino. Para ello se utiliza una maximización de la función de la distancia recorrida por el camino, comparada con la distancia máxima que puede recorrer el camino para ir de un punto de inicio a un punto de llegada particular, sin salirse de los límites de la pista. Dicha distancia máxima es acotada por su codificación binaria, en el caso de un cromosoma δ con 5 genes (5 bits), 31 parejas dirección-distancia. La distancia recorrida por el robot controlado en un paso es normalizada, así cada celda tiene un tamaño de 1 X 1, y en los pasos diagonales 1.41 unidades, si la pista es de 2 metros de ancho por 2 metros de largo, y se divide en 32 celdas iguales, cuadradas; entonces el valor del

paso real del móvil es de 6.25cm. Es decir, por cada paso que el algoritmo genético le ordene dar el móvil es desplazado 6.25cm en línea recta o 8.83cm en la diagonal.

Ecuación 2. $Peso = 200 - Distancia\ Camino$

La ecuación 2 da como resultado el valor maximizado en función de la distancia y es llamado peso del camino, esta ecuación es fruto de las observaciones y ajustes empíricos realizados a la ecuación sugerida por los autores [3], en la cual el peso es igual a la máxima distancia posible codificada menos el largo del camino, al observar que dicho valor es muy grande y que los caminos solución no podían ser muy largos se recortó dicho valor de distancia máxima de 200. Expulsando así toda respuesta superior a este valor al considerar el camino demasiado largo, la función de evaluación busca el valor máximo positivo, luego si aparece un camino con un valor de distancia superior a 200 este será rechazado inmediatamente. En el caso de un planeamiento de trayectorias es lógico que la característica más importante es la capacidad que tenga el camino de llegar a la meta a partir de un origen dado. Esto debe verse reflejado en la función de evaluación y considerando que es un punto critico, se escoge que cualquier camino que no llegue a la meta se le castiga dividiendo su peso en 3 como resultado de su función de evaluación. Otra condición a cumplir es que el camino nunca debe chocar con un obstáculo; a estos caminos se les asigna un valor de peso igual 1, este valor es definitivo por lo tanto no es necesario analizar los demás criterios para un camino con ese peso, lo que ahorra tiempo de computo.

Debido a características ajenas a la programación del algoritmo tales como la forma del carro o el margen de error del software de visión encargado de proveer la ubicación del móvil y los obstáculos en la pista, se hace necesario demarcar una zona de riesgo alrededor de los obstáculos. Dicha zona representa posibles ubicaciones por las cuales no es conveniente que pase el camino auque puede pasar por allí en caso de no haber más soluciones, así que se castiga el paso por la zona de riesgo dividiendo su peso en 2.

Los caminos que sobrepasan los limites de la pista no son utilizables debido a que es una zona prohibida para el transito de un robot móvil real; sin embargo si no tienen choques con obstáculos podrían ser de utilidad en el proceso de cruce del material genético, así que no se eliminan; solo se les reduce su peso lo suficiente como para evitar que sean identificados como solución pero que aún tengan probabilidad de cruzarse. Esto se logra dividiendo su peso en 3.

Los distintos criterios se aplican en el siguiente orden:

- Primero: Choques con obstáculos.
- Segundo: Salidas de la pista.
- Tercero: Zonas de riesgo.
- Cuarto: No llegada a la meta.

La aplicación secuencial provoca un castigo más severo a aquellos caminos que violen más criterios, permitiendo que dichos caminos no sean escogidos por el algoritmo como solución.

La función de evaluación es aplicada a cada uno de los individuos. Dependiendo de qué tan elevado sea el resultado, estos serán mejor solución al problema. El valor resultante de la función de evaluación es llamado peso del individuo.

Los distintos valores de castigo usados son producto de pruebas realizadas al algoritmo con una serie de valores, buscando obtener una mejor velocidad de respuesta y mayor calidad en los caminos. Los procesos aleatorios llevados a cabo a nivel interno del algoritmo, buscando hacer impredecible al mismo, también permiten escoger los valores según criterio del programador.

2.1.3 CRITERIO DE PARADA.

Dado que la aplicación real de los algoritmos genéticos en el planeamiento de trayectorias exige un tiempo mínimo de procesado para de esta forma poder aplicar la trayectoria obtenida de manera inmediata, y así evitar alteraciones muy grandes del entorno en el que se desenvuelve el móvil controlado, se hace necesario definir un criterio o grupo de criterios para detener las iteraciones del algoritmo genético, para asegurar en lo posible que el camino seleccionado cumpla con un mínimo de exigencias. El criterio más idóneo es el valor de la función de evaluación. Este criterio es puesto por el usuario y consiste en fijar el peso mínimo que puede tener un camino para ser considerado una respuesta apta para el problema. Si obtenemos un camino que supere o sea igual a dicho valor, este será considerado como la solución y las iteraciones del algoritmo se detendrán. A la hora de definir el peso mínimo es de recordar que entre más alto sea el peso del individuo, implicara una mejor respuesta pero a la vez tomará mayor tiempo encontrarlo.

Cuando el algoritmo se enfrenta a un entorno variable, por ejemplo un obstáculo móvil, la posición actual del móvil controlado pasa a ser la posición inicial, se definen automáticamente los parámetros de funcionamiento del algoritmo tales como tasa de cruce, mutación, peso mínimo y número de generaciones e individuos, buscando que la velocidad con la que el algoritmo calcula el camino sea máxima, sacrificando la búsqueda de un camino optimo y permitiendo que caminos no muy buenos sean considerados respuesta.

2.1.4 OPERACIÓN CRUCE

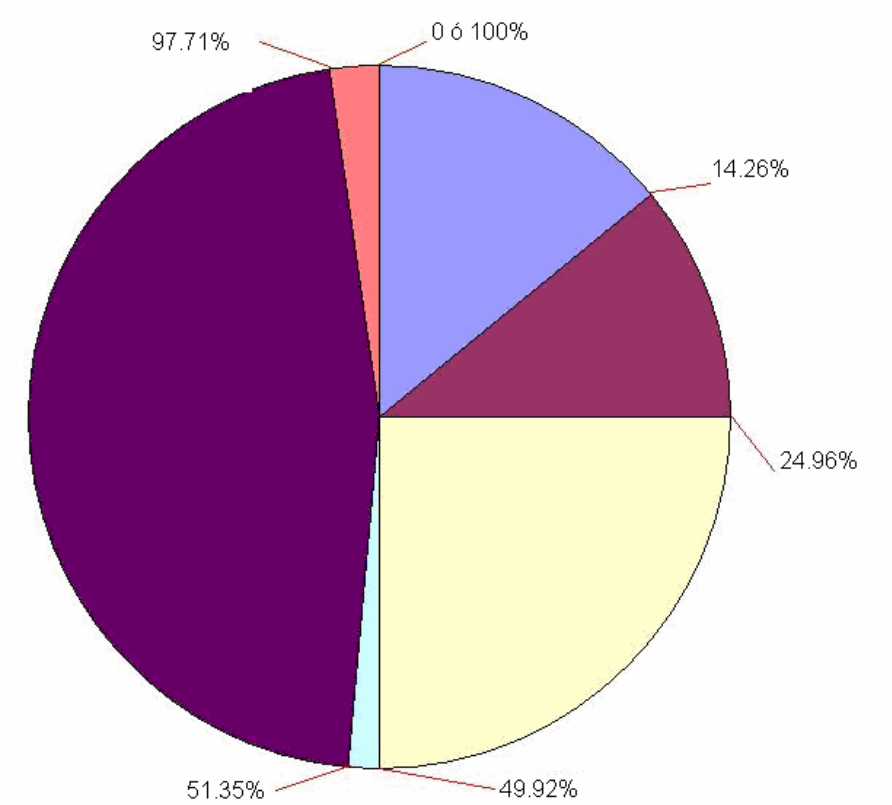
Se utiliza el método de cruce de un solo punto, debido a que al ser simple es también más veloz, lo que permitiría realizar el cruce de una gran cantidad de individuos.

Una vez se han evaluado los individuos, estos pasan a la piscina de apareamiento, donde están los individuos que han sido seleccionados para aparearse. El método de selección es tipo ruleta. Primero se calcula la suma total de los pesos de los individuos, luego se calcula el porcentaje que representa un individuo respecto del total y se ubican en una especie de torta. Cada porción de la torta representa un individuo de la población. Los límites de la misma están dados por la sumatoria del porcentaje de peso de los individuos. Ver tabla 2 y figura 4.

Tabla 2. Piscina de Apareamiento

Individuo	Peso	Sumatoria de porcentaje de pesos
1	200	14.26%
2	150	24.96%
3	350	49.925%
4	20	51.35%
5	650	97.71%
6	32	100%
Sumatoria de pesos total	1402	

Figura 4. Piscina de apareamiento



En la tabla 2 se observa un listado que ejemplifica el resultado de una función de evaluación del algoritmo genético. Está compuesta por los datos de seis individuos con sus respectivos pesos; la sumatoria total de los pesos y la sumatoria del porcentaje de los pesos de los individuos.

En la figura 4 se observa la distribución en la piscina de apareamiento de los individuos del ejemplo. El primer individuo se encuentra entre 0% y 14.26%, el segundo entre 14.26% y 24.96%, y así sucesivamente hasta llegar al 100%. Los diferentes rangos se encuentran abiertos en la cota inferior y cerrada en la superior.

Una vez asignadas las ubicaciones en la piscina de apareamiento, se obtiene un número aleatorio entre cero y cien; este número indica cual es el individuo que se

cruza al compararlo con los intervalos de porcentajes asignados en la torta. Debido a que el rango que los individuos ocupan de la piscina de apareamiento es proporcional a su peso, las probabilidades de obtener un individuo de alto peso son mayores que la de los individuos de bajo peso. Este proceso se repite un número de veces igual a la cantidad de individuos que se tienen.

Antes de realizar el siguiente procedimiento es necesario que el usuario defina una tasa de cruce global. Esta tasa de cruce representa la probable cantidad de individuos que serán cruzados. La constante propia de cada individuo se obtiene de manera aleatoria; si la constante del individuo es mayor que la tasa de cruce global, el individuo no se aparea. Si es inferior o igual se aparean. Las parejas que se cruzan se escogen de manera secuencial, el primero individuo con el segundo, el tercero con el cuarto y así sucesivamente.

Una vez realizado el calculo de la tasa de cruce de las parejas se obedece la siguiente regla: dado el caso de que la tasa sea mayor los individuos duplicaran su código y pasaran como descendientes a la siguiente generación, si la tasa es inferior a la tasa establecida de cruce entonces se define un punto de cruce de forma aleatoria. El punto de cruce no puede ser 0 o el máximo valor de genes del individuo pues esto implica que el individuo no intercambia código con el otro, solo se transmiten los dos completos. Ver Figura 1.

2.1.5 OPERACIÓN MUTACIÓN

Una vez obtenidos los nuevos individuos se procede a la operación mutación. Lo primero es asignar una variable global, cuyo valor se encuentre entre cero y uno, llamada tasa de mutación. Esta tasa de mutación representa la cantidad probable de individuos que mutan; cuanto más alta más individuos mutarán. Esta variable la define el usuario. El siguiente paso es obtener de manera aleatoria una tasa de mutación para cada individuo; a continuación se compara la tasa de mutación de cada individuo con la tasa global de mutación, los individuos que tengan una tasa

individual de mutación mayor que la tasa de mutación global no mutarán, los que la tengan menor o igual se les aplicará el proceso de mutación.

En el proceso de mutación se le asigna un gen de mutación, de manera aleatoria, a cada individuo, luego este gen es cambiado por su complemento, es decir si el gen tenia un valor igual a 0 este mutara a un valor igual a 1 y viceversa, de esta forma se alteran las características del camino dependiendo de en que gen ocurría la mutación. Los resultados de estas mutaciones son impredecibles, provocando tanto individuos débiles como también produciendo individuos fuertes.

2.1.6 OPERACIÓN EXTINCIÓN

Dado que el número de generaciones necesarias para obtener la solución de un problema es elevado y que esto implica un tiempo alto de procesamiento, y que este número de generaciones es función de que tan lejana se encuentre la población inicial de la respuesta, se diseñó esta nueva operación genética, que permite al usuario programar un número máximo de generaciones para que el algoritmo encuentre una respuesta, si el algoritmo no encuentra en este número máximo basándose en el criterio de parada, la población y su descendencia se consideran extintos, y se procede a realizar de nuevo la producción de la generación inicial y a aplicarle a esta las demás operaciones genéticas. Este proceso es repetido hasta obtener un individuo válido como respuesta. El número de veces que se reinicia el algoritmo debido a esta operación es llamado número de razas.

2.1.7 OPERACIÓN SOBREVIVIENTE

Esta técnica se aplica buscando una mejora en los individuos resultantes y una mayor velocidad en la obtención del individuo ganador. Esto se hace guardando el mejor individuo de cada raza, este individuo sobreviviente pasa a formar parte de la generación cero de la nueva raza aportando su material genético, este individuo

es escogido en la ultima generación de su raza usando como criterio el valor resultado de la función de evaluación, siendo el que tenga el mayor valor.

