TECNOLÓGICO NACIONAL DE MÉXICO

INSTITUTO TECNOLÓGICO DE TAPACHULA

SEP
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MEXICO

INSTITUTO TECNOLÓGICO DE TAPACHULA

IMPLEMENTACIÓN DE UN ALGORITMO DE SINCRONIZACIÓN DE SEMÁFOROS USANDO INTELIGENCIA ARTIFICIAL

INGENIERÍA EN SISTEMAS COMPUTACIONALES

PRESENTAN:

CAMACHO BELLO DARCY MICHELLE	13510636
ESCOBAR ZAMORA CARLOS	13510650
FARRERA RAMOS JAVIER ANTONIO	12510510

ASESOR INTERNO:

LIC. ANAMIM VILLARREAL WONG

ASESOR EXTERNO:

LIC. MONICA SIBLINA MARTINEZ SOLIS

Periodo:

Enero- Junio
2018

Índice general

Índice de figuras

Capítulo 1

Generalidades del proyecto

1.1. Introducción

En la ciudad de Tapachula Chiapas el índice demográfico se encuentra en constante crecimiento por lo que la movilidad urbana o el total de desplazamientos que se realizan en la ciudad es un factor importante que genera congestionamiento vehicular. La adecuada operación de los semáforos en las intersecciones es otro factor importante, debido a que, son estos quienes dirigen el flujo del tráfