UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

IMPLEMENTACIÓN y VALIDACIÓN del ALGORITMO de ROBÓTICA de ENJAMBRE PSO en SISTEMAS FÍSICOS

Protocolo de trabajo de graduación presentado por Alex Daniel Maas Esquivel, estudiante de Ingeniería Mecatronica

Guatemala,

2021

Resumen

El siguiente protocolo de trabajo de graduación tiene como objetivo principal poder implementar y validar el algoritmo de robótica de enjambre *PSO* en un sistema físico. Para proceder con la implementación se tiene como objetivo el determinar tanto microcontrolador a utilizar como el lenguaje de programación y evaluar al tipo de robot comercial con el cual se estará trabajando en futuras pruebas. Se tomara en cuenta al algoritmo *PSO* desarrollado a nivel de simulación en fases anteriores, el cual se modificara ajustando sus parámetros al nuevo entorno de desarrollo, siendo este la mesa de pruebas con la que cuenta la Universidad del Valle de Guatemala.

Se procederá a realizar la migración y posteriormente la verificación del algoritmo por medio de pruebas simples, como calculo de rutas, transferencia de información hacia otros microcontroladores, re-calculo de rutas, ser capaz de localizar la meta deseada y comunicar estos resultados. Finalmente se buscara obtener los mismos resultados implementando el algoritmo en un robot físico comercial. Se buscara dejar robots y un algoritmo funcional a disposición de la Universidad del Valle de Guatemala para su posterior uso en cursos, investigación o futuras tesis y/o proyectos.

Antecedentes

En el departamento de Ingeniería Electrónica, Mecatrónica y Biomédica de la Universidad del Valle de Guatemala se tuvo la iniciativa de investigar, aprender y trabajar con la inteligencia de enjambre (*Robótica Swarm*), gracias a esto surgieron las fases 1, 2 y 3 del conocido "Megaproyecto Robotat".

Megaproyecto Robotat

En la fase I se procedió con el diseño y la elaboración de una plataforma para el Robotat donde se implementó un algoritmo de visión de computadora para poder obtener la posición y orientación del robot desde una perspectiva planar (2 dimensiones). Posteriormente en la fase II se tomo el modelo estándar y existen del PSO y se procedió a modificarlo para que este tome en consideración las dimensiones de robots físicos y la velocidad a la que estos pueden moverse, además se realizaron distintas pruebas buscando encontrar el mejor conjunto de parámetros y probando distintas funciones objetivo, todo esto para hacer un algoritmo mucho mas completo.

PSO

En la tercera fase, desarrollada por Eduardo Santizo se mejoro el desempeño del algoritmo de optimización de partículas PSO asiendo uso del método denominado *PSO Tunner*, el cual consiste de una red neuronal recurrente la cual es capaz de tomar diferentes métricas de las partículas PSO y tornarlas atravesé de su procesamiento por medio de una red LSTM, GRU

o BiLSTM y busca realizar una predicción de los hiper parámetros que debería emplear el algoritmo. Esta predicción es de carácter dinámico, lo que hace que en cada iteración se generan las métricas que describen al enjambre y use estos resultados para la siguen te iteración buscando así reducir el tiempo de convergencia y susceptibilidad a mínimos locales del PSO original [1].

Además elabora una alternativa al algoritmo de navegación alrededor de un ambiente conocido por medio de programación dinámica basándose en el ejemplo de programación dinámica Gridworld. Este nos presenta el caso donde un agente se mueve dentro de un espacio de estados representado en la forma de una cuadrícula y únicamente puede realizar 4 movimientos: Moverse hacia arriba, abajo, izquierda o derecha. De acuerdo a su estado actual y ultimo movimiento es capaz de moverse a un nuevo estado y obtiene una recompensa, el agente busca obtener el máximo de recompensas posibles generando así una ruta óptima desde cada estado hasta la meta [1].

Finalmente estos avances fueron compactados en un grupo de herramientas llamado Swarm Robotics Toolbox para ser usado en futuras fases. Debido a la pandemia ocasionada por el COVID-19, esta nueva propuesta de algoritmo únicamente fue probada a nivel de simulación, haciendo uso del programa Webots para las diversas pruebas realizadas con el robot e-puck.

Ant Colony

En la tercera tesis, desarrollada por Gabriela Iriarte se busco crear una alternativa al Modified Particle Swarm Optimization (MPSO) para su futuro uso en los Bitbots de la UVG, desarrollando así el algoritmo de planificador de trayectorias Ant Colony. Se implemento el algoritmo Simple Ant Colony, y después el algoritmo Ant System [2].