2.1.8 DIAGRAMAS DE FLUJO DEL ALGORITMO GENÉTICO

Figura 5. Diagrama General

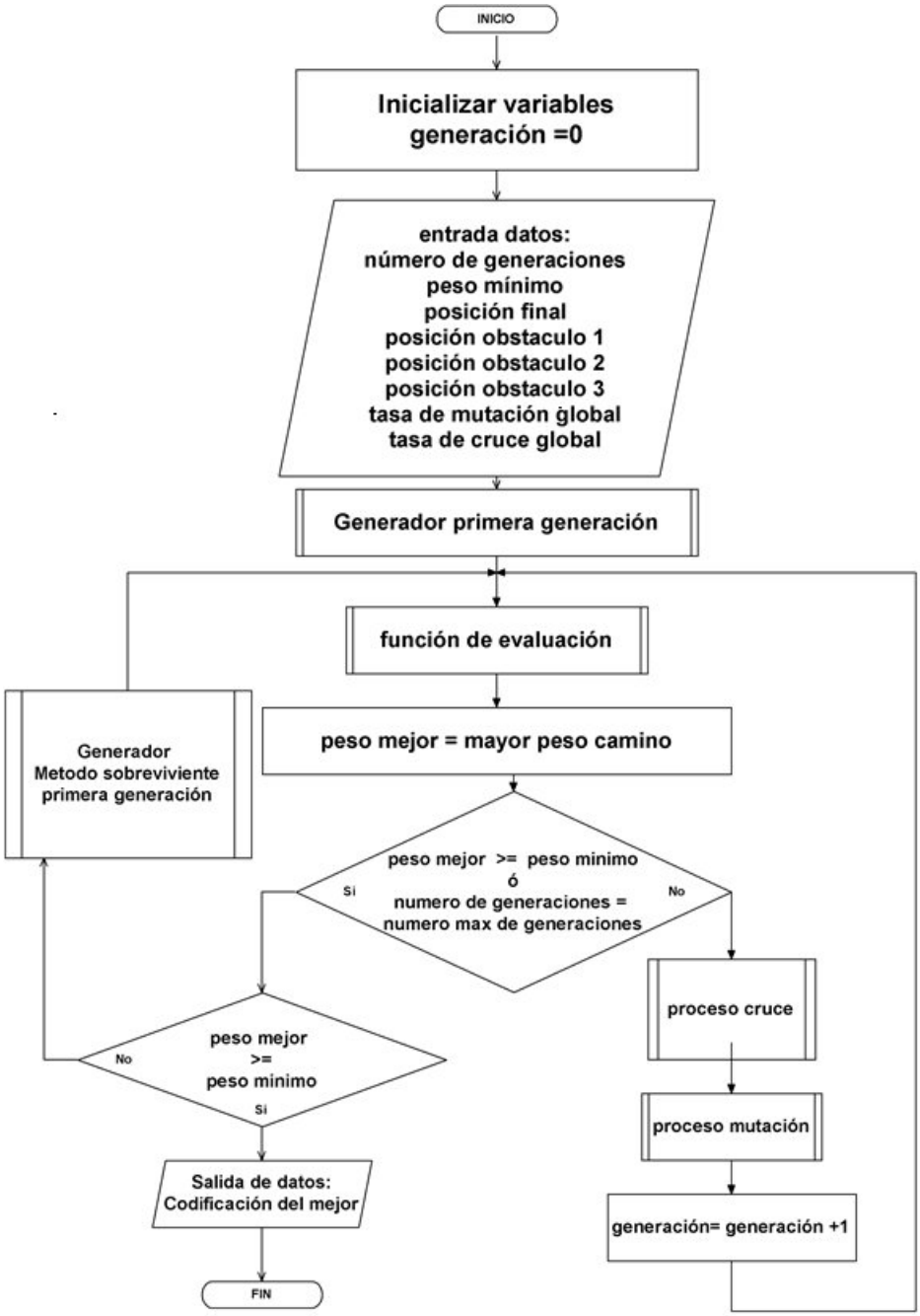


Figura 6. Primera Generación

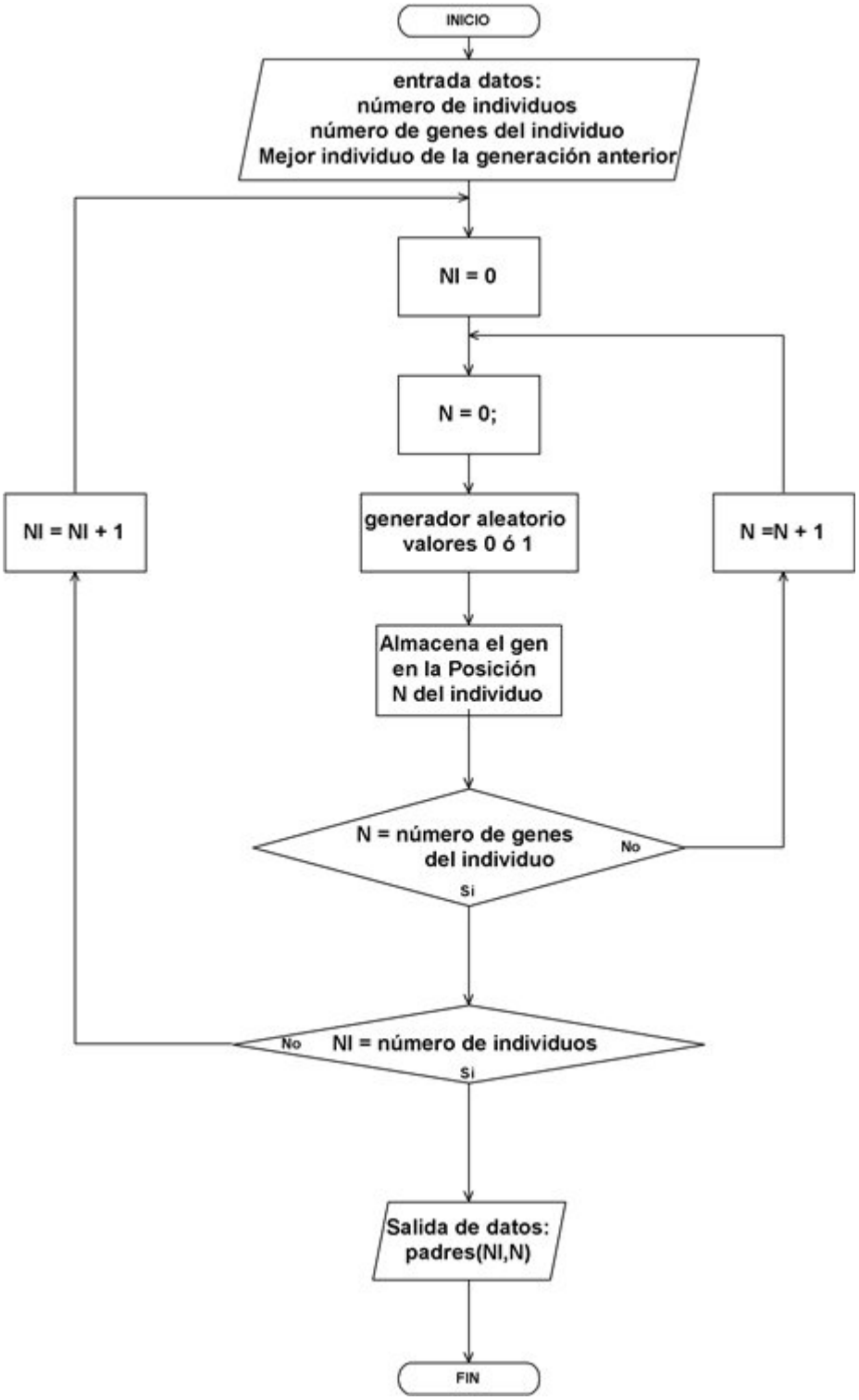


Figura 7. Función de Evaluación

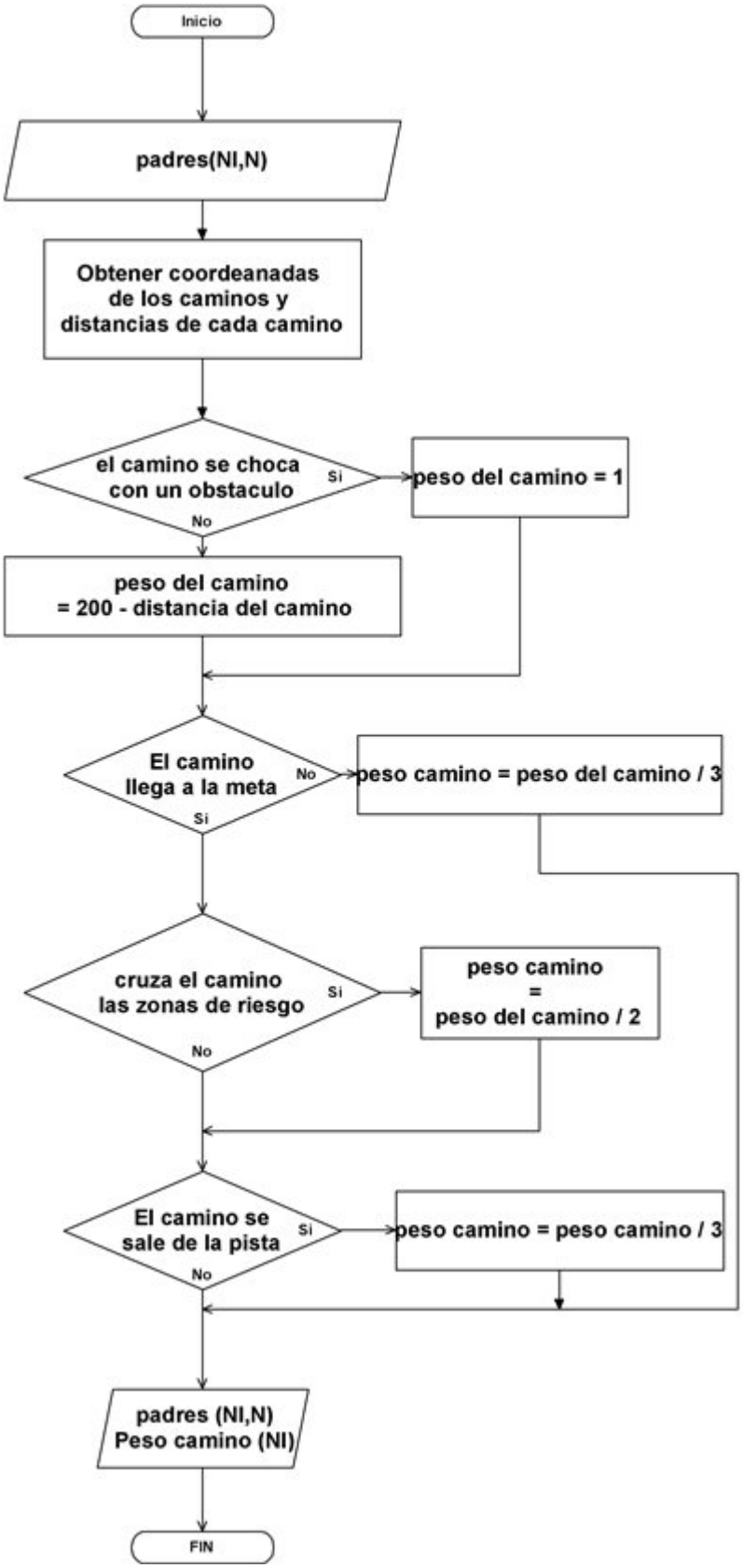


Figura 8. Cruce

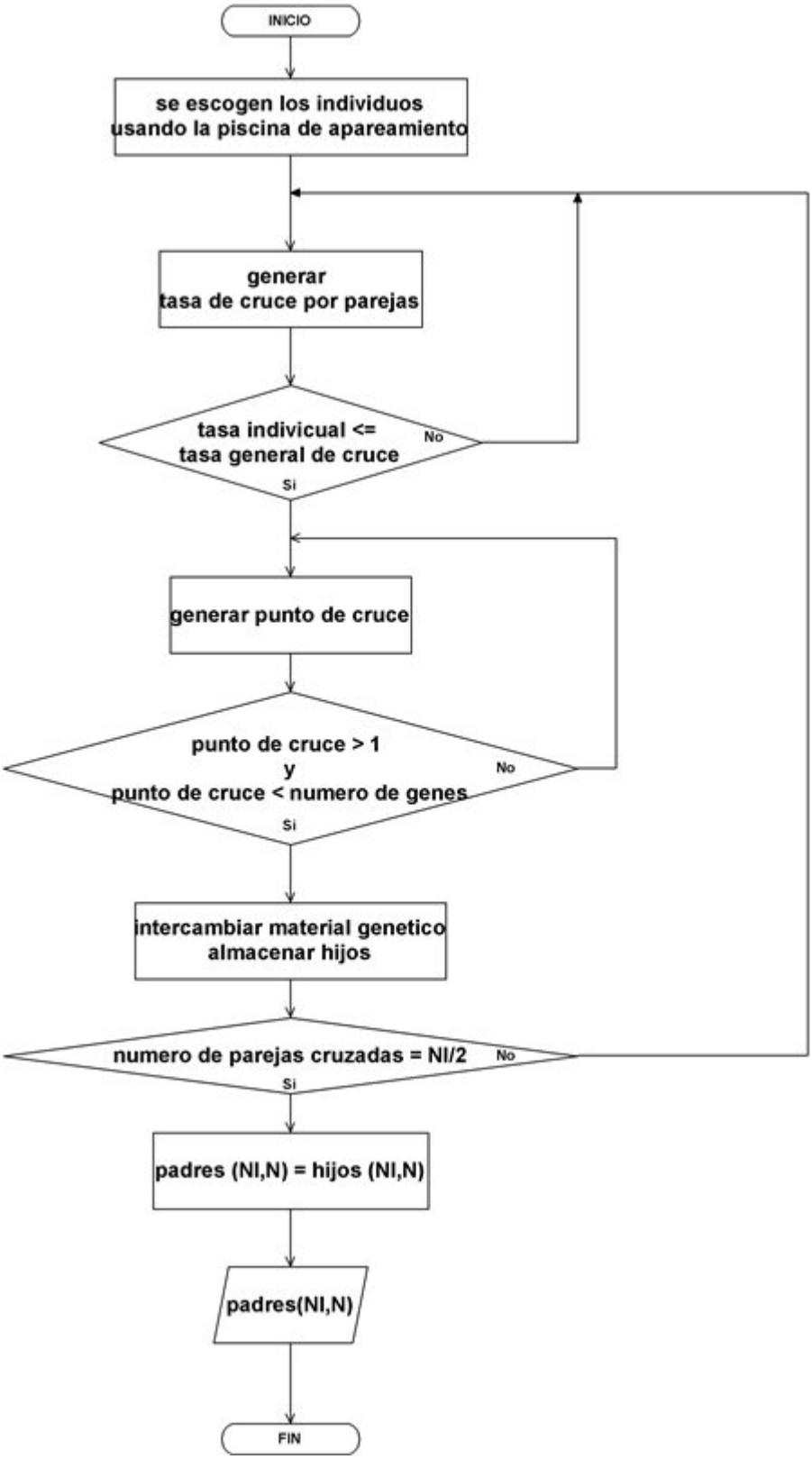


Figura 9. Mutación

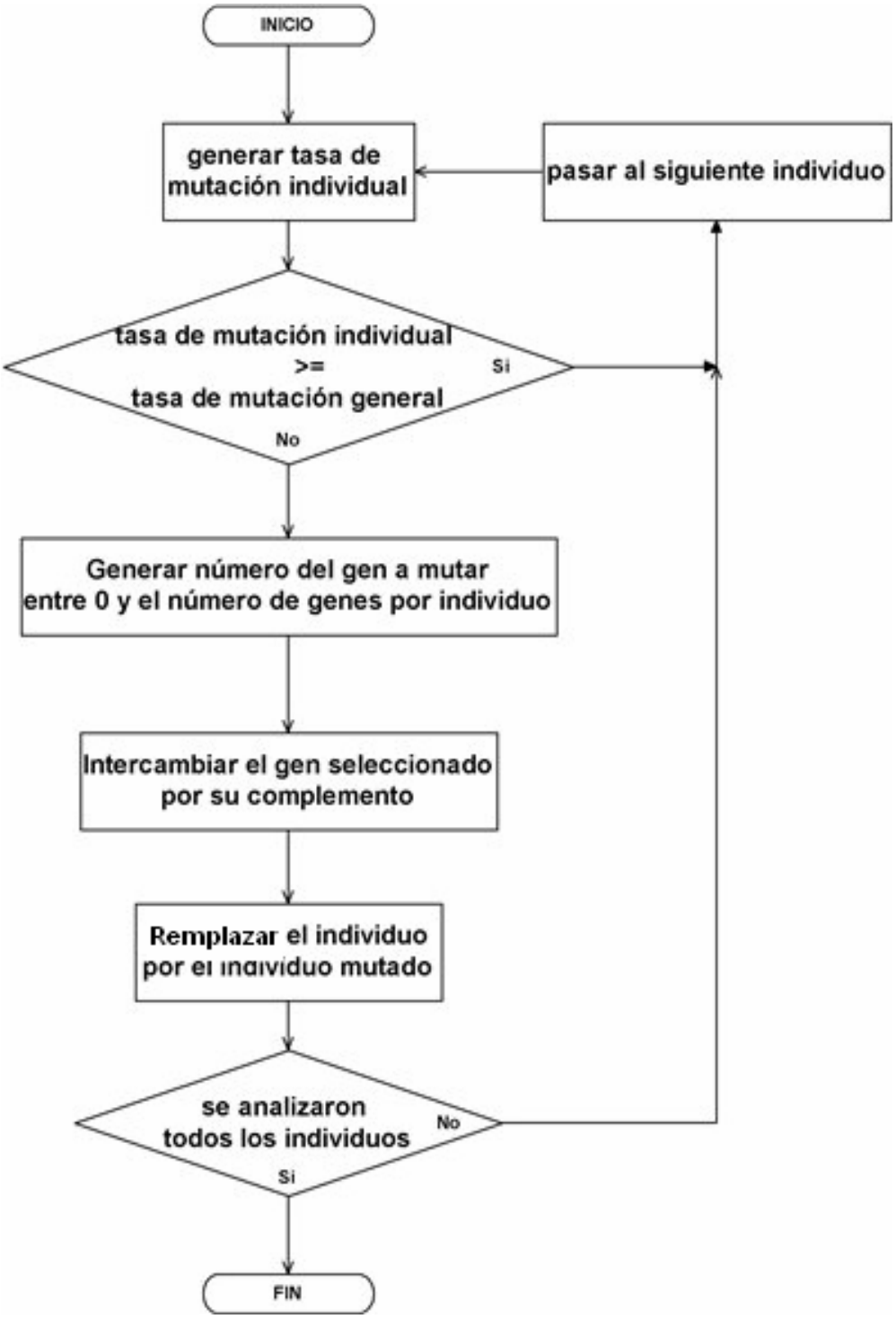
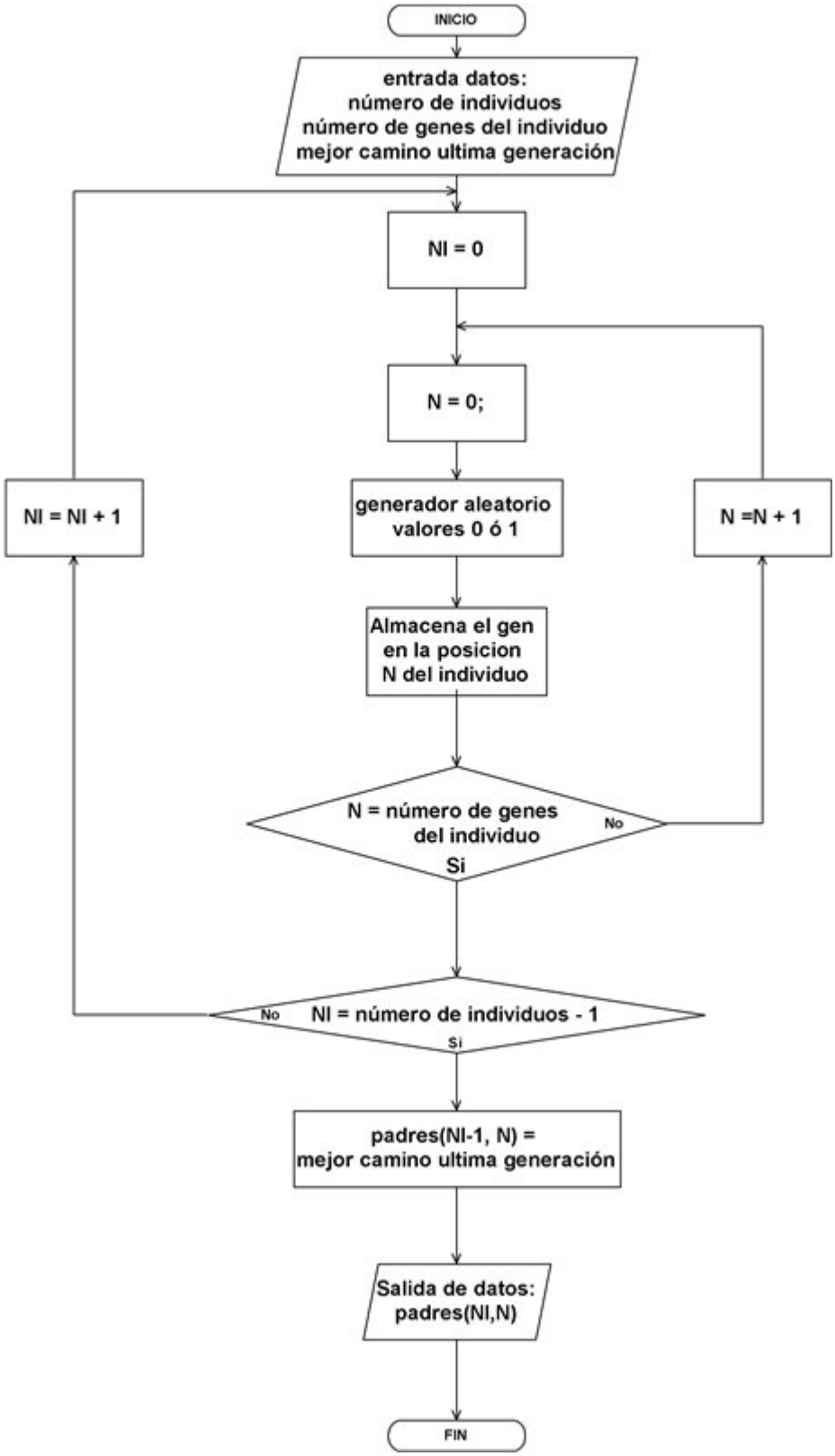


Figura 10. Sobreviviente.



2.2 IMPLEMENTACIÓN DEL ALGORITMO GENÉTICO

Teniendo en cuenta la necesidad de un prototipo para implementar la codificación seleccionada para el algoritmo genético, se desarrolló una aplicación programada en C plano y ejecutable desde el modo consola. Esta aplicación consta de una serie de rutinas que representan las operaciones genéticas fundamentales y su función de evaluación, las cuales permiten ejecutar el algoritmo genético simple variando los parámetros más esenciales (tasa de cruce y tasa de mutación) así como la posición de los diferentes obstáculos. Sin embargo las limitaciones de esta aplicación impiden que tanto los obstáculos como los parámetros puedan ser variados sin recompilar el programa, además, las trayectorias que genera están codificadas en sistema binario y deben ser reinterpretadas como coordenadas espaciales para finalmente ser modeladas con un software graficador. Teniendo como base esta aplicación se iniciaron una serie de pruebas que buscaban los individuos más representativos (trayectorias) variando los parámetros del algoritmo, de estas pruebas se obtuvieron los parámetros ideales que debía manejar la función de evaluación para que el algoritmo genético arrojara respuestas correctas según la codificación seleccionada, no obstante el tiempo de proceso era aún demasiado alto y el programa era difícil de manipular por el usuario.