Se emplearon diversos controladores y se modifico el camino encontrado interpolándolo para crear más metas y lograr una trayectoria suave. Los parámetros encontrados se validaron por medio de simulaciones computarizadas permitiendo visualizar el comportamiento de colonia y adaptar los modelos del movimiento y de la cinemática a los Bitbots [2]. Lamentablemente por la pandademima ocasiona por el COVID-19 igual que el caso del PSO esta nueva propuesta de algoritmo se vio limitada a pruebas únicamente en simulación por medio del programa Webots.

Robotarium de Georgia Tech

El proyecto de Robotarium proporciona una plataforma de investigación robótica de enjambre de acceso remoto. En esta se permite a personas de todo el mundo hacer diversas pruebas con sus algoritmos de robótica de enjambre a sus propios robots con fines de apoyar la investigación y seguir buscando formas de mejorar las trayectorias definidas. Para utilizar la plataforma se debe descargar el simulador que se encuentra en la pagina del Robotarium el cual presenta la opción de descargarlo en Matlab o Python, se debe registrar en la página del Robotarium y esperar a ser aprobado para crear el experimento [3].

Se presenta una base plana con bordes que limitan el movimiento de los robots a utilizar, estas cumplen la función de fronteras del espacio de búsqueda. La base es de un color blanco para facilitar el procesamiento de imágenes y se cuenta con una cámara ubicada en el techo apuntando a toda la base, con esta lograr capturar las posiciones de los diferentes robots de pruebas [3].

Figura 1: Mesa de Pruebas del Robotarium de Georgia Tech [3]

Justificación

La parte primordial de cada nueva propuesta de algoritmo es su validación en una aplicación y ambiente real. Como proyecto la Universidad del Valle de Guatemala y su departamento de Ingeniería Macatrónica, Electrónica y Biomédica han buscado introducirse al mundo de la robótica Swarm y poder desarrollar su propio laboratorio enfocado tanto para la enseñanza como para la investigación.

El enfoque principal de esta Tesis, es dar el primer paso para hacia la creación de este laboratorio, tomando conocimientos ya existentes de fases anteriores y poder darles un cierre adecuado cumpliendo todos los objetivos previos propuestos para los algoritmos de robótica Swarm pero aplicados en un ambiente real.

Para esta Tesis se busca poder realizar la correcta migración del algoritmo de optimización de partículas PSO, que actualmente se encuentra en fase de simulación (Webots), y por medio de diferentes pruebas poder verificar el correcto funcionamiento de este y validar la correcta migración del algoritmo hacia una plataforma y entorno real, donde sera posible su futuro uso en robots físicos comerciales o desarrollados por el mismo departamento y poder usarlos en la mesa de pruebas ya existente en la Universidad del Valle de Guatemala.

Objetivos

Objetivo General

Implementar y validar el algoritmo de robótica de enjambre Particle Swarm Optimization desarrollados en años anteriores, a nivel de simulación, dándoles un enfoque hacia su implementación en robots físicos y un ambiente real.

Objetivos Específicos

- Evaluar distintas opciones de robots, microcontroladores, lenguajes de programación y entornos de desarrollo, y seleccionar los más adecuados para su uso en aplicaciones de robótica de enjambre, enfoque en el PSO.
- Migrar el algoritmo de robótica de enjambre PSO hacia el microcontrolador seleccionado y posteriormente a los robots físicos seleccionados.
- Validar la migración del algoritmo de robótica de enjambre y verificar el desempeño de los sistemas físicos mediante pruebas simples en ambientes controlados.

Marco teórico

Robótica Swarm

La robótica Swarm o en español Robótica de Enjambre es una nueva aproximación a la coordinación de un gran numero de robots relativamente simples. De forma que estos mismos robots sean capaces de de llevar a cabo tareas colectivas que están fuera de las capacidades de un único robot.

Se supone que un comportamiento colectivo deseado surge de las interacciones entre los robots y las interacciones de los robots con el entorno. Este enfoque surgió en el campo de la inteligencia artificial de enjambres, así como en los estudios biológicos de insectos, hormigas y otros campos en la naturaleza.

Este campo de investigación tiene justamente su inspiración en el comportamiento observado en los insectos sociales, en los que se destacan, las hormigas, termitas, abejas o avispas; los cuales son ejemplos de como un gran numero de individuos simples pueden interactuar para crear sistemas inteligentes colectivos.