Figura 11. Modo Consola

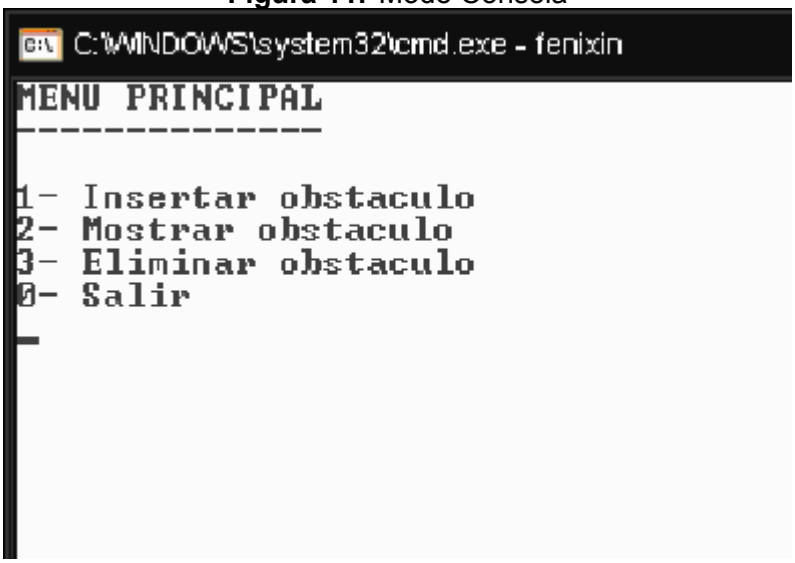


Figura 12. Resultados Modo Consola

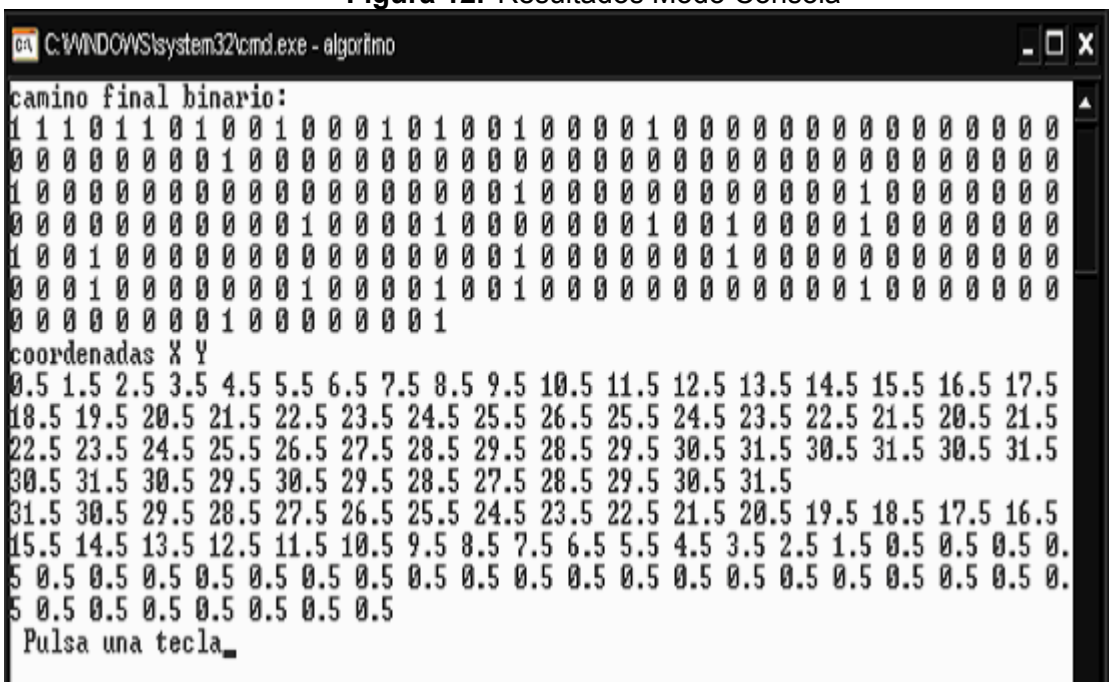
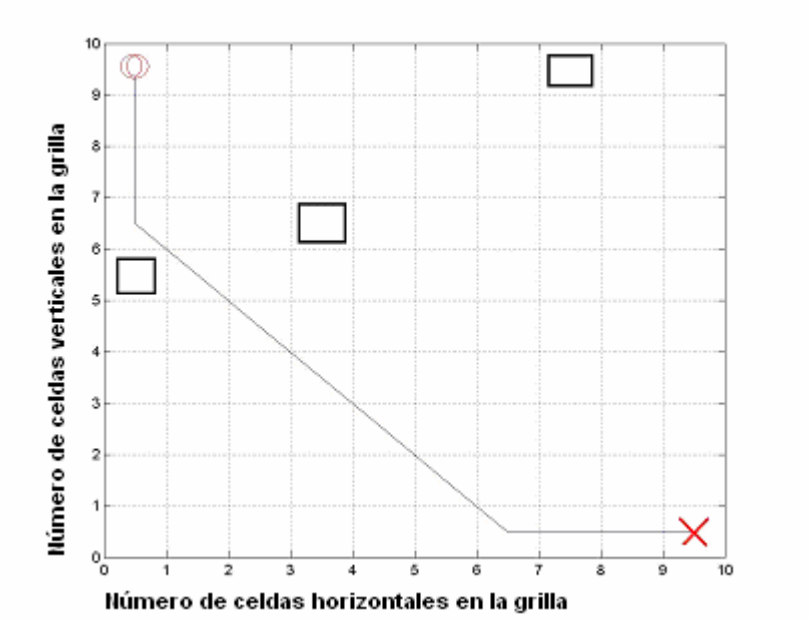


Figura 13. Resultados Visualizados en Matlab



Se debía mejorar la implementación del algoritmo para aumentar la velocidad de ejecución y la amabilidad hacia el usuario, además de integrarlo a los robots disponibles, al sistema de visión y al modulo de radiofrecuencia recreando la aplicación del planeamiento de trayectorias completamente. Por ese motivo se

inició la migración al lenguaje C# y por consiguiente a un paradigma de programación orientado a objetos. El uso de los objetos (clases, eventos, funciones) permitió organizar una ejecución paralela de la estructura lógica del algoritmo haciéndolo acorde con el estado del arte de los mismos [14], los resultados fueron ideales ya que al tener un entorno gráfico y un modo de simulación se pudieron hacer más pruebas que permitieron refinar aún más la función de evaluación, y con las mejoras en el código el tiempo que se gano en la ejecución se aprovechó para incluir nuevos elementos al algoritmo que lo diferenciaban de un algoritmo genético simple, estas inclusiones fueron las operaciones genéticas “extinción” y “sobreviviente”. Además, se le adiciono la capacidad para generar trayectorias sobre la marcha con la función “SkipMovil” pudiendo de esa forma planear trayectorias en ambientes con obstáculos móviles; esta adición le dio fuerza a la implementación online del sistema donde se capturaba la posición de los obstáculos desde un sistema de visión y se enviaba la trayectoria codificada por medio de un modulo de radiofrecuencia. Finalmente teniendo en cuenta el objetivo didáctico del sistema se le dio un cuidado especial a las validaciones y opciones de configuración de sistema haciéndolo así fácilmente operable por un usuario final que desconozca el tópico. A esta aplicación se le llamo FREYJA.

2.2.1 FREYJA

FREYJA es el resultado de la implementación del algoritmo genético abordado en secciones anteriores en una plataforma de programación a alto nivel; los algoritmos genéticos usualmente se presentan a la comunidad desarrollados en lenguajes de marco lógico como LISP, sin embargo el uso del modelo de programación orientada a objetos se ha vuelto cada vez más popular en el desarrollo de los mismos, lo que ha permitido la aparición de soluciones implementadas en lenguajes diferentes como la API “JavaGenes” desarrollada por la NASA.

FREYJA corresponde al nombre de la deidad nórdica de la fertilidad en alusión a las analogías reproductivas que maneja el algoritmo genético.

2.2.2 PROPÓSITO

El propósito principal de FREYJA como solución basada en algoritmos genéticos, es manejar el planeamiento de caminos en un ambiente online (generar el camino y ordenar la ejecución de la trayectoria) , sin embargo no se puede dejar a un lado propósitos ligados del software como lo es el plantearse a si misma como una plataforma de aprendizaje e investigación sobre computación evolutiva, ya que la capacidad de FREYJA de funcionar no solo en ambientes online sino también en ambientes simulados así como la completa parametrización de las variables principales de la técnica genética, lo convierten en un modelo ideal para el aprendizaje de este tipo de algoritmos.

Esto sin mencionar la amable plataforma gráfica que facilita al usuario final el manejo y control de la aplicación, enfocándolo así en los análisis fundamentales que generan conclusiones determinantes acerca del algoritmo como tal.

FREYJA se desarrolló teniendo en cuenta el modelo de programación orientada a objetos, por ello el código resultante de la misma es un código ordenado y fácil de entender permitiendo, inclusive, extraer de ella la librería fundamental que contiene las operaciones genéticas separada de la que maneja la función de evaluación (los dos bloques fundamentales del algoritmo genético).

2.2.3 LENGUAJE C#

Desarrollado por Microsoft y parte fundamental de la suite Visual Studio.NET, C# (pronunciado C NUMERAL ó C SHARP) se ha erigido dentro de la comunidad desarrolladora como un lenguaje popular y se le reconoce de cierta forma como la evolución del C++. Algunas de las características que hacen a este lenguaje sobresalir son:

- Soporte de estructuras y componentes (Accesibilidad a diferentes modelos, por ejemplo un componente ActiveX)
- Garbage Collector (Manejo inteligente de la memoria disponible)
- Código Seguro (asignaciones de variables explicitas que disminuyen la probabilidad de desbordamientos por errores en tipo de datos)
- Aplicabilidad de la OOP (encapsulación, herencia y polimorfismo fácilmente implementados)
- Documentación XML sobre la marcha

Además de esto la ejecución de C# sobre una maquina virtual (Framework .NET) permite un eficiente uso de la memoria que consume del sistema.

Actualmente no solo la suite .NET de Microsoft puede dar soporte a C#, ya que se encuentra regulado y estandarizado como un lenguaje independiente, inclusive diferentes alternativas libres han venido desarrollando compiladores y e IDE para C#, sobre Windows tenemos a SharpDevelop (http://www.icsharpcode.net) y sobre Linux a MONO (http://www.mono-project.com) y a DOTGNU (http://www.dotgnu.com)

2.2.4 REQUERIMIENTOS

Los requisitos de sistema para el funcionamiento de FREYJA son:

- Microsoft Windows XP.
- .NET Framework Versión 1.1.
- Microsoft Communication Center (Disponible como componente individual).
- 256 MB RAM.
- Procesador Pentium 4 o equivalente.
- Puerto serial disponible.
- Modulo de radiofrecuencia adaptable al puerto serial.

- Sistema Fenix funcionando si se desea planeamiento online.

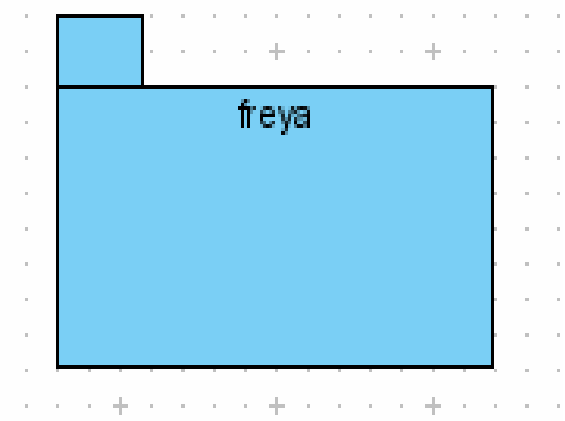
Los requisitos del móvil a controlar son:

- Módulo compatible de radiofrecuencia.
- Capacidad de interpretar la codificación usada por Freyja.

2.2.5 DIAGRAMA DE CLASES

Freyja consta del paquete principal FREYA y de las clases asociadas con sus respectivos métodos, eventos y funciones. (Anexo A “Diagrama de Clases”). Estos diagramas se generaron con la utilidad Visual Studio Paradigm For UML en su Versión de evaluación.

Figura 14. Freya



2.2.6 FUNCIÓN SKIPMOVIL

La función SkipMovil, se encarga de realizar un sensado constante de la posición del individuo y su cercanía del obstáculo móvil, implementando así un lazo cerrado con el sistema de visión y Freyja. Cuando el obstáculo móvil se encuentra a una distancia menor o igual a 10 centímetros del robot controlado se envía una señal vía radiofrecuencia al móvil para que detenga su movimiento y se procede a recalcular una trayectoria para la nueva disposición de obstáculos; una vez obtenida la trayectoria, se verifica nuevamente si el obstáculo se encuentra bloqueando la trayectoria del robot móvil, si es así envía la nueva trayectoria, de lo contrario ordena al móvil continuar con la trayectoria anterior. Cuando el robot móvil llega a su meta el sistema de visión lo identifica y detiene la rutina.

2.2.7 USO DE HILOS

Los hilos o “Threads” son un estilo de ejecución de rutinas donde se desliga el proceso en particular (una función, una variable, etc.) del programa principal, permitiendo un uso exclusivo de la memoria en un proceso sin alterar el funcionamiento del otro. En el caso de Freyja la generación de trayectorias y su evaluación es un proceso que requiere un uso computacional muy elevado, de tal forma que si se hubiese embebido el proceso de generación en la interfaz grafica, los resultados serian que mientras se estuviese generando una trayectoria, Freyja se bloquearía, provocando una caída de la aplicación por desbordamientos de memoria en trayectorias difíciles de generar.

Otra aplicación de los hilos en Freyja se dio en la función SkipMovil donde simultáneamente se llaman procesos para verificar posición de obstáculos y generación de trayectorias; a lo cual se le llama “multithreading“o “hilos paralelos”. Aunque realmente no se tienen 2 procesadores y solo se ejecuta un proceso a la vez, el multithreading da la apariencia de que se hiciese concurrentemente. Esto es posible utilizando el “scheduling”, el cual determina como y en que momento se

activan los hilos, de acuerdo a las prioridades que se le indiquen como argumentos.

2.2.8 CARACTERÍSTICAS DE ENTORNO

Freyja se destaca por su amabilidad hacia el usuario final, esto debido a características del programa que facilitan su operación, dentro de ellas destacamos.

- Las trayectorias y los obstáculos se grafican dentro de la grilla que muestra la ventana principal del sistema, otorgando una idea muy aproximada de la trayectoria que el sistema generó antes de que esta sea enviada por radiofrecuencia. Esto se logró utilizando un evento tipo “Paint” que se refrescaba cada vez que cambiaban ya fuese obstáculos o trayectorias generadas, la ubicación de obstáculos fijos en modo simulación también se logro combinando eventos del Mouse con el evento “Paint” general.
- Se pueden guardar las coordenadas de las trayectorias generadas como un documento de texto (.txt) haciendo más sencillo el análisis de trayectorias generadas.
- Identificación de la posición de los obstáculos ubicados ya sea por el archivo generado por el sistema de visión o por el usuario en modo simulación.
- Configuración del puerto serial desde Freyja.
- Calibración del robot móvil, desde una ventana de Freyja, buscando una mejor ejecución de las trayectorias.
- Acceso a ayudas desde el menú (ver Anexo E)

2.2.9 VALIDACIONES

Pensando en el uso de la aplicación por parte del usuario final se hizo un énfasis especial en las validaciones del ambiente de usuario y en las posibles excepciones que pudiese arrojar el sistema ante comportamientos inesperados y

variables mal dispuestas. El trabajo de debugging fue extenso y metódico siguiendo técnicas propuestas por los diversos artículos encontrados en la MSDN [8]. El uso de bloques Try-Catch para manejo de excepciones y el uso de eventos disparados por el teclado fueron fundamentales para una validación a conciencia del sistema.

3. APLICACIÓN: PLANEAMIENTO DE TRAYECTORIAS DE UN ROBOT MÓVIL

En este capitulo se describen las adaptaciones que tuvieron que realizarse en la plataforma de pruebas disponible (ver Anexo D), y se presentan las pruebas del desempeño del algoritmo junto a su respectivo análisis, validando así la herramienta software con una aplicación real.

3.1 ADAPTACIÓN PLATAFORMA DE PRUEBAS

3.1.1 MODIFICACIÓN PROGRAMA FÉNIX

FREYJA se apoya en el sistema FENIX desarrollado como una tesis de grado de la Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones de la Universidad industrial de Santander. FENIX permite una percepción del marco donde se desea planear la trayectoria del robot controlado mediante un sistema de visión, sin embargo el programa no almacena los resultados instantáneos de la posición de los obstáculos en el marco analizado. Debido a la necesidad del planeamiento online con herramientas independientes (FENIX para la visión y FREYJA para el control) se tuvo que desarrollar un componente adicional que hace que la herramienta de visión exporte los datos de posición espacial como un archivo binario .DAT que puede ser leído por FREYJA, esto cada vez que la cámara detecte un cambio en la posición de los obstáculos presentes en el marco de referencia. En el Anexo B se pueden apreciar los cambios sensibles que se realizaron sobre el código fuente de la herramienta.

3.1.2 COMUNICACIÓN ETHERNET

Debido al alto consumo de recurso computacional tanto de FREYJA como de FENIX, fue necesario dividir su funcionamiento, es decir tanto FENIX como FREYJA funcionan en un ordenador diferente. Por este motivo se debió implementar una red local entre los 2 sistemas, Para ello se escogió una configuración de red cruzada entre las tarjetas de red de ambos ordenadores, y en el módulo que se agregó a FENIX, se especificó una ubicación en la red para el sitio donde debía ser almacenado el archivo .dat.

3.1.3 COMUNICACIÓN POR RADIOFRECUENCIA

Se utilizaron los módulos RX-433 y TX-433 para establecer la comunicación por radiofrecuencia entre FREYJA y el robot móvil. Esta se hizo en AM a 433 MHZ. Para el transmisor se implementó una pequeño modulo con conexión al puerto serial, y el receptor se conectó a la tarjeta de control del robot móvil, la información acerca de estos módulos se encuentra en el ANEXO B de este libro. Desde FREYJA se puede configurar el puerto COM, y la velocidad en bits por segundo. Por defecto se utiliza un bit de parada en la configuración. Cabe recalcar que cada vez que se envía un camino resultante con FREYJA, también se envían datos correspondientes a la calibración del móvil que se utilizó en las pruebas de aplicación. (Ver Anexo C. “Hojas de datos módulos de Radiofrecuencia”).

3.2 PRUEBAS

Utilizando a FREYJA se generaron una serie de caminos junto con sus respectivas características (peso, individuo, razas, generaciones) buscando de esa forma encontrar las tendencias que describían el comportamiento del algoritmo genético y de su función de evaluación ante la variación de sus parámetros internos, igualmente se analizó la carga computacional que ejerce la solución sobre dos ordenadores de diferentes características buscando con esto determinar la eficiencia de la aplicación en función del tiempo.

3.2.1 VARIACIÓN DE PARÁMETROS

Las pruebas de variación de parámetros se realizaron teniendo en cuenta estudios anteriores en algoritmos genéticos, los cuales determinaron una serie de valores para los parámetros del algoritmo, considerados usuales [3] en el problema del planeamiento de la trayectoria de un robot móvil, estos valores son:

Tabla 3. Valores Nominales

Tasa de Cruce = 0.7
Tasa de Mutación = 0.001
Número de individuos = 50
Número de generaciones = 100
Peso Mínimo = 115

Se variaron los parámetros con respecto a estas condiciones iniciales, con dos arreglos de obstáculos diferentes, así como con dos computadores de diferentes características buscando ver la incidencia de cada parámetro sobre los resultados, Tales resultados son presentados en las tablas 5 a la 36.

En las pruebas se modificaron, utilizando valores extremos y medios, los parámetros: tasa de cruce, tasa de mutación, peso mínimo, número de generaciones, y Número de individuos. El análisis de los resultados comprende las variaciones arrojadas por el algoritmo en el número de raza, generación, individuo y peso del camino.

Las pruebas que se realizaron con valores extremos (tasa de mutación 1, tasa de cruce 1, tasa de mutación 0, tasa de cruce 0) buscan anular o maximizar el efecto de las operaciones genéticas cruce y mutación, esto con el fin de ver la importancia de estas operaciones sobre el algoritmo.

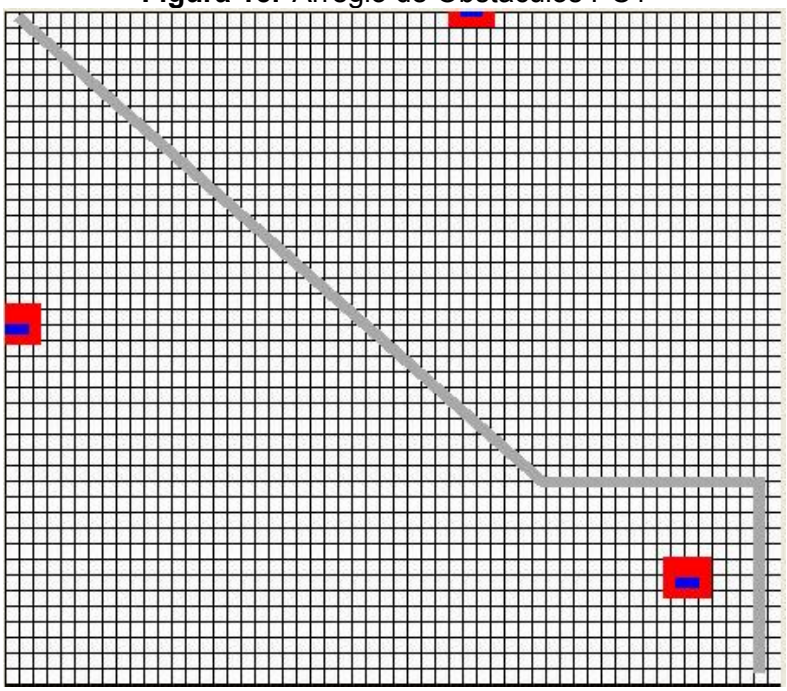
Se realizaron además las pruebas denominadas “especiales” donde utilizando un alto número de generaciones, se variaron los parámetros buscando un arreglo ideal de parámetros (mejor camino en función de su peso) para el algoritmo que se desarrolló en este proyecto.

Tabla 4. Descripción Computadoras

PC Número 1 → P4 2.4 GHZ, 512MB de RAM, Procesos Limpios en Ejecución
PC Número 2 → P4 1.8 GHZ, 256MB de RAM, Procesos Limpios en Ejecución

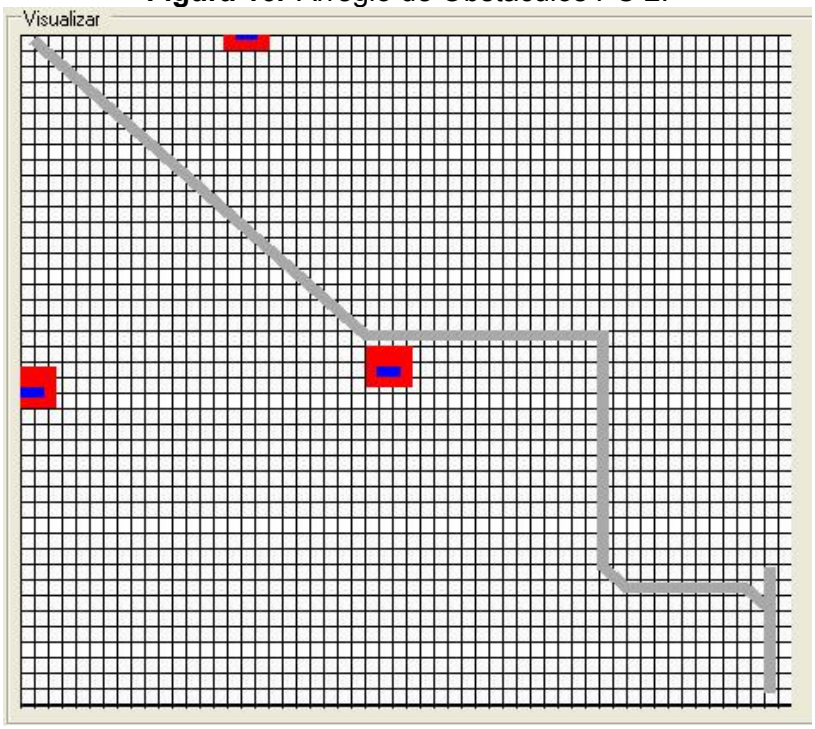
La siguiente distribución de obstáculos se utilizó para las pruebas en la PC 1.

Figura 15. Arreglo de Obstáculos PC1



La siguiente distribución de obstáculos se utilizo para las pruebas en la PC 2.

Figura 16. Arreglo de Obstáculos PC 2.



Condiciones Nominales

PC 1.

Tabla 5. Nominales PC 1

Peso del Camino	Número de razas	Posición del Individuo	Número de Generaciones
130.8874	10	1	0
130.6447	9	7	1
122.1595	1	45	0
115,89	5	10	4

PC 2.

Tabla 6. Nominales PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
115,2306	13	39	0
136,8579	7	49	62
123,0295	13	42	0
115,13	5	2	0

Peso mínimo del camino 100

PC 1.

Tabla 7. Peso Mínimo Camino 100 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
112.1595	13	16	0
117.2305	2	1	0

PC 2.

Tabla 8. Peso Mínimo Camino 100 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
129,2305	23	34	0
121,23	2	23	5

Peso mínimo del camino 130

PC 1.

Tabla 9. Peso Mínimo Camino 130 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
131.3796	3	37	1
130.6447	27	5	0

PC 2.

Tabla 10. Peso Mínimo Camino 130 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
130,1595	41	28	0
130,059	118	36	0

Para Número de Individuos 10

PC 1.

Tabla 11. Número de Individuos 10 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
131.4732	53	4	0
115.13	81	4	0

PC 2.

Tabla 12. Número de Individuos 10 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
130,059	118	3	0
122,5442	70	4	0

Para Número de Individuos 80

PC 1.

Tabla 13. Número de Individuos 80 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
135.9584	3	9	7
119.9584	2	64	0

PC 2.

Tabla 14. Número de Individuos 80 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
127,3726	8	54	0
125,3726	4	62	0

Para generaciones 40

PC 1.

Tabla 15. Número de Generaciones 40 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
126.1595	6	29	5
124.3016	2	36	0

PC 2.

Tabla 16. Número de Generaciones 40 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
128,5422	10	7	5
127,13	17	48	4

Para generación 160

PC 1.

Tabla 17. Número de Generaciones 160 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
135.059	14	3	0
126.7868	1	23	4

PC 2.

Tabla 18. Número de Generaciones 160 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
120,1595	14	41	0
116,95	1	25	0

Para Número de tasa de cruce 0

PC 1.

Tabla 19. Tasa de cruce 0 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
130.059	9	1	0
128.1595	15	4	0

PC 2.

Tabla 20. Tasa de cruce 0 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
126,059	7	44	0
132,7868	18	15	0

Para tasa de cruce 0.5

PC 1.

Tabla 21. Tasa de cruce 0.5 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
120.4437	7	26	0
129.8874	40	49	7

PC 2.

Tabla 22. Tasa de cruce 0.5 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
125,0295	16	42	7
126,4437	17	47	1

Para tasa de cruce 1

PC 1.

Tabla 23. Tasa de cruce 1 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
134.7868	1	49	17
129.2305	1	47	8

PC 2.

Tabla 24. Tasa de cruce 1 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
132,7868	18	15	0
125,8163	6	35	1

Para tasa de mutación 0

PC 1.

Tabla 25. Tasa de mutación 0 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
129.2305	2	20	2
121.13	1	22	1

PC 2.

Tabla 26. Tasa de mutación 0 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
121,7868	4	20	10
134,8874	4	49	6

Para tasa de mutación 0.5

PC 1.

Tabla 27. Tasa de mutación 0.5 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
116.059	21	3	27
130.059	23	10	0

PC 2.

Tabla 28. Tasa de mutación 0.5 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
116,6448	6	47	1
130,6447	1	1	0

Para tasa de mutación 1

PC 1.

Tabla 29. Tasa de mutación 1 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
120.059	2	30	0
131.2305	1	40	0

PC 2.

Tabla 30. Tasa de mutación 1 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
138,7868	18	49	32
124,8874	14	24	0

Para tasa de cruce 0 y mutación 1

PC 1.

Tabla 31. Tasa de cruce 0 y Tasa de mutación 1 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
117.7158	26	16	0
124	3	33	0

PC 2.

Tabla 32. Tasa de cruce 0 y Tasa de mutación 1 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
116.7868	5	25	0
122.059	4	8	0

Para tasa de cruce 1 y mutación 0

PC 1.

Tabla 33. Tasa de cruce 1 y Tasa de mutación 0 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
123.5147	3	25	1
128.1595	1	4	13

PC 2.

Tabla 34. Tasa de mutación 1 y Tasa de mutación 0 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
120,2011	2	18	0
120,6863	1	32	2

Para tasa de cruce 1 y mutación 1

PC 1.

Tabla 35. Tasa de cruce 1 y Tasa de mutación 1 PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
137.6447	35	49	50
131.2305	11	30	2

PC 2.

Tabla 36. Tasa de cruce 1 y Tasa de mutación 1 PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones
125,13	4	15	0
124,4437	7	4	1

Altas Generaciones con variación de Parámetros

PC 1.

Tabla 37. Variación de parámetros con altas generaciones PC 1

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones	Cambio Realizado
137,2721	45	49	5892	TC = 1 Peso=130
135,2305	3	49	4362	TC=0.5 TM=0.5
143,6153	2	49	3025	TC=0.9 TM=0.5
145,3726	11	99	6065	TC=0.9 TM=0.5 IND = 100
153,8163	128	49	78	TC=0.9 TM=0.35 GEN=1000 PESO =145

PC 2.

Tabla 38. Variación de parámetros con altas generaciones PC 2

Peso del Camino	Número de Razas	Posición del Individuo	Número de Generaciones	Cambio Realizado
132,5442	14	49	6674	TC = 1 Peso=130
141,9584	1	49	8029	TC=0.5 TM=0.5
141,0295	1	49	7284	TC=0.9 TM=0.5
140,4437	2	99	5801	TC=0.9 TM=0.5 IND = 100
147,5737	324	20	9	TC=0.9 TM=0.35 GEN=1000 PESO =145

3.2.2 TIEMPO DE COMPUTO

Esta prueba se realizó con el propósito de observar el tiempo que le toma al software FREYJA encontrar una trayectoria en función del número de generaciones que ejecuta para hallarla. Para ello se tomó una muestra de 12 ejecuciones del software en los computadores descritos en la tabla numero 4 denominada “Descripción Computadoras”. Los tiempos se tomaron cronometrando con el “profiler” ANTS para C# cada uno de los caminos que se generaron.

Tabla 39. Tiempos computacionales PC 1 Y PC2

PC1		PC2	
Tiempo Por Generación en segundos.	Tiempo Total Para obtener Una Respuesta Optima en segundos.	Tiempo Por Generación en segundos.	Tiempo Total Para obtener Una Respuesta Optima en segundos.
0.0054	9.2	0.0133	9.35
0.0071	4.3	0.0117	68.06
0.0095	2	0.011	1.22
0.0069	4.9	0.0119	13.52
0.0067	4.7	0.0144	17.4
0.0058	12.2	0.0246	2.49
0.0063	5.1	0.0181	3.65
0.0078	0.8	0.01199	21.69
0.0076	4.6	0.01185	33.2
0.0063	9.6	0.12	20.54
0.0065	7.9	0.01161	55.21
0.0062	11.2	0.012	41.05

Figura 17. Eficiencia Computacional PC 1.
Número de Generaciones Contra Tiempo en segundos

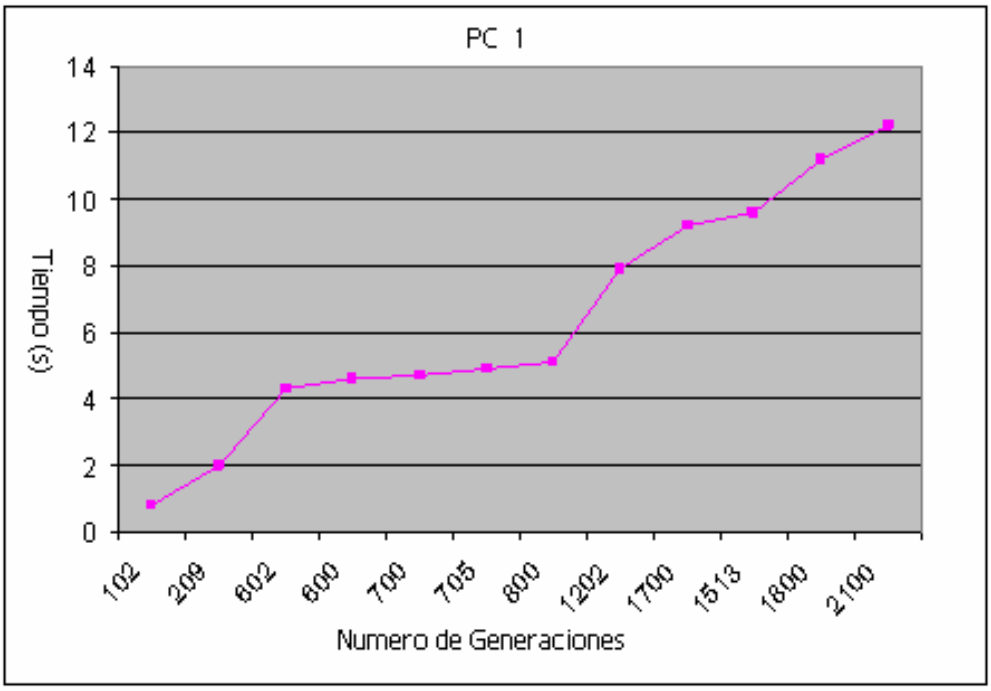


Figura 18. Eficiencia Computacional PC 2.
Número de Generaciones Contra Tiempo en segundos

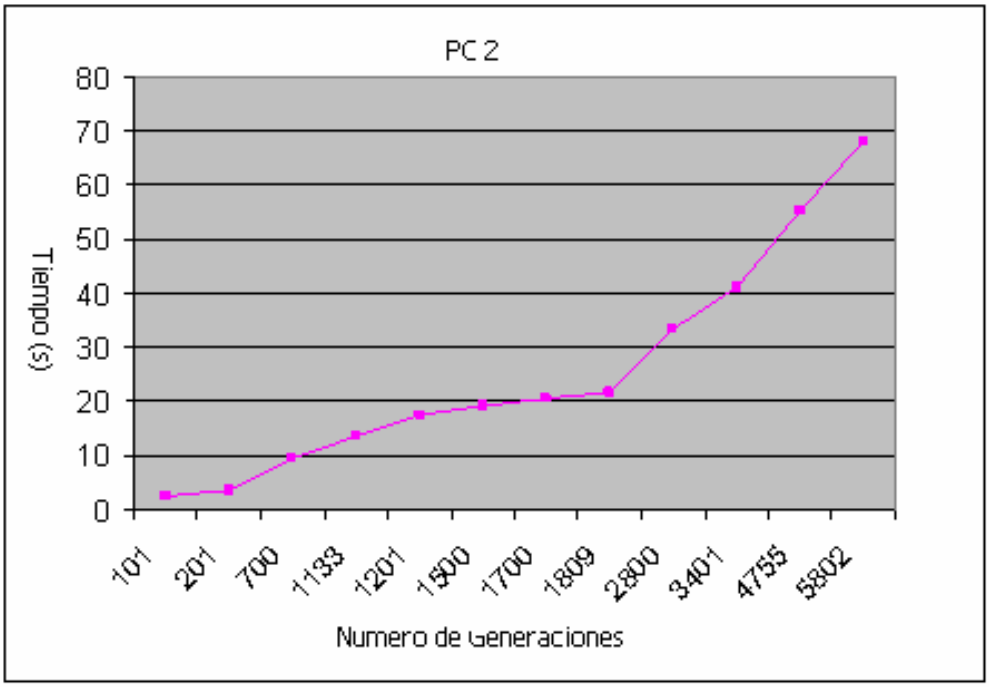


Tabla 40. Medias Tiempo Computacional

Computador	Tiempo Medio por Generación en segundos	Tiempo Total Medio Para obtener una respuesta optima en segundos
PC Número 1	0.0068	6,375
PC Número 2	0.0147	25,55

3.2.3 ANÁLISIS DE RESULTADOS.

Los algoritmos genéticos han sido analizados bajo la teoría de esquemas [1]. Estos estudios han concluido que si bien los algoritmos genéticos no pueden ser modelados en su totalidad, si pueden ser guiados a obtener cierto tipo de respuestas como resultado de la manipulación de uno o varios de sus parámetros.