Ventajas de la Robótica Swarm

Los sistemas robóticos de enjambre son tolerantes a fallos y robustos, ya que pueden continuar con su misión ante el fallo de alguna unidad. Aprovechan al máximo el paralelismo ya que el conjunto de robots ejecuta mas rápido cualquier tarea que un único robot, descomponiendo la tarea en subtareas y ejecútenlas de manera concurrente. Además que estos robots son bastante mas baratos y el coste de cualquier reparación también es bastante menor en comparación con cualquier gran robot [4].

Particle Swarm Optimization PSO

El algoritmo de Optimización del enjambre de partículas (PSO) fue creado por por el Dr. Russell Eberhart y el Dr. James Kennedy en el año de 1995. Este tiene origen la simulación de de comportamientos sociales utilizando herramientas e ideas tomadas por gráficos computa rizados e investigación sobre psicología social; además de una clara inspiración en el comportamiento social de las crías de aves y la educación de los peces [5] .

El algoritmo PSO pertenece a las técnicas denominadas optimización inteligente y se clasifica como un algoritmo estocástico de optimización basado en población. A esta clasificación igualmente pertenecen los Algoritmos Genéticos (AG). PSO es intrínsecamente paralelo. La mayoría de algoritmos clásicos operan secuencialmente y pueden explorar el espacio de solución solamente en una dirección a la vez [5].

Este algoritmo es iniciado y simula un grupo aleatorio de partículas a las cuales se les asigna una posición y velocidad inicial, a estas partículas se les conoce como "soluciones", para luego proceder a actualizar las generaciones de estas para encontrar la solución óptima. En cada iteración cada partícula es actualizada por los siguientes 2 mejores resultados [6].

El primero de estos se le conoce como la mejor solución lograda hasta ahora por cada partícula y recibe el nombre de *local best*. El segundo mejor valor es rastreado por el PSO y este proceso se repite hasta que se cumpla con un número de iteraciones especifica o se logre la convergencia del algoritmo. La convergencia se alcanza cuando todas las partículas son atraídas a la partícula con la mejor solución la cual recibe el nombre de *global best* [6]. La figura No. 2 representa una ejemplificación de como los varios *local best* deben de converger a un único *global best*.

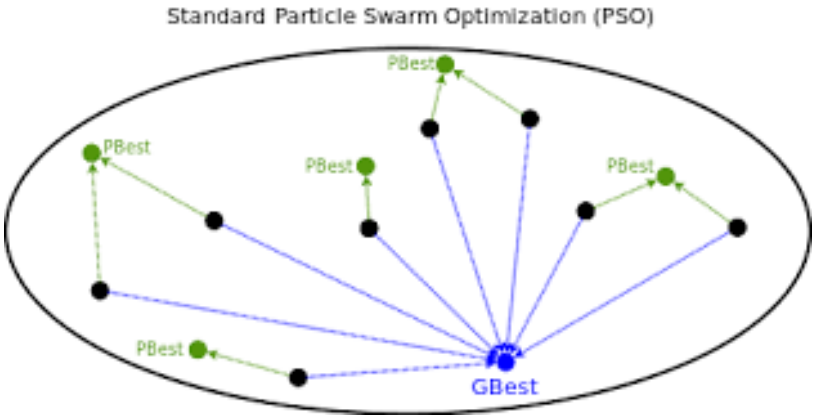


Figura 2: Ejemplo de convergencia [6]

Con estos datos es posible calcular los dos primeros factores principales de la ecuación del *PSO* para cada una de las partículas, los cuales son:

- Factor Cognitivo: $P_{local} - P_i$
- Factor Social: $P_{global} - P_i$

Donde P_i representa cada una de las soluciones/posiciones actuales. Para verificar que tan buena es la posición actual para cada partícula se usa la denominada *funcion fitness* que se da por $f(P)$ [7], el valor escalar que genera como resultado se le denomina costo el objetivo de las partículas es encontrar un conjunto de coordenadas que generen el valor de costo más pequeño posible dentro de una región dada.

Se toma en cuenta una diversidad de factores, entre ellos, el factor de escalamiento el cual consiste en hacer que las partículas tengan una mayor área de búsqueda o hacer que se concentren en un área mas reducida, definidas como $C1$ y $C2$. El factor de uniformidad son 2 numeros aleatorios que van entre 0 y 1; se definen como $r1$ y $r2$. El factor de constricción φ el cual se encarga de controlar la velocidad de cada partícula [7]. El factor de inercia w el cual se encarga de controlar cuanta memoria puede almacenar cada partícula. De forma que podemos armar la ecuación de velocidad brindada por el PSO.

$$V_{i+1} = \varphi[wV_i + C1r1(P_{local} - P_i) + C2r2(P_{global} - P_i)] \tag{1}$$

Una vez actualizadas las velocidades de todas las partículas,se calcula las posiciones de cada una de estas:

$$X_{i+1} = X_i + V_{i+1} * \Delta t \tag{2}$$

En la figura No. 3 vemos un ejemplo de los diferentes parámetros y como estos tienen efectos sobre cada una de las partículas.

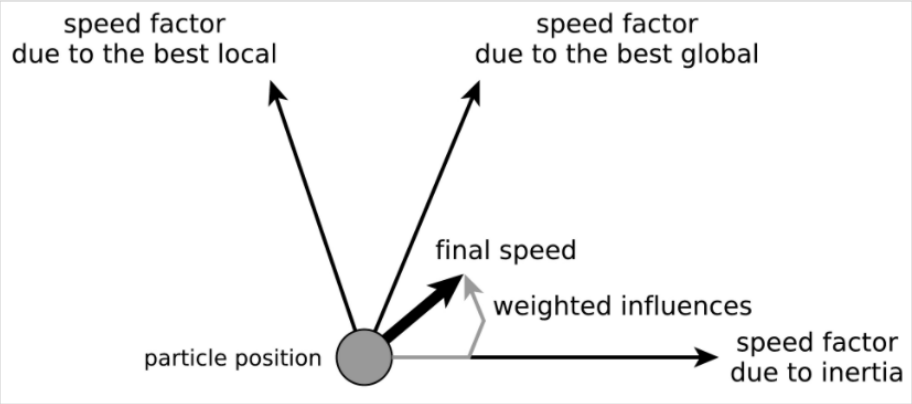


Figura 3: Efectos sobre la partícula [5]

Ant Colony Optimization

Este fue creado por Marco Dorigo en 1992 tomando como base el comportamiento de las hormigas las cuales a pesar de las limitadas capacidades de trabajo de cada hormiga individual, la colonia es capaz de realizar tareas de gran complejidad a través de la coordinación entre los individuos. La principal forma de comunicarse de las hormigas es modificar su ambiente esto debido a su escasa visión por lo que estas liberan feromonas para marcar rastros en el suelo e indicar el camino que deben seguir las demás [8].

Simple Ant Colony Optimization

El proceso de tomado para el caso mas simple del Ant Colony Optimization trata del problema general de la búsqueda del camino mas corto entre nodos presentes en un grafo $G = (V, E)$ donde E es la matriz que representa el conjunto de todas aristas del grafo y V el conjunto de todos los nodos del grafo. Cada posición dentro del grafo viene dado por un par de coordenadas (i, j) y en cada arista se tiene presenta cierta concentración de la feromona, denotado $\tau_{i,j}$. La concentración inicial de feronomas en cada arista se calcula de manera aleatorio, cabe resaltar que esto no ocurre en la vida real ya que no habría pasado ninguna hormiga aun por esos lugares [8].

Las hormigas son colocadas en una posición inicial denominado nodo fuente y estas tienen libertad de escoger hacia que arista deciden dirigirse inicialmente, esto se realiza aleatoria-

mente. Casa hormiga es representada por el símbolo K y el largo del camino recorrido viene dado por L^K. Se construye iterativamente una solución para el problema en cuestión, las soluciones intermedias se denominan estados solución y en cada iteración del algoritmo, cada hormiga se mueve de un estado i a un estado j correspondiendo a una solución intermedia más completa. La de terminación del movimiento de las hormigas esta basado en la probabilidad $p_{i,j}^k$ [2].

$$p_{i,j}^k(t) = \frac{(\tau_{i,j}(t))^\alpha}{\sum_{k \epsilon j_k}(\tau_{i,j}(t))^\alpha} \tag{3}$$

Esto siempre que $j \ \epsilon \ N_i^k$. Donde N representa al conjunto de nodos viables conectados al nodo i. Además α es una constante usada para modular la influencia de la concentración de feromona. Como se menciona previamente, luego de encontrar comida en un nodo denominado nodo destino estas vuelven al nodo fuente esparciendo feromonas en cada arista que pasaron, este despliegue de feromonas esta dada por:

$$\tau_{i,j}(t+1) = (1-\rho)\tau_{i,j}(t) + \sum_{k=1}^{m} \Delta\tau_{i,j}^k(t) \tag{4}$$

Donde ρ es el factor de evaporación de las feromonas y m es el numero de hormigas, $\Delta\tau_{i,j}^k(t)$ es la cantidad de feromonas depositadas por la kth hormiga y esta dado por Q/L_k donde Q es una constante y L_k es el costo de la ruta de la kth hormiga [2]. En la figura No.4 se puede ver un ejemplo de como un camino esta mucho mas resaltado que los demás, esto se debe a que en el hay una mayor concentración de feromonas dejadas por las hormigas haciendo referencia a que este es el mejor camino del puno A hacia el punto B.