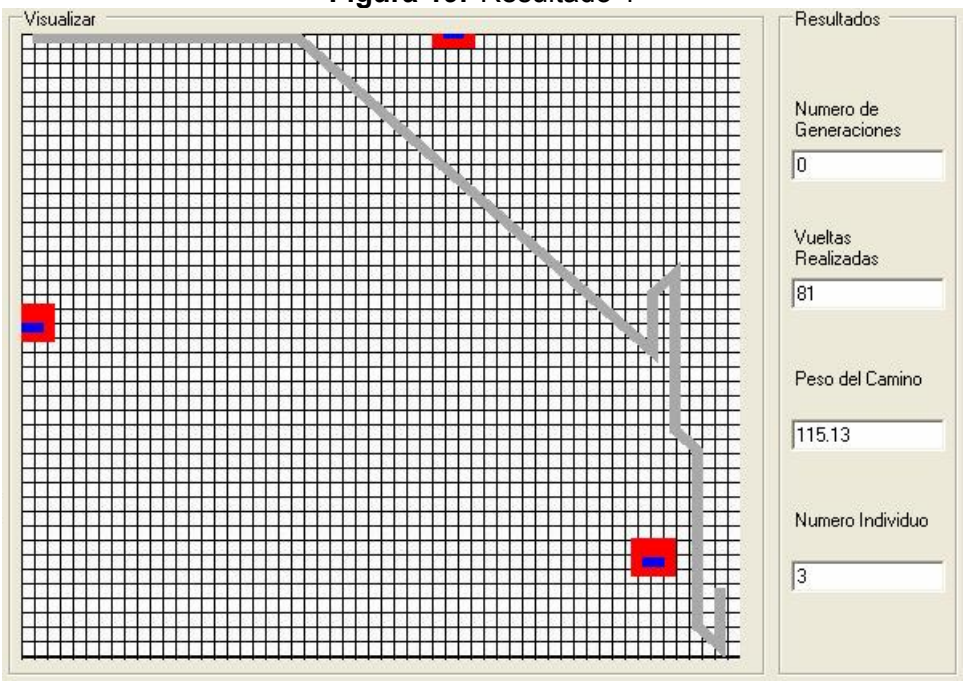
De los resultados que este proyecto obtuvo se evidencian tendencias particulares que son presentadas a continuación:

Necesidad de la técnica de extinción a un bajo Número de generaciones y de individuos.

El algoritmo genético implementado parte de una población totalmente aleatoria, por lo consiguiente es usual que la totalidad de los individuos generados en una población no sean aptos para sobrevivir. Cuando se cruzan estos individuos con un ADN defectuoso, las probabilidades de obtener un individuo solución son muy bajas, esto se vislumbra en las tablas a condiciones nominales con un bajo número de individuos y bajo Número de generaciones; cuando se encuentran soluciones en estos casos, usualmente son producto de la generación aleatoria de

nuevos individuos capaces de cumplir las condiciones mínimas del problema, estos nuevos individuos son producto de la técnica de extinción que acaba con una raza (población) y crea una nueva.

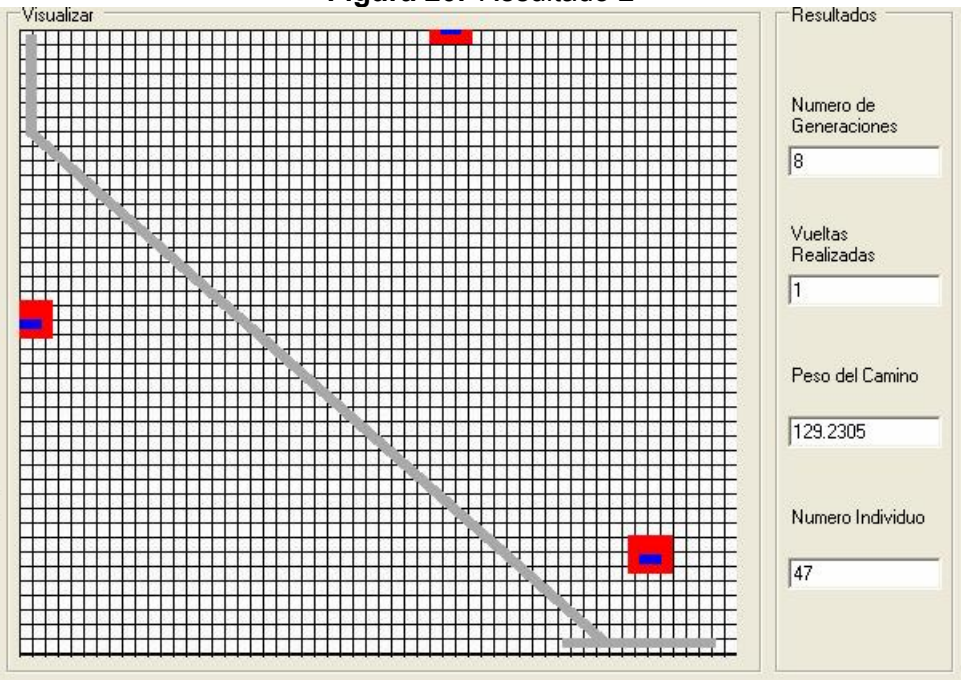
Figura 19. Resultado 1



Efectos de la operación genética cruce

Al observar los resultados de las pruebas realizadas con el parámetro “Tasa de Cruce” con valor nulo o muy pequeño, se evidencia que al no existir cruce, el peso de la solución recae en otras operaciones genéticas, provocando una demora en la obtención de un individuo apto. En este caso se considera que esto no es un algoritmo genético real [1], ya que no esta efectuando un cruce sexual entre los individuos de su población. De esta forma la respuesta es consecuencia de la aleatoriedad a la hora de generar nuevos individuos o de la mutación de individuos existentes.

Figura 20. Resultado 2



Determinación del peso mínimo considerando el coste computacional.

Lo óptimo de una trayectoria depende de su peso. Este determina la distancia que tiene que recorrer la misma para llegar a la meta. Sin embargo un peso muy alto usualmente se traduce en un consumo de recursos computacionales mayor, lo cual sucede porque al aumentar el peso mínimo se disminuye la probabilidad de que en los cruces o en las generaciones aleatorias aparezca un camino que cumpla con los requisitos establecidos, razón por la cual el peso ideal en un algoritmo genético debe ser un equilibrio entre la excelencia del camino y los recursos computacionales existentes.

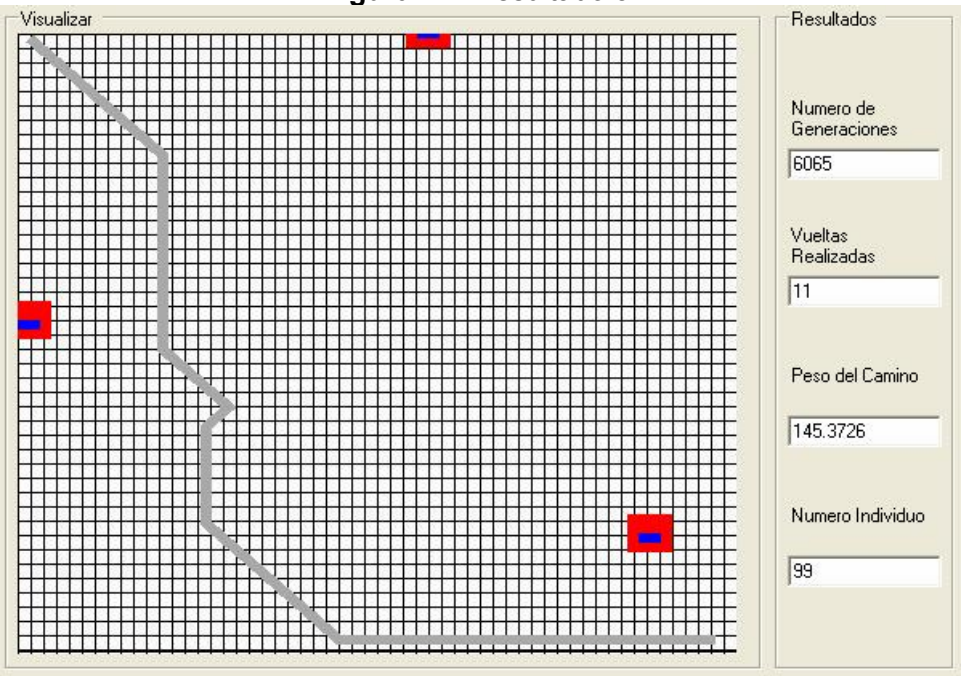
En las figuras 17 y 18 se puede observar como una diferencia de 600 MHz en la capacidad de procesamiento entre 2 computadores proporciona una velocidad significativamente superior para el PC 1. Un algoritmo genético ejecutado en un

cluster (computación paralela) daría respuestas de alta calidad y no existiría real preocupación por el recurso computacional.

Importancia de la correcta selección de la tasa de mutación.

Muchos autores tratan a la tasa de mutación como un parámetro sin importancia que se incluye para darle rigurosidad al nombre “genético”. Esto sucede porque ellos toman el algoritmo genético como un problema matemático y de algoritmia en lugar de un problema computacional. Al abordarlo desde la perspectiva computacional se observa que la tasa de mutación es la manera perfecta de evadir el problema de la dificultad de generar una semilla estrictamente aleatoria. En consecuencia la incapacidad del lenguaje computacional de compilar un número aleatorio realmente válido se vería disminuida debido a los efectos cambiantes sobre los individuos de la mutación, esta le aporta una mejora a la aleatoriedad que el computador no puede emular totalmente.

Figura 21. Resultado 3.



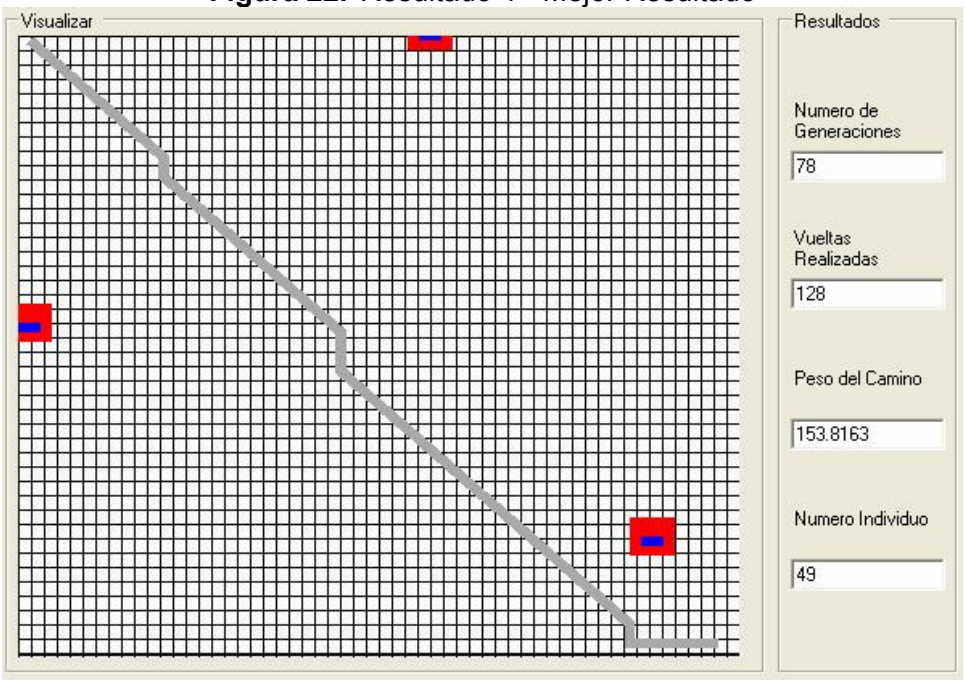
Obtención de una trayectoria óptima y la eficiencia del sobreviviente.

En la búsqueda de un peso excelente (superior a 140 en los arreglos de obstáculos planteados) se buscó un adecuado surtido de parámetros que llevó a una combinación basada en un alto número de generaciones, un alto valor en la tasa de cruce y un valor mediano en la tasa de mutación. Se puede apreciar como la mayoría de los individuos vencedores son los que se almacenan en la última posición del vector población. Un individuo de estas características es raza tras raza el que corresponde al "*Sobreviviente*" y se propaga generación tras generación cruzándose de tal manera que se convierte en el individuo más apto de toda la población.

Los resultados de las pruebas demuestran que aunque el número de generaciones sea muy alto, no necesariamente la población inicial generada puede llegar a cruzarse y a "*procrear"* un individuo capaz de satisfacer los requerimientos del problema planteado, de ahí el que se generen nuevas razas de individuos. Sin embargo cuando se cuenta con una buena raza un alto número de generaciones asegura un cruce suficiente como para generar un individuo con un ADN excelente para solucionar el problema.

La tabla denominada "Altas Generaciones con Variación de Parámetros" del numeral 5.1 demuestra como una buena selección de parámetros se ve representada en una trayectoria de calidad aunque sacrificando tiempo computacional.

Figura 22. Resultado 4 - Mejor Resultado



4. CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES.

- El resultado final del proyecto es la integración funcional del sistema de visión “FENIX” con el software de control basado en algoritmos genéticos “FREYJA” y con el periférico de comunicaciones basado en radiofrecuencia que emite las ordenes a cumplir por el robot móvil. El algoritmo implementado en FREYJA cumple con generar una trayectoria capaz de eludir dos obstáculos fijos y un obstáculo móvil, el robot móvil será capaz de ejecutarla siempre y cuando se cumplan las especificaciones del numeral 3.4 (Requerimientos).

- FREYJA permite visualizar y mejorar la comprensión de la técnica de inteligencia artificial “Algoritmos Genéticos” debido a sus opciones de simulación o tiempo real además de la posibilidad de variar los parámetros fundamentales del algoritmo genético. Este bagaje académico facilita posteriores investigaciones en el área de la computación evolutiva y las técnicas de inteligencia artificial.

- Al algoritmo genético simple se le incorporaron dos nuevas operaciones genéticas desarrolladas a partir de la técnica *Elitismo,* las cuales además de representar una innovación en el área facilitan la obtención de trayectorias en entornos sin suficientes recursos computacionales.

- FREYJA es una herramienta desarrollada en un lenguaje de programación de alto nivel y de estándares actuales como lo es C#. Esto permite una eficiencia en su ejecución además de un orden implícito que facilita su programación. El uso de un lenguaje popular de alto nivel diferente a los convencionales en inteligencia artificial (LISP, PROLOG, etc...) facilita su lectura y posterior modificación motivando así la evolución de la herramienta.

- El uso de hilos en un software que intenta consumir el total de los recursos computacionales de un sistema mantiene la estabilidad de ejecución en la aplicación.

- A pesar de la evidente aleatoriedad de los algoritmos genéticos se logró encontrar una serie de valores de partida en los parámetros para una óptima ejecución del algoritmo, la combinación apropiada de estos en conjunto con la disponibilidad de recursos computacionales soluciona de manera adecuada el problema del planeamiento de trayectorias en terrenos bidimensionales.

- Los algoritmos genéticos pueden ser aplicados en infinidad de problemas. La metodología que se utilizó para resolver el problema puntual del planeamiento de trayectorias puede ser igualmente aplicada en otros problemas.

- La modularidad del sistema facilita su posterior reusabilidad, debido a que al ser objetos independientes los bloques de operaciones genéticas y de la función de evaluación, pueden usarse como elementos de otro sistema que maneje una distinta codificación y que tenga otro propósito.

- Los sistemas de cómputo a diferencia del ser humano no pueden generar un número al azar, ellos se basan en *semillas* que proporcionan una base matemática para crear el número aleatorio. Usualmente estas semillas son valores como el último microsegundo del reloj. En una técnica iterativa como lo es el algoritmo genético la seudo-aleatoriedad se traduce en predictibilidad del algoritmo y por consiguiente se rompe con el principio fundamental de la técnica, de ahí que a la hora de programar un algoritmo genético se debe tener particular cuidado en la forma como se escogen las semillas y en la manera como se usan. La tasa de mutación ayuda a mitigar este inconveniente.

4.2 RECOMENDACIONES

- Las interfaces de comunicación entre el sistema de control y el robot móvil deben vigilarse y procurar mantenerse libres de ruido. Agentes como controles tipo PWM pueden interferir en las comunicaciones por radiofrecuencia limitando así el desempeño del robot.

- El algoritmo genético adquiere su máxima utilidad cuando funciona sobre sistemas de cómputo en paralelo; podría entonces implementarse este tipo de técnicas en un cluster de ordenadores.

- Los algoritmos genéticos deben ser implementados en un entorno adecuado, un entorno demasiado cambiante exige demasiado al algoritmo y no le permite ejercer su mejor labor como algoritmo de optimización, en ese caso aún se pueden aprovechar las bondades de los algoritmos genéticos implementándolos junto con otras técnicas de inteligencia artificial, estos modelos híbridos son más potentes para ciertas aplicaciones. Por ejemplo un sistema de lógica difusa donde sus reglas de inferencia sean determinados por un algoritmo genético.

- La programación genética como evolución de los algoritmos genéticos puede ser implementada para resolver cualquier tipo de problema de una manera mucho más eficiente y con una intromisión humana aún menor que en los algoritmos genéticos.

5. REFERENCIAS BIBLIOGRÁFICAS

[1]. HOLLAND, John H. Adaptation in Natural and Artificial Systems, University of Michigan Press, 1975.

[2]. KOZA, John R. Genetic Programming. On the Programming of Computers by Means of Natural Selection, The MIT Press, 1992.

[3]. SUGIHARA, Kazuo; SMITH Jhon; Genetic Algorithms for Adaptive Planning of Path and Trajectory Of a Mobile Robot in 2D terrains.

[4].COELLO COELLO, Carlos A; "Introducción a los Algoritmos Genéticos", Soluciones Avanzadas. Tecnología de Información y estrategias de Negocios, Año 3, No 17, Enero de 1995, pp5-11.

[5].OLLERO BATURONE, Aníbal; Robótica: Manipuladores y Robots móviles. España. Alfaomega Marcombo S.A.

[6]. GIAMARCHI, Frédéric; Robots Móviles: Estudio y Construcción. Thomson Editores Paraninfo S.A. España 2001.

[8]. Microsoft; MSDN: Microsoft Source Developer Network, 2003

[9] TIBADUIZA, Diego Alexander, Planeamiento de Trayectorias en Robot Móviles, Tesis de Maestría Universidad Industrial de Santander, 2005.

[10]. Federación internacional de fútbol de robots. Disponible en http://www.fira.net/about/overview.html [citado en junio 2005]

[11] NAVAS, Oscar; ORTIZ, Nikolai; Algoritmos genéticos aplicados al planeamiento de trayectorias de un robot móvil, ISBN 958-8028-49-3, Colciencias 2005.