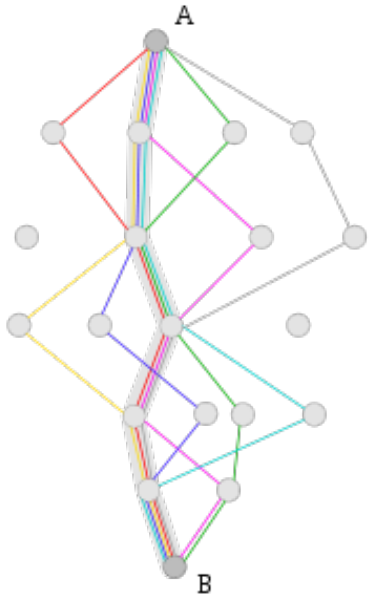


Figura 4: Demostración del camino con mayor concentración de feromonas [9]

Ant System (AS)

Este presenta ciertas mejoras al algoritmo descrito previamente, siendo este mas complejo. Una de las mejoras es la inclusión de la llamada información heurística en la ecuación de probabilidad y agregando capacidad de memoria con una lista tabú.

$$p_{i,j}^{k}(t) = \frac{(\tau_{i,j}(t))^{\alpha}(\eta_{i,j}(t))^{\beta}}{\sum_{k \epsilon j_k}(\tau_{i,j}(t))^{\alpha}(\eta_{i,j}(t))^{\beta}} \tag{5}$$

Donde η representa el inverso del costo de la arista [2].

Propuesta de Robots

Kilobot

El Kilobot es un sistema robótico de bajo costo, estos enjambres están inspirados en insectos sociales, como colonias de hormigas, que pueden buscar y encontrar eficientemente fuentes de alimento en grandes ambientes complejos, transportar colectivamente grandes objetos y coordinar la construcción de nidos y puentes en tales ambientes [9].

El Kilobot está diseñado para proporcionar a los científicos un banco de pruebas físicas para avanzar en la comprensión del comportamiento colectivo y realizar su potencial para ofrecer soluciones para una amplia gama de desafíos. Este tiene un máximo de 33mm de diámetro [9].

Algunas de sus características físicas son:

- Diámetro: 33 mm
- Altura: 34 mm
- Costo: 12 dolares
- Max. distancia de comunicación : 7 cm
- Autonomía: 1 horas en movimiento

E-puck

Este robot fue desarrollado por la Escuela Politécnica Federal de Lausana (EPFL) el cual fue creado para fines educativos y de investigación. Este pequeño robot es cilíndrico y cuenta con dos ruedas, equipado con una variedad de sensores cuya movilidad está garantizada por un sistema de accionamiento diferencial. Tanto su diseño como librerías y manuales de uso son open source y todo se encuentra disponible en su pagina oficial [10].

Algunas de sus características físicas son:

Figura 5: Kilobot [9]

- Diámetro: 70mm
- Altura: 50 mm
- Peso: 200 g
- Max. velocidad: 13 cm/s
- Autonomía: 2 horas en movimiento

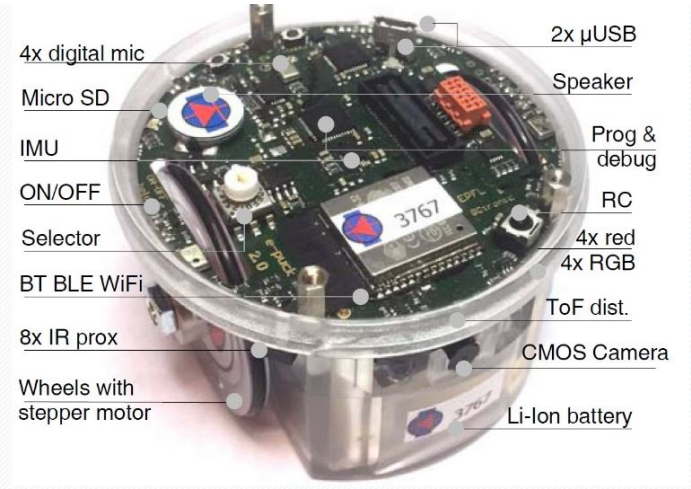


Figura 6: E-puck [10]

Pi-puck

El Pi-puck es una plataforma robótica desarrollada por el *York Robotics Laboratory* en la Universidad de York y GCtronic. Este básicamente es una extensión del e-puck y el e-puck2, el cual posee integrado una Raspberry Pi Zero agregando soporte Linux así como periféricos adicionales. Al ser de uso comercial se cuenta con una distribución de software YRL para el Pi-Puck en donde se incluye una imagen personalizada de Raspian y varios paquetes que permiten un completo control sobre el hardware y una seria de bibliotecas ya instaladas para hacer mas fácil el uso del robot.

La Raspberry Pi Zero se conecta al robot por medio de I2C, posee un micrófono digital y un altavoz. Consta te 2 baterías internas las cuales se cargan por medio de cable USB. Posee seis canales de I2C, dos entradas ADC así como múltiples leds para verificar funciones de encendido.

El precio de cada pi-puck es de Q 3,163.89 los cuales son distribuidos en Guatemala por *GCtronic*

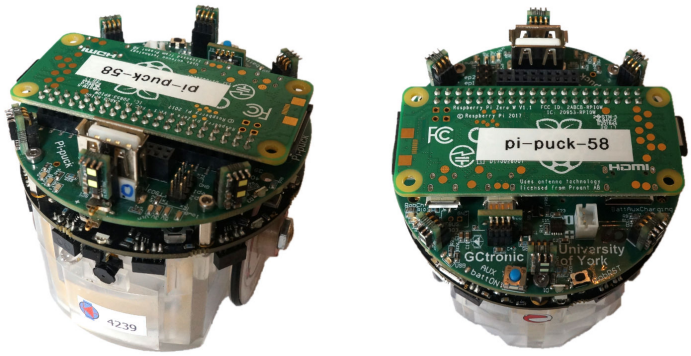


Figura 7: E-puck [10]

Metodología

Con el fin de alcanzar los objetivos planteados se llevara acabo la siguiente metodología:

Investigación sobre algoritmos de robótica de enjambre y determinación del algoritmo a desarrollar

Inicialmente se investiga sobre que es un algoritmo de robótica de enjambre, aplicaciones y como funcionan estos algoritmos. Gracias a la documentación de tesis anteriores se logra visualizar de mejor forma las aplicaciones que se les dieron a los dos algoritmos de robótica de enjambre desarrollados para la UVG, estos son el *PSO* y el *Ant-Colony*. Se determino que este proyecto de graduación se enfocara en el *PSO* y se procede a ejecutar y evaluar todo lo desarrollado hacia el *PSO* en tesis anteriores.

Evaluación de Microcontroladores

En esta etapa se debe seleccionar el microcontrolador con el cual se buscara ejecutar el algoritmo de optimización de partículas *PSO*, para esto se toman en cuenta los diferentes tipos de microcontroladores usados a lo largo de los años de carrera en la Universidad del Valle de Guatemala ya que se busca que el proceso de aprendizaje de este sea lo menos complicado posible y se tenga experiencia previa utilizándolo.

El proceso de selección comienza al comparar los microcontroladores, buscando una buena capacidad de procesamiento, compatibilidad con diferentes sensores, protocolos de comunicación, capacidad para diferentes lenguajes de programación y facilidad de adquirirlo para el desarrollo del algoritmo. Entre los tomados en cuenta tenemos al PIC17F877A, Arduino UNO, Tiva C y Raspberry pi 4.

Evaluación de Lenguaje de Programación

Para esta fase se deberá haber definido previamente el microcontrolador a utilizar para la implementación del algoritmo de optimización de partículas, ya que como parte de la evaluación del lenguaje de programación se realizara una investigación de los distintos lenguajes que maneja el microcontrolador seleccionado.

Se buscara un lenguaje de programación que haga sentido con el microcontrolador, además , que se tenga o exista un previo conocimiento parcial o totalmente del mismo ya que no es de interés de este proyecto el conocer desde cero todo un nuevo lenguaje de programación.

Evaluación de Tipos de Robots

El proceso de evaluación de diferentes robots móviles se basara en realizar investigaciones de los actuales robots móviles en le mercado para su futura comparación. La comparación

estará basada en aspectos tales como, tamaño, peso, precio (tomando en cuenta el presupuesto de la Universidad), Hardaware con el que cuente y su facilidad de adaptarle el microcontrolador seleccionado.