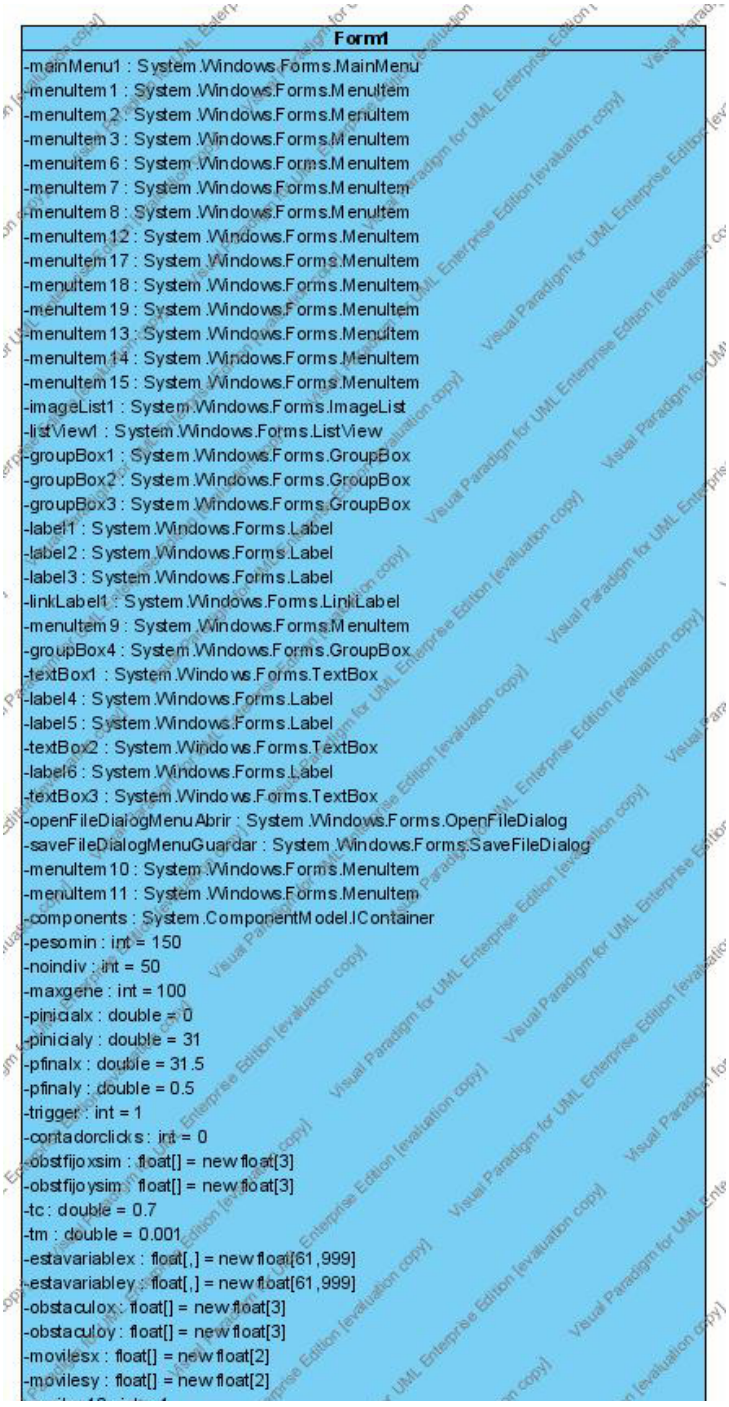
[12] AMAYA, Yamit; RUIZ, Jhon; Localización dinámica de móviles y obstáculos en una escena controlada para aplicaciones en robótica; tesis de pregrado Universidad Industrial de Santander; 2005.

[13]ALBARRACIN, Carlos; MENDOZA, Edward; Control de dos móviles en un entorno dinámico; tesis de pregrado Universidad Industrial de Santander; 2005.

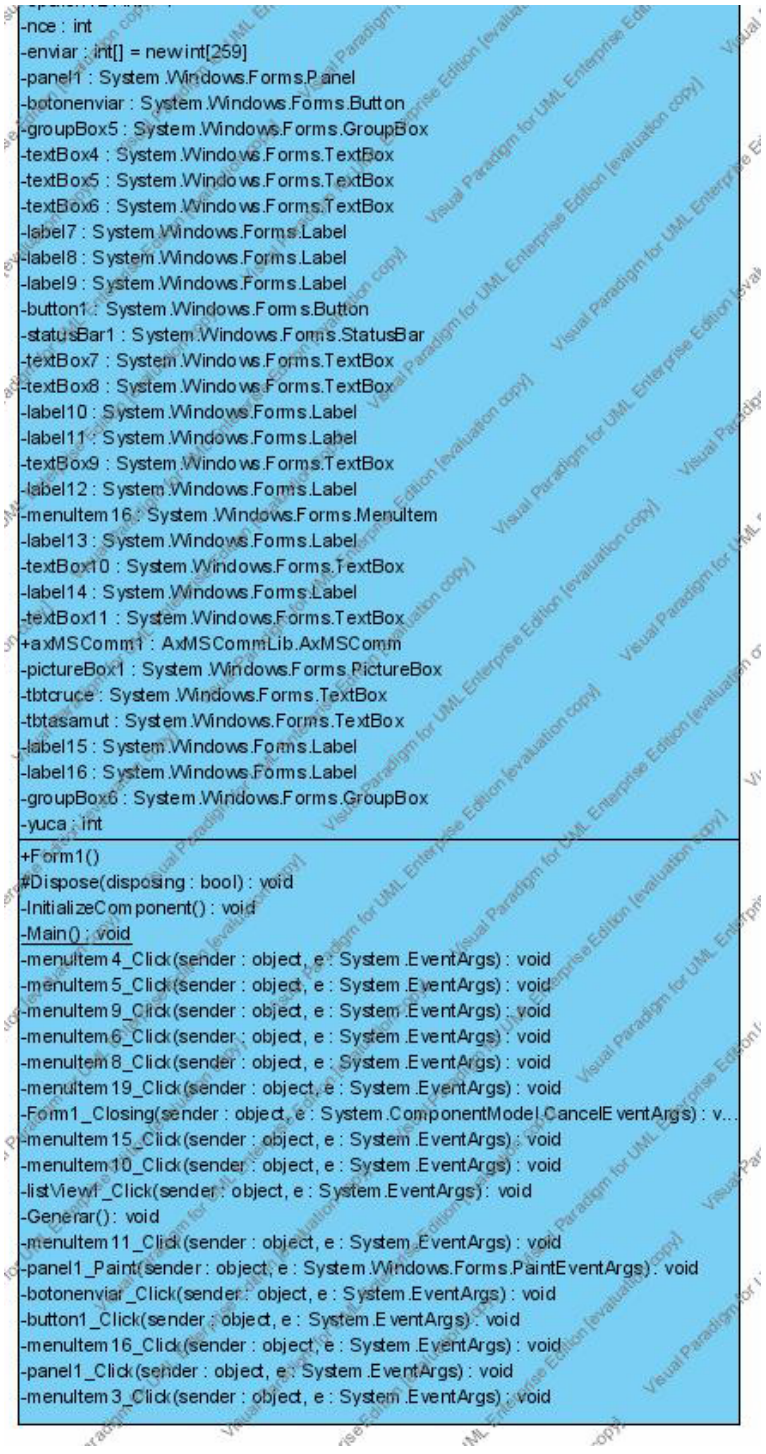
[14]LARRAZABAL, Germán; Paralelización de métodos iterativos; Revista Ingenierías Universidad de Carabobo; Vol. 10; Número 2; Pág. 55-59; 2003.

ANEXO A.

DIAGRAMA DE CLASES

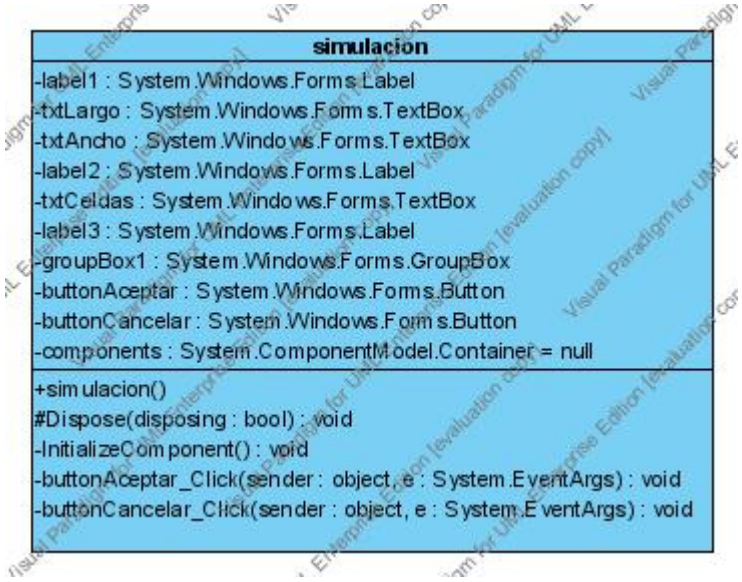


A.1. Clase Form1 Primera Parte

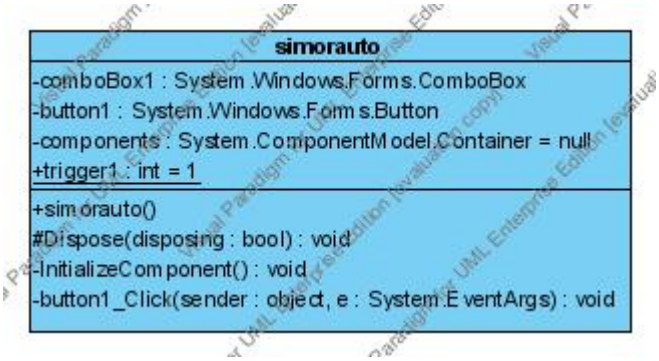


A.2. Clase Form1 Segunda Parte

A.3. Clase Genética

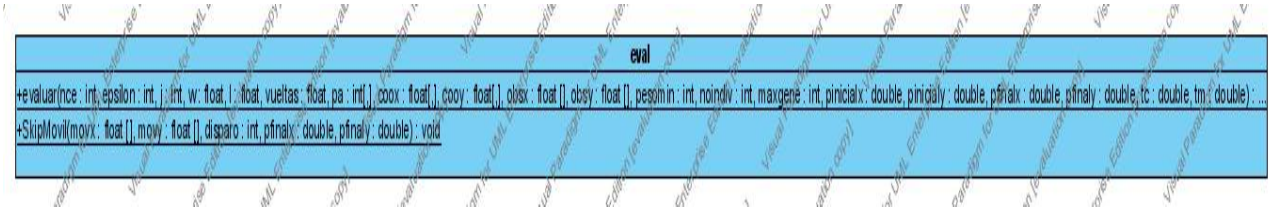


A.4. Clase Simulación

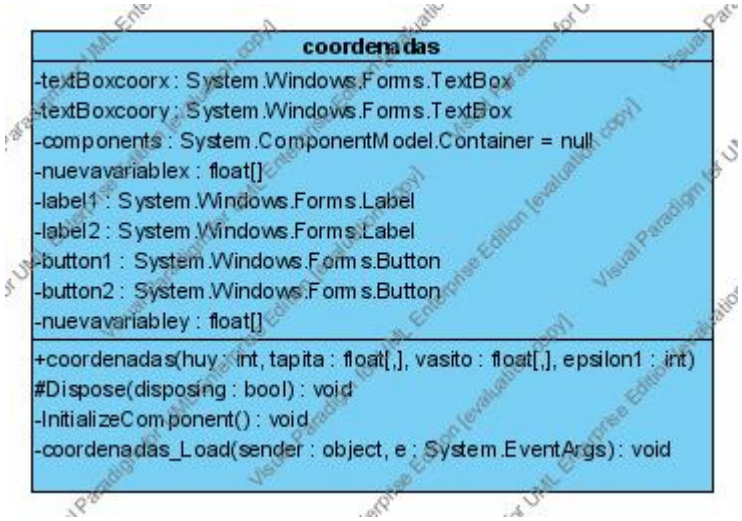


A.5. Clase Simorauto

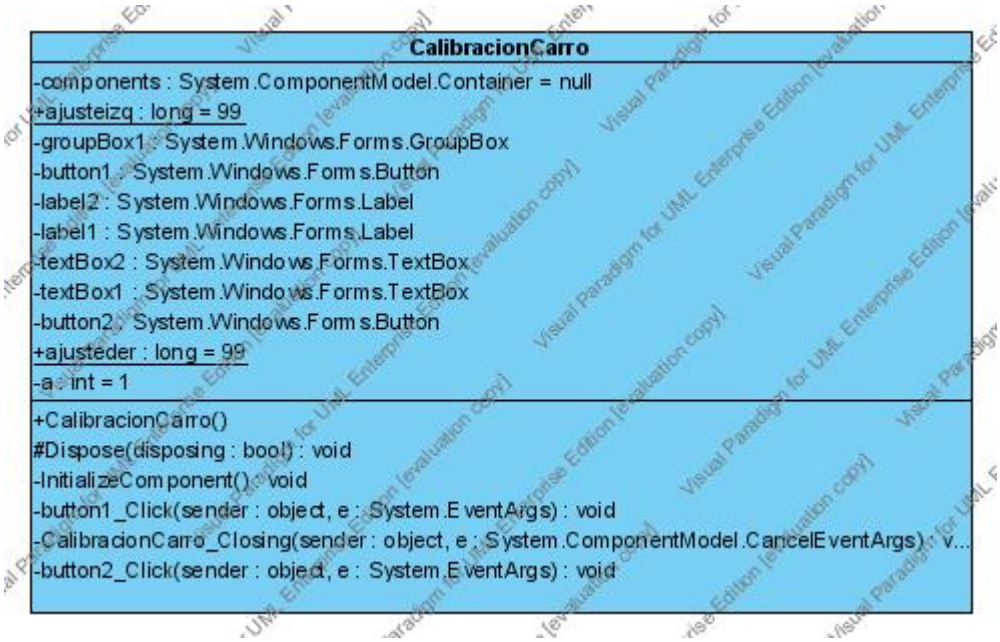
A.5. Clase Auto



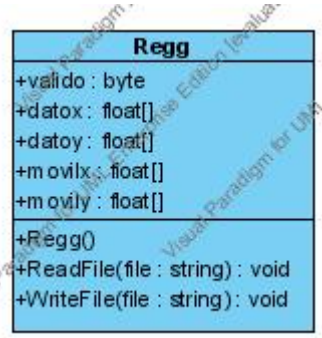
A.6. Clase Eval



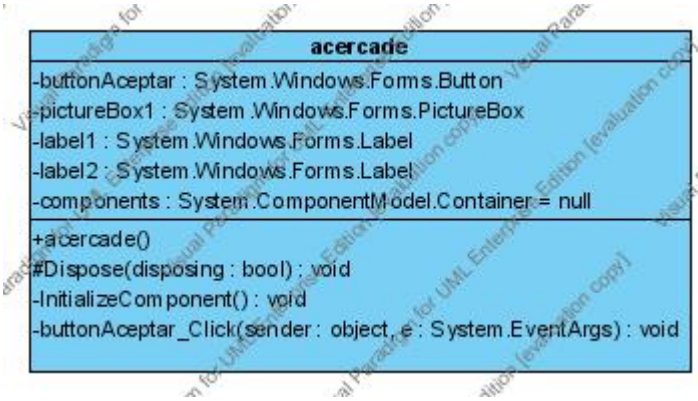
A.7. Clase Coordenadas



A.8. Clase CalibracionCarro



A.9. Clase Regg



A.10. Acercade

ANEXO B.
MODIFICACIÓN FENIX

En el presente anexo se muestra las modificaciones hechas al software de visión FENIX.

- Clase “obstaculo”:

En esta clase se definen las funciones que permiten agregar los valores a las variables datox, datoy, movilx y movily. Estas variables pertenecen al registro que se graba en el archivo obta.dat

La clase se encuentra en el archivo obstaculos.h

```
#include <iostream>
#include <fstream>
#include <iomanip>
#include <cstdlib>
#include <cstring>
using namespace std;

class obstaculo
{
public:
    obstaculo(char *n=NULL, float d1=0) :
      valido('S')
      {
              int i;
              for (i=0;i<=4;i++){
                      datox[i] = d1;
                      datoy[i] = d1;
              }
      movilx[1]=0;
      movilx[2]=0;
      movily[1]=0;
      movily[2]=0;
      }
      float copiar(float coox[5],float cooy[5]);
      float ingresar(int ix, double x, double y);
      float ingresar2(int ix, double x, double y);
```

```
      const bool Valido() { return valido == 'S'; }
private:
    char valido; // Campo que indica si el registro es válido
    // S->Válido, N->Inválido
    float datox[5];
    float datoy[5];
    float movilx[2];
    float movily[2];
};

float obstaculo::ingresar(int ix, double x, double y)
{
    datox[ix]=x;
    datoy[ix]=y;
    return 0;
}


float obstaculo::ingresar2(int ix, double x, double y)
{
    movilx[ix]=x;
    movily[ix]=y;
    return 0;
}

obstaculo regg;
```

- Copiar obstáculos fijos al archivo obta.dat:

Esta función se agregó en el archivo FENIXDoc.cpp, la función se encarga de copiar las coordenadas de los obstáculos fijos, generadas por FENIX, en el registro creado en la clase obstáculo, luego procede a escribir el registro en el archivo obta.dat

```
    static int ix = -1;
++ix;

    regg.ingresar(ix, px[0], py[0]);
    if (ix > 3)
    {

    FILE *of = fopen("N:/ag2/obta.dat", "w+b");
    fwrite((void *)&regg, sizeof(obstaculo), 1, of);
```

```
    fclose(of);


ix = -1;

    }
```

- Copiar móviles al archivo obta.dat:

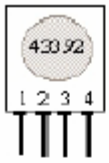
Esta función se agrego en el archivo FENIXDoc.cpp, la función se encarga de copiar las coordenadas de los robots móviles, generadas por FENIX, en el registro creado en la clase obstáculo, luego procede a escribir el registro en el archivo obta.dat.

```
    static int ix = -1;
++ix;

    regg.ingresar2(ix, movilx[0], movily[0]);
    if (ix > 2)
    {

    FILE *of = fopen("N:/ag2/obta.dat", "w+b");
    fwrite((void *)&regg, sizeof(obstaculo), 1, of);
    fclose(of);
ix = -1;
```

ANEXO C.

HOJAS DE DATOS RADIOFRECUENCIA

TWS-434A RF Transmitter

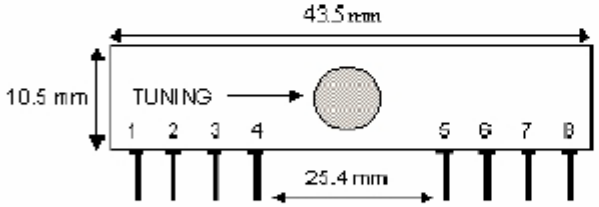


Pin - 1 Gnd
Pin - 2 Data Input
Pin - 3 Vcc
Pin - 4 RF Output

Frequency: 433.92MHz
Modulation: AM
Operating Voltage: 2 - 12 VDC

Symbol	Parameter	Condition	Min	Typ	Max	Unit
Vcc	Supply Voltage		2.0	-	12.0	V
Ip	Peak Current	2V / 12V	-	1.64 / 19.4	-	mA
Vh	Input High Voltage	Idata = 100uA (High)	Vcc-0.5	Vcc	Vcc+0.5	V
Vl	Input Low Voltage	Idata = 0 uA (Low)	-	-	0.3	V
Fo	Operating Frequency		433.90	433.92	433.94	MHz
Tr / Tf	Modulation Rise / Fall Time	External Coding	-	-	100 / 100	uS
Po	RF Output Power – Into 50Ω	Vcc = 9 to 12 V Vcc = 5 to 6V	-	16 14	-	dBm
Dr	Data Rate	External Coding	-	2.4K	3K	Bps

RWS-434 RF Receiver



Pin—1 Gnd
Pin—2 Digital Data Output
Pin—3 Linear Output
Pin—4 Vcc
Pin—5 Vcc
Pin—6 Gnd
Pin—7 Gnd
Pin—8 Antenna (approx 30 – 35 cm)

Frequency: 433.92MHz
Modulation: AM
Operating Voltage: 4.5 - 5.5 VDC
Output: Digital & Linear

Symbol	Parameter	Condition	Min	Typ	Max	Unit
Vcc	Supply Voltage		4.5	5	5.5	V
It	Operating Current		-	3.5	4.5	mA
	Channel Width	+ / - 500				kHz
Rd	Data Rate				3k	Bps
Vdat	Data Out	Idata = +200 uA (High)	Vcc-0.5	-	Vcc	V
		Idata = -10 uA (Low)	-	-	0.3	V

ANEXO D.

DESCRIPCIÓN DEL ENTORNO

La sección de robótica del grupo de investigación CEMOS (control, electrónica, modelado y simulación) adelanta un proyecto destinado a crear las herramientas para poner en funcionamiento un grupo de robots interactuantes. Para este propósito se ha desarrollado una serie de proyectos de maestría y pregrado dentro de los cuales se cuenta el presente proyecto. Estos proyectos incluyen el desarrollo de un software de visión que permite identificar el entorno y la posición del robot en su interior. Un software para el planeamiento de la trayectoria y el hardware necesario para transmitir a los robots las instrucciones derivadas del software de planeamiento.

Para realizar el planeamiento se implementaron dos estrategias diferentes. En una primera aproximación se utilizó la metodología de los campos de potencial y posteriormente se implementó la estrategia basada en algoritmos genéticos, cuyo desarrollo es el objeto fundamental del presente proyecto.

Para la realización de las pruebas se utilizó un grupo de robots didácticos, propiedad de la escuela, los cuales se desplazan sobre una plataforma de pruebas. Los elementos que conforman la plataforma se describen a continuación:

PISTA

La pista construida es de madera con dimensiones 2.44 m X 1.52 m y marcos delimitantes, también en madera ubicados en el borde de la pista (para indicarle al robot con ultrasonidos la existencia de un limite), su color negro opaco destaca los elementos que se encuentran en escena dándole así más confiabilidad al sistema de visión.

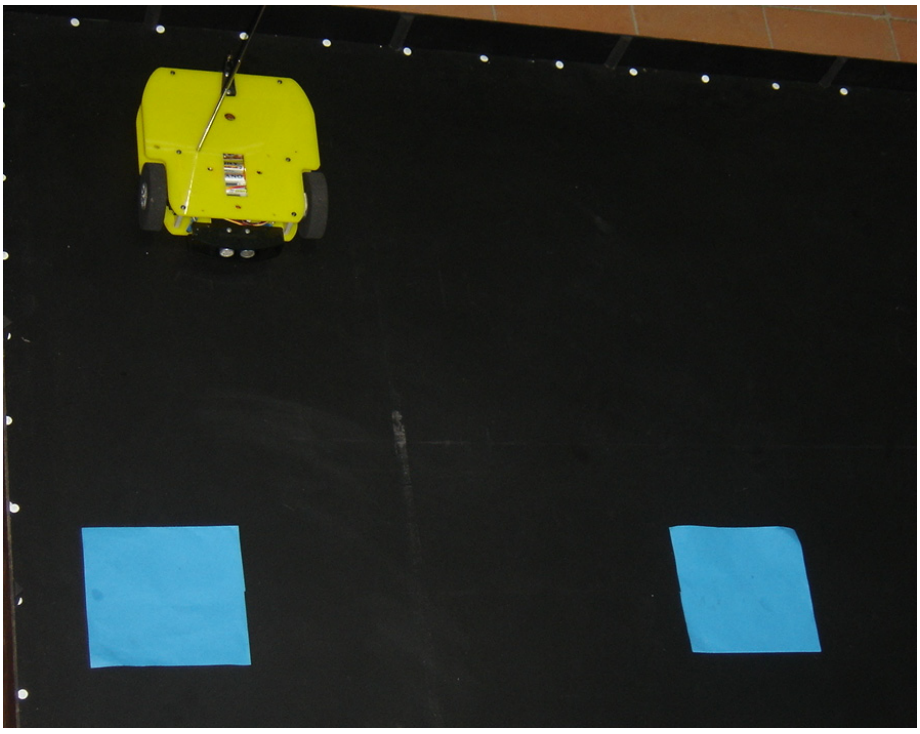
ROBOTS MÓVILES

Dos robots móviles fabricados en acrílico de color amarillo (para destacarse en el sistema de visón). Cada uno con una rueda loca y dos ruedas de tracción impulsadas independientemente por un servomotor. Su fuente de alimentación son 6 baterías de 1.2 V y 2100 mA/h que proporcionan potencia a los servomotores, a la tarjeta de control, y a los ultrasonidos que les sirven de sensores.

El robot móvil encargado de ejecutar la trayectoria planeada en la pista, posee además un modulo receptor de radiofrecuencia y una tarjeta de control diferente a la del robot que se desplaza aleatoriamente para hacer las veces de obstáculo móvil. El sistema de cómputo de ambos robots se basa en el Microcontrolador MOTOROLA MC68HC908GP32 programado de acuerdo a la función específica de cada robot.

OBSTÁCULOS

Se utilizaron como obstáculos cubos de cartón de color azul opaco para evitar la reflexión de la luz y de dimensiones 0.15m X 0.15m, características suficientes para poder ser identificados correctamente por el sistema de visión.



Pista, Obstáculos y Robot

SISTEMA DE VISIÓN

El sistema de visión consta de una cámara analógica DFK50H13/N ubicada en el laboratorio de robótica, equipada con un lente de 4 mm y salida de video compuesto. Una tarjeta digitalizadora para la compresión de video tipo PCI DFG/Compress de 32 bits con entrada y salida de video compuesto instalada en un ordenador con procesador PENTIUM IV a 1.5 GHz y 256 MB de RAM el cual también aloja el software FENIX encargado del procesamiento de las imágenes capturadas por la cámara.

El software FENIX muestra en pantalla las coordenadas bidimensionales que representan la ubicación de obstáculos y robots en la pista donde se encuentran, identificando como obstáculo fijo el color azul y como robot móvil el color amarillo, y usando modelos de visión XYY y HLS

Cámara Fenix

Captura y digitalización de la imagen, Fuente [9]

HOST DE CONTROL

Un computador con un procesador PENTIUM IV y 256 MB de RAM es el encargado de ejecutar el software de control que planea las trayectorias que ejecuta el robot móvil. El software de control utiliza Algoritmos Genéticos como técnica de inteligencia artificial.

RED ETHERNET

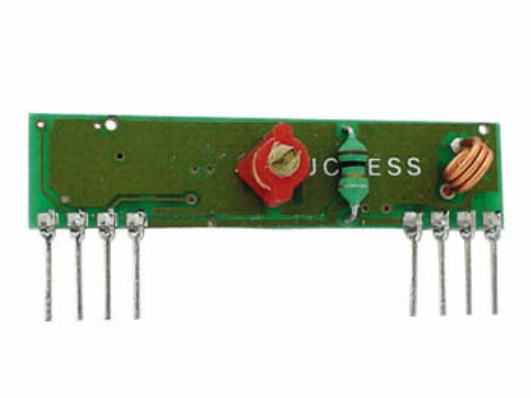
Para comunicar los sistema de visión y de control, se implementó una red de computadores tipo estrella (ETHERNET) con velocidad 10/100 Mbps, compuesta por dos tarjetas de red tipo “fast ethernet” y cuyo medio de transmisión es un cable UTP categoría 5 configurado en norma IEEE 802.3 A.

Los datos a transmitir por la red de computadores corresponden a las coordenadas de los obstáculos móviles y fijos obtenidas por el sistema de visión.

MÓDULOS RADIOFRECUENCIA

La comunicación se realiza entre el host de control y el móvil que ejecuta la trayectoria a través de unos módulos de radiofrecuencia que transmiten a una frecuencia de 433 Mhz, usando modulación AM.

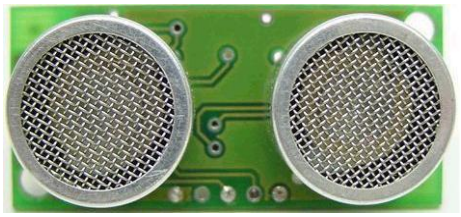
Las señales son enviadas desde el host de control vía puerto serial al módulo de transmisión, y captadas en el robot móvil por el receptor quien las entrega al controlador del mismo.



Receptor modulo de radiofrecuencia

SENSORES DE ULTRASONIDO

Los sensores de ultrasonido se encuentran alojados en la parte delantera del robot móvil, indican la presencia de obstáculos y limites de la pista, sirven como sistema de apoyo para el robot controlado y como sensor para el robot obstáculo móvil.



Sensor Ultrasonido

SOFTWARE DE CONTROL

Campos de Potencial

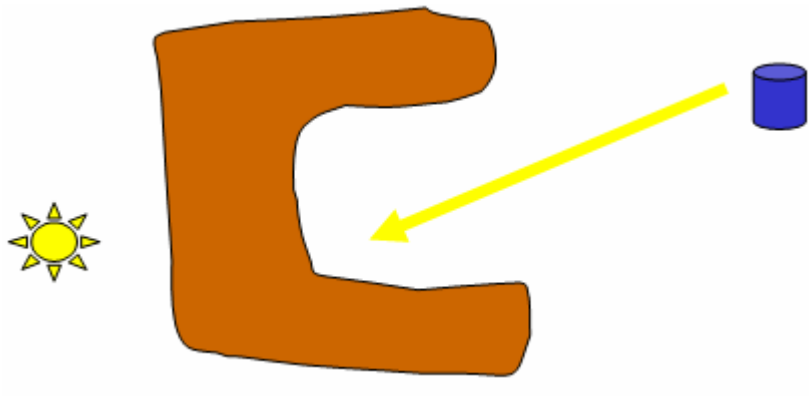
El método para planeamiento de campos de potencial trata el ambiente de trabajo como un campo de fuerzas, dando un valor diferente a cada objeto sobre este. El objetivo o punto de llegada se considera como un imán de polaridad contraria a la del móvil, de esta forma el móvil se ve atraído hasta la meta. Los obstáculos en el terreno son considerados imanes de igual polaridad al del móvil, produciendo como efecto que el robot se aleje de ellos.

Para obtener las fuerzas hay que modelar las funciones de potencial de la meta y de los obstáculos, calculando el potencial para cada punto del espacio libre.

Algunas ventajas del método de campos de potencial son:

- Se pueden generar trayectorias en tiempo real a partir del campo de fuerzas
- Las trayectorias generadas son suaves
- Permite acoplar las etapas de planeación y de control

La principal desventaja de este método es que conduce a que el robot caiga en mínimo locales en donde las diferentes fuerzas se cancelan entre si tal como se aprecia en la figura 7.



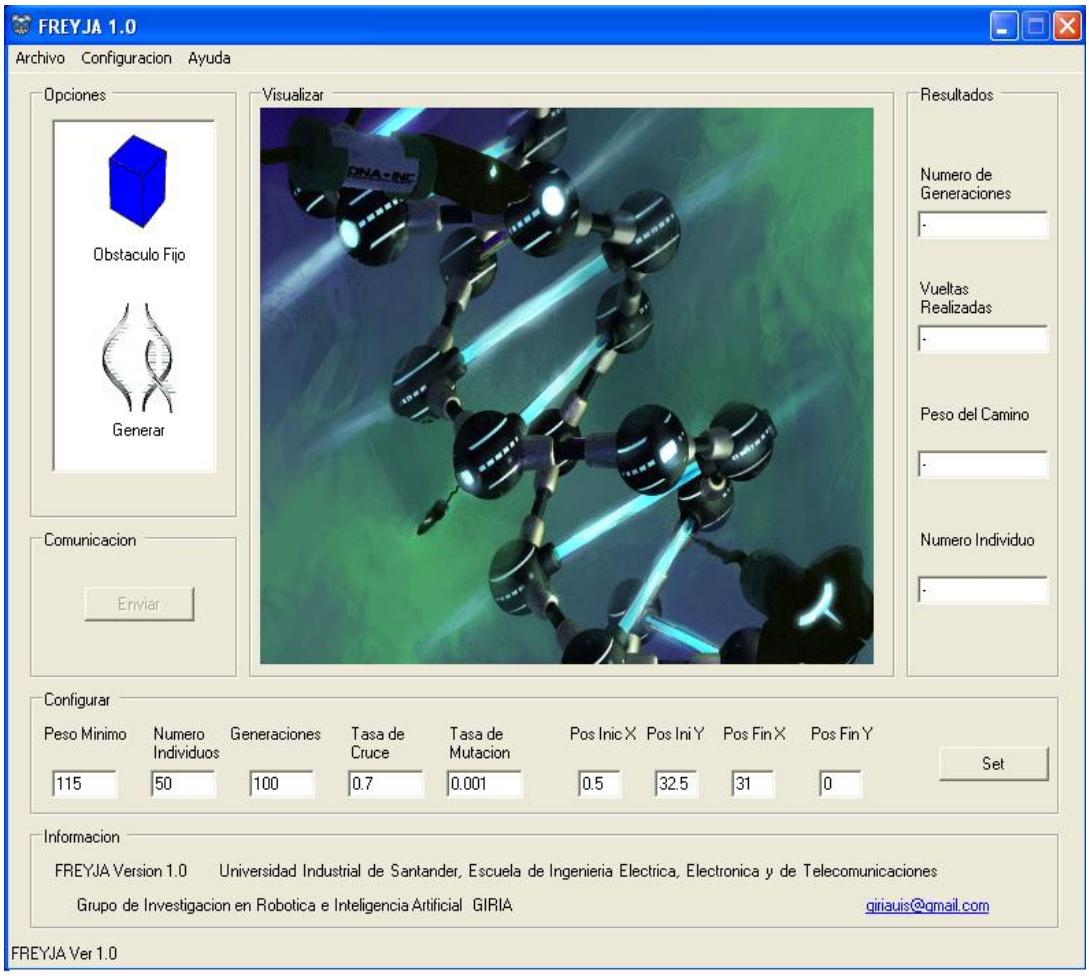
Mínimo Local en un Campo de Potencial, Fuente: Presentación: Robótica Inteligente. L. Enrique Sucar, Marco López

ANEXO E.

MANUAL DEL USUARIO

FREYJA 1.0

Captura de Pantalla Inicial



Captura de Pantalla Inicial

MENÚ SUPERIOR

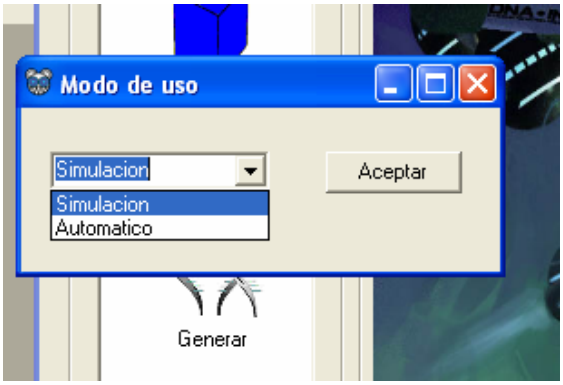
ARCHIVO

En la opción ARCHIVO del menú superior se encuentran las siguientes opciones.

Archivo

Nuevo.

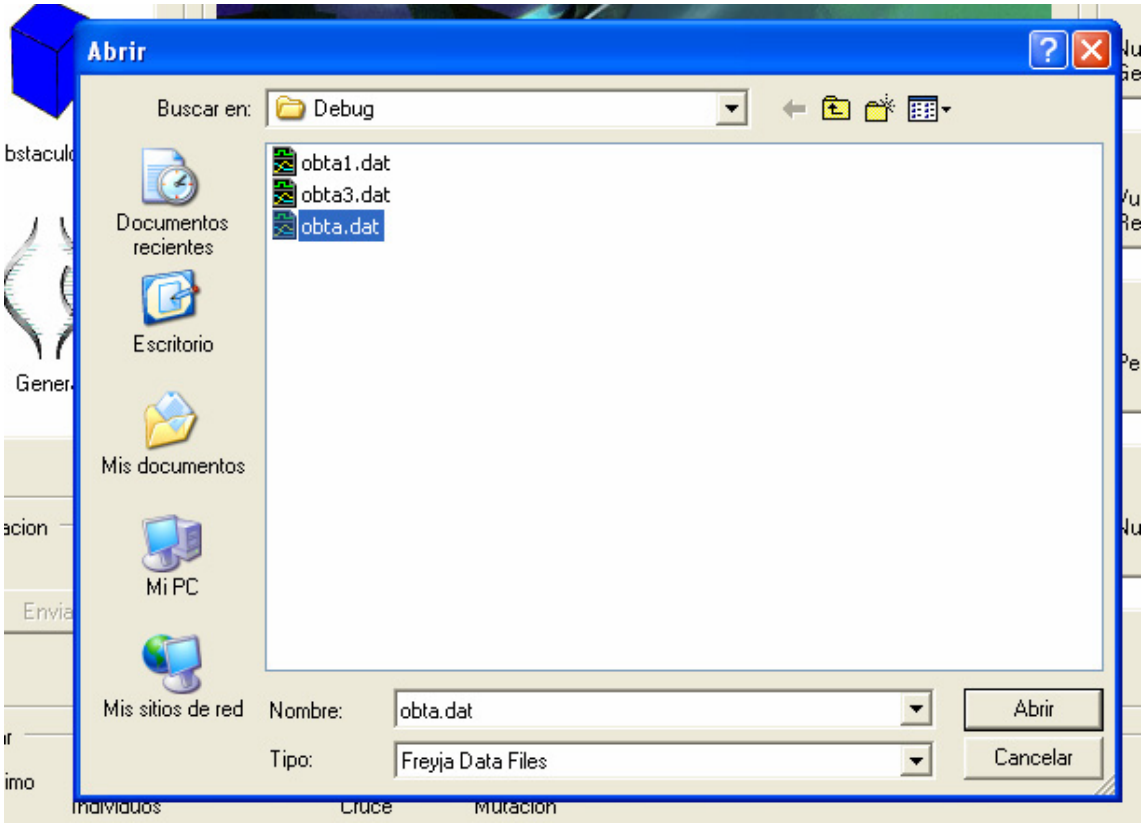
La opción “nuevo” permite escoger el modo de trabajo, ya sea el modo Automático (online), o el modo de simulación.