Migración de Algoritmos

Para esta fase y para poder hacer el proceso de migración ya se habrán definido el microcontrolador a utilizar así como el lenguaje de programación y el tipo de robot al cual estará enfocado el algoritmo de optimización de partículas, para la migración se tomara como base/guía los códigos previamente desarrollados en fases anteriores para su uso en la Universidad del Valle de Guatemala.

Se buscaran maneras de optimizar y adecuar estos códigos para su correcta implementación en la nueva plataforma y nuevo lenguaje de programación seleccionado. Se desarrollaran funciones, cases, condicionales buscando que el código se ejecute de la mejor forma posible.

Validación de migración de Algoritmos

Para la validación de la migración de Algoritmos primero se verificara por medio de software que la migración haya sido correcta, esto se comprobara al momento de compilar no se tengan errores o advertencias de ningún tipo.

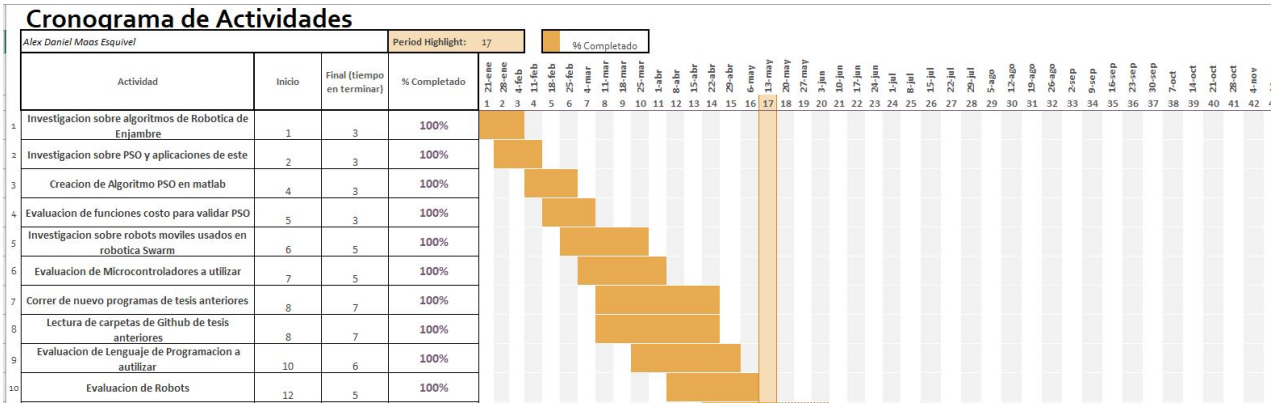
Una vez lograda esta verificación se procederá a validar el algoritmo de robótica de enjambre PSO ya como tal. Para esto se usara únicamente el microcontrolador realizando pruebas simples o pequeñas tareas como calcular una ruta, detectar obstáculos y poder recalcular la ruta, poder comunicarse con otros microcontroladores simulando el proceso de búsquedas de partículas y poder detenerse al momento de llegar a la meta.

Se planea utilizar el equipo de la Universidad del Valle de Guatemala para que sea posible la realización de las tareas previamente descritas para contar con la mayor cantidad de microcontroladores posibles y evaluar de la mejor forma el PSO. Además se buscara utilizar la mesa de pruebas que ya se cuenta en el departamento para realización de estas pruebas.

Validación por medio de robots físicos

De ser posible se planea evaluar la migración de algoritmos repitiendo las mismas pruebas pero ya en un robot físico comercial, seleccionado previamente que ya cuente con sensores propios, ruedas móviles y una carcasa apropiada.