Nuevo

Abrir.

En el caso de que se seleccione el modo “Automático” se debe utilizar la opción “Abrir” para localizar el archivo binario en formato .dat que contiene la captura de los obstáculos.



Abrir

Guardar.

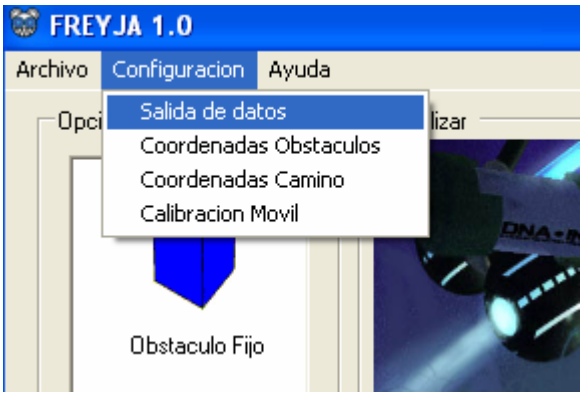
La opción guardar permite almacenar la trayectoria generada como un documento de texto.

Cerrar y Salir.

Ambas opciones cumplen la misma función que es cerrar “Freyja”.

CONFIGURACIÓN

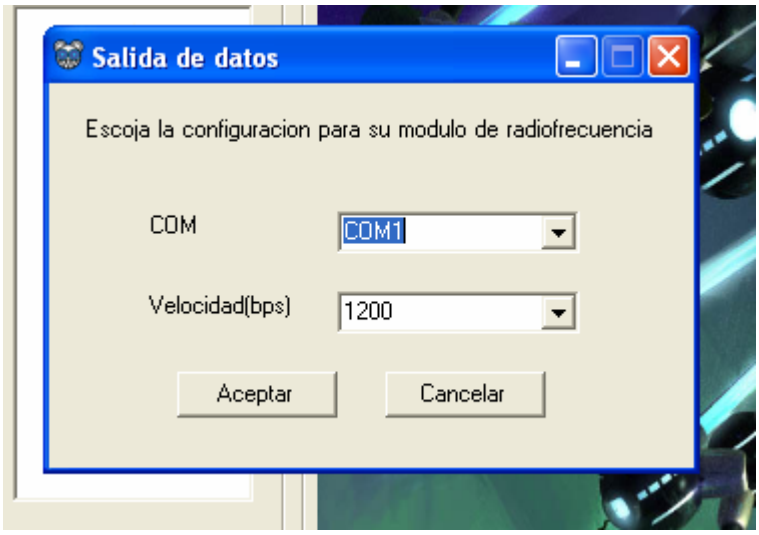
En el menú de configuración se encuentran las opciones, “Salida de datos”, “Coordenadas Obstáculos”, “Coordenadas Camino” y “Calibración Móvil”.



Configuración

Salida de Datos

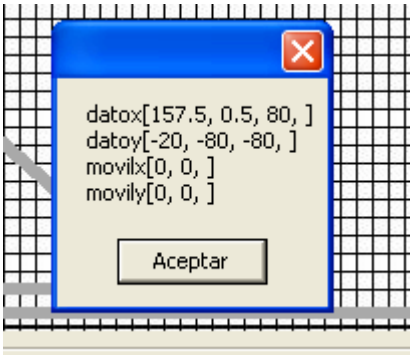
En la opción “Salida de datos” se configuran los parámetros de comunicación para que su sistema receptor pueda captar las señales que contienen las coordenadas del camino enviado. Estos parámetros son Número de COM y velocidad en bits por segundo.



Salida

Coordenadas obstáculos

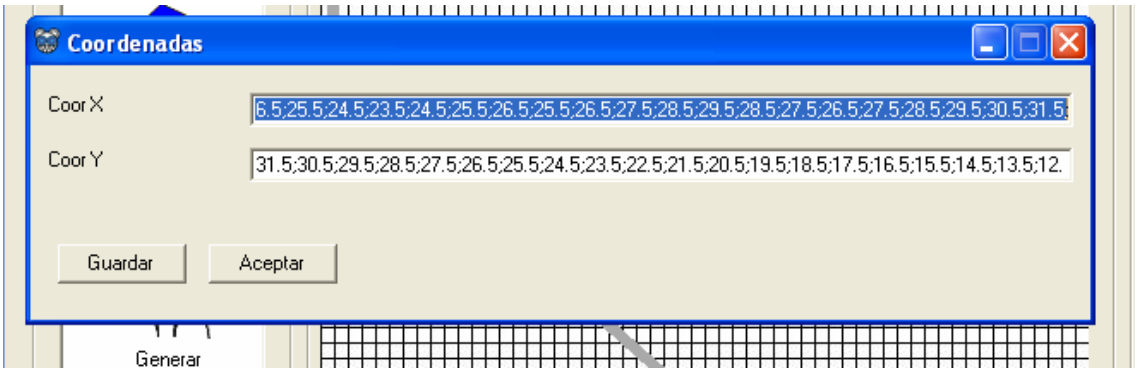
Muestra la ubicación en coordenadas espaciales de los obstáculos definidos para el robot móvil.



Obstáculos

Coordenadas camino

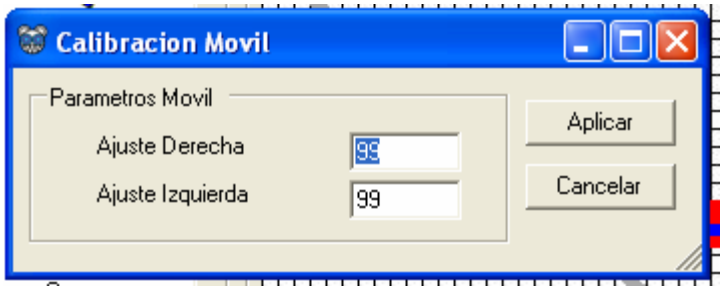
Indica las coordenadas numéricas que describen la trayectoria vencedora (en termino de coordenadas bidimensionales).



Coordenadas

Calibración móvil

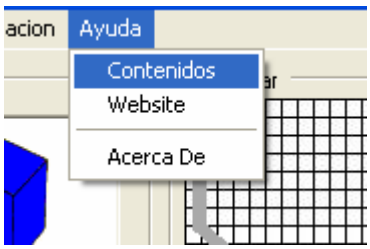
Contiene las opciones de calibración del robot móvil. Estos parámetros se ajustan para obtener ejecuciones más satisfactorias de las trayectorias por parte del robot, estas opciones son: Ajuste rueda derecha y Ajuste Rueda Izquierda.



Calibración

AYUDA

En el menú de ayuda se observan las opciones “Contenidos”, “Website” y “Acerca de”.



Ayuda

Respectivamente se accede a esta ayuda, a la Web del proyecto y a las características de la versión.

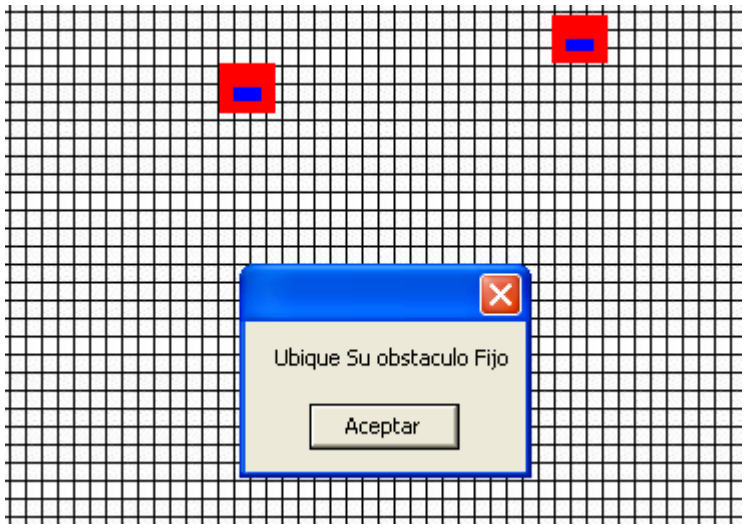
MENÚ DE OPCIONES

Opciones

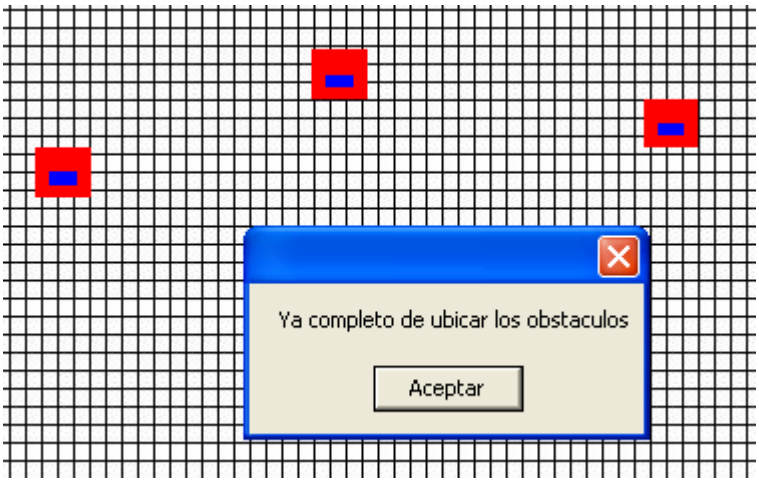
OBSTÁCULO FIJO

Con este proceso se ubican los obstáculos fijos dentro de la grilla mientras se este con el modo simulación activado.

Una vez se hace click en el cubo azul, el cursor cambia por una mano (dentro de la zona activa) y haciendo click de nuevo se ubicara un obstáculo fijo. Una vez se ubiquen 3 obstáculos, el sistema muestra un aviso que indica que ya no se puede ubicar más obstáculos. La zona azul del recuadro hace referencia al obstáculo físicamente, y la zona roja al área de riesgo alrededor de el.



Obstáculo

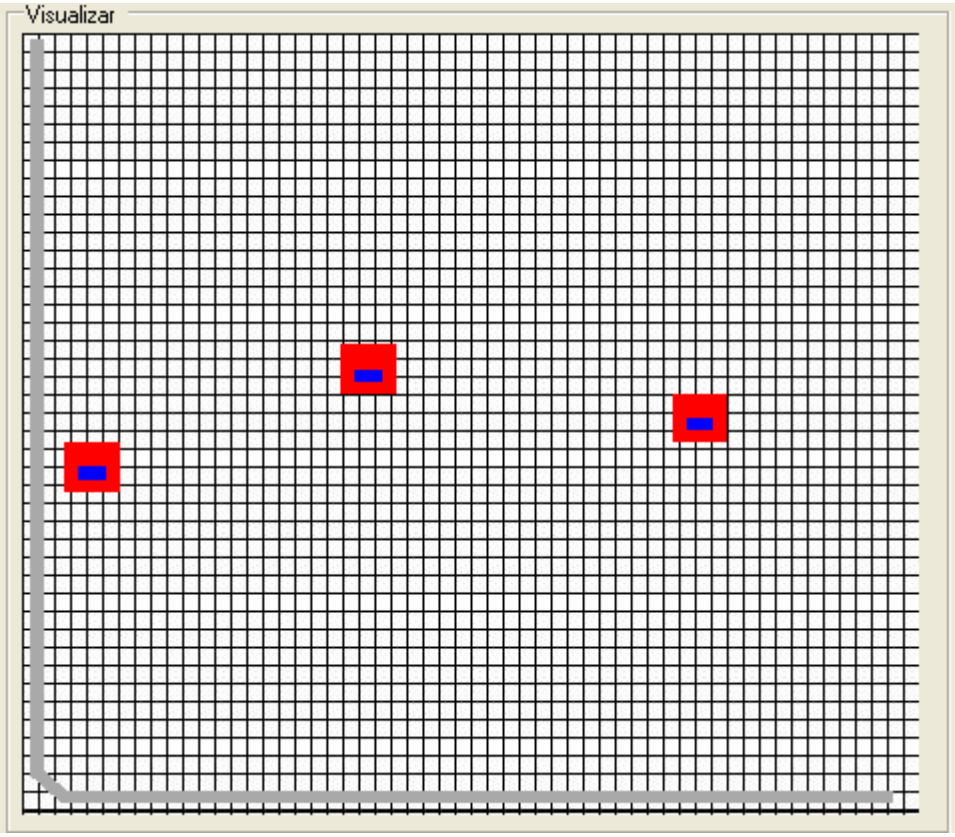


Final

GENERAR

La opción “Generar” activa un procedimiento que ejecuta el algoritmo genético. Cuando este finaliza, arroja un mensaje de alerta con el texto “Terminé” y el número del individuo donde se obtuvo una respuesta, además se visualiza en la grilla el camino resultante.

Terminé

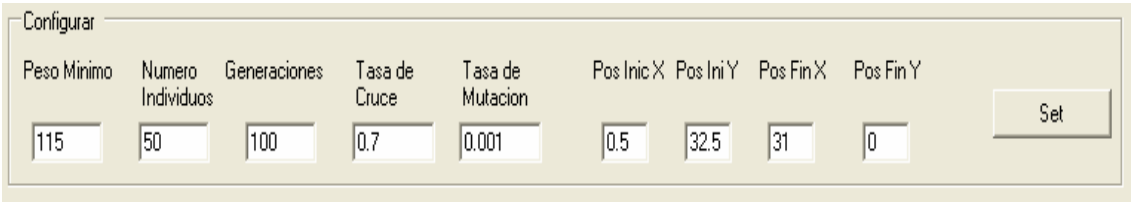


Resultado

MENÚ CONFIGURACIÓN

En el menú configuración se varían los parámetros fundamentales de los algoritmos genéticos, alterando así su funcionamiento y con ello sus respuestas.

También se pueden redefinir las posiciones de inicio y llegada del camino. Con el botón “Set” se define la nueva configuración y se podrá generar una trayectoria en función del nuevo arreglo de parámetros.

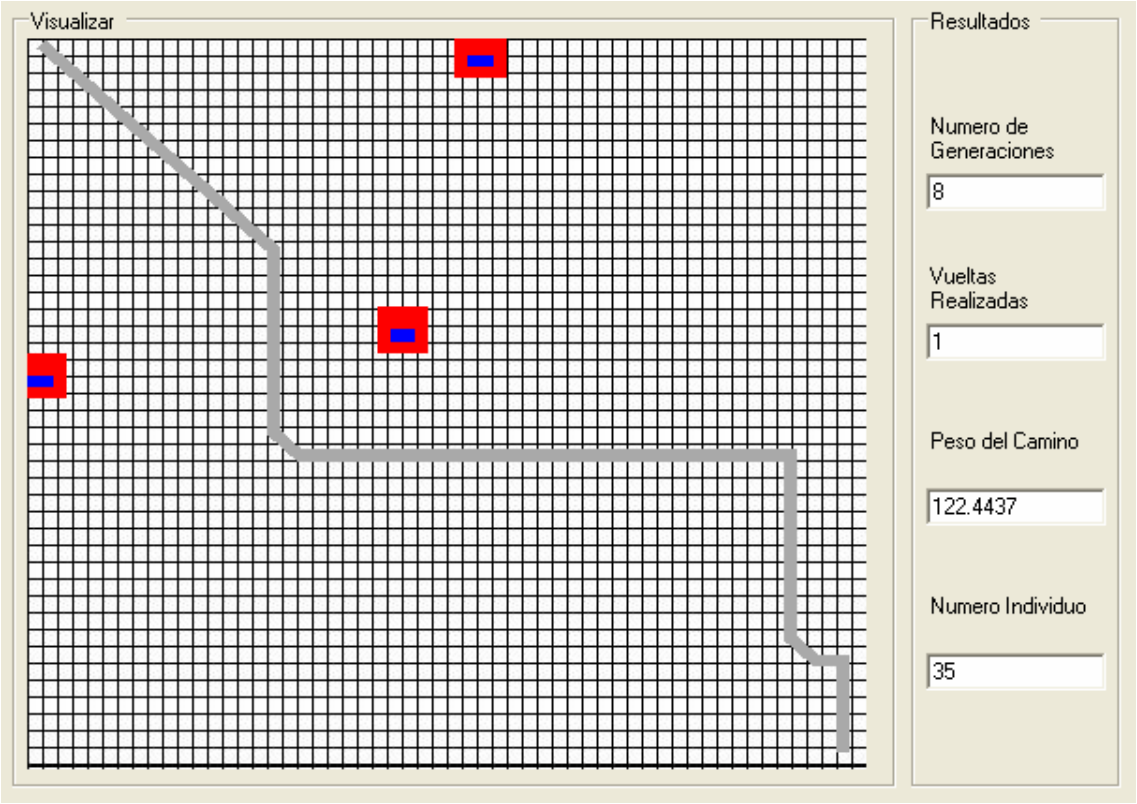


Configuración Parámetros

Las variables Tasa de Cruce y Tasa de Mutación solo son validas entre 0 y 1, y las de posición solo son validas entre (0.5 y 31.5) cada 0.5. El peso mínimo generado y las generaciones máximas deben ser enteros positivos, el número de individuos debe ser un entero positivo par superior a 4.

MENÚ RESULTADOS

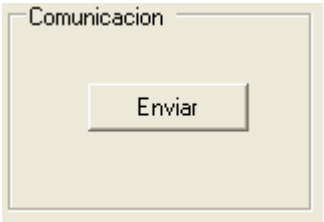
El menú resultados, es informativo y define el camino resultante en función de la generación en que se gestó (“Número de Generaciones”), las “Vueltas Realizadas”, el peso del camino resultante y el Número del individuo dominante de la generación ganadora.



Final Resultado

MENÚ COMUNICACIÓN

Consta del botón “Enviar” que se habilita automáticamente cuando hay un camino generado listo para ser emitido vía radiofrecuencia. El camino una vez generado, se puede enviar cuantas veces se desee siempre y cuando no se vuelva a utilizar la opción “Generar”.



Enviar