Cronograma de actividades

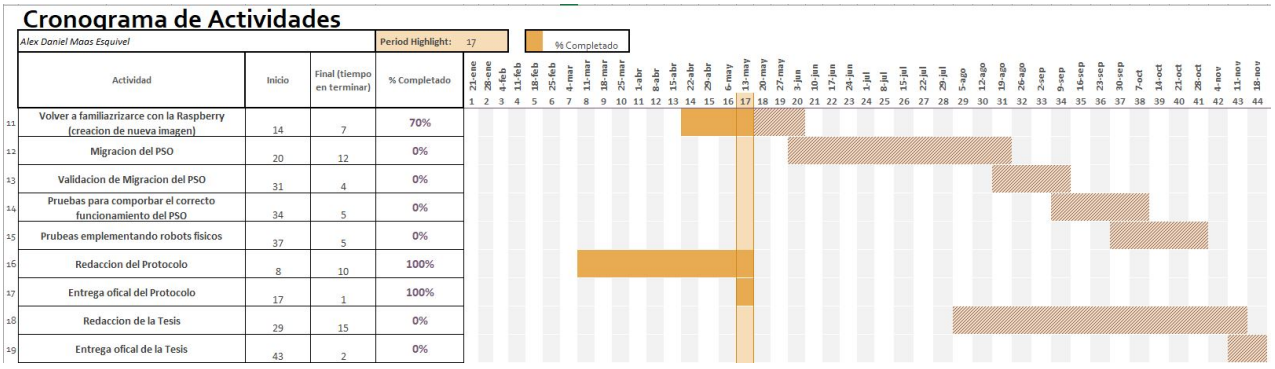


Cronograma de Actividades

Alex Daniel Maas Esquivel — Period Highlight: 17 — % Completado

	Actividad	Inicio	Final (tiempo en terminar)	% Completado
1	Investigacion sobre algoritmos de Robotica de Enjambre	1	3	100%
2	Investigacion sobre PSO y aplicaciones de este	2	3	100%
3	Creacion de Algoritmo PSO en matlab	4	3	100%
4	Evaluacion de funciones costo para validar PSO	5	3	100%
5	Investigacion sobre robots moviles usados en robotica Swarm	6	5	100%
6	Evaluacion de Microcontroladores a utilizar	7	5	100%
7	Correr de nuevo programas de tesis anteriores	8	7	100%
8	Lectura de carpetas de Github de tesis anteriores	8	7	100%
9	Evaluacion de Lenguaje de Programacion a autilizar	10	6	100%
10	Evaluacion de Robots	12	5	100%

Figura 8: Cronograma de actividades



Cronograma de Actividades

Alex Daniel Maas Esquivel — Period Highlight: 17 — % Completado

	Actividad	Inicio	Final (tiempo en terminar)	% Completado
11	Volver a familiazrizarce con la Raspberry (creacion de nueva imagen)	14	7	70%
12	Migracion del PSO	20	12	0%
13	Validacion de Migracion del PSO	31	4	0%
14	Pruebas para comporbar el correcto funcionamiento del PSO	34	5	0%
15	Prubeas emplementando robots fisicos	37	5	0%
16	Redaccion del Protocolo	8	10	100%
17	Entrega ofical del Protocolo	17	1	100%
18	Redaccion de la Tesis	29	15	0%
19	Entrega ofical de la Tesis	43	2	0%

Figura 9: Cronograma de actividades

Índice preliminar

Referencias

[1] E. A. S. Olivet, "Aprendizaje Reforzado y Aprendizaje Profundo en Aplicaciones de Robótica de Enjambre," Tesis de licenciatura, Universidad del Valle de Guatemala, 2020.

[2] G. I. Colmenares, "Aprendizaje Automático, Computación Evolutiva e Inteligencia de Enjambre para Aplicaciones de Robótica," Tesis de licenciatura, Universidad del Valle de Guatemala, 2020.

[3] Jason Maderer, *Robotarium: A Robotics Lab Accessible to All*, `https://www.news.gatech.edu/features/robotarium-robotics-lab-accessible-all`, Accessed: 2021-03-27, 2017.

[4] L. S. Tortosa, "Ajentes y enjambres artificiales: modelado y comportamientos para sistemas de enjambre robóticos," Tesis doct., Universidad de Alicante, España, 2013.

[5] R. F. R. Grandi y C. Melchiorri, *A Navigation Strategy for Multi-Robot Systems Based on Particle Swarm Optimization Techniques*. Dubrovnik, Croatia: Dubrovnik, 2012.

[6] C. Duarte y C. J. Quiroga, *PSO algorithm*. Ciudad Universitaria, Santander, Colombia: Santander, 2010.

[7] A. S. A. Nadalini, "Algoritmo Modificado de Optimización de Enjambre de Partículas (MPSO)," Tesis doct., Universidad del Valle de Guatemala, Guatemala, Guatemala, 2019.

[8] M. Pedemonte, "Ant Colony Optimization," Tesis doct., Instituto de Computación, Universidad de la República Montevideo, Uruguay, Montevideo, Uruguay, 2015.

[9] K-Team, *KILOBOT*, `https://www.k-team.com/mobile-robotics-products/kilobot`, Accessed: 2021-02-25, 2017.

[10] GCtronic, *e-puck education robot*, `http://www.e-puck.org`, Accessed: 2021-03-27, 2018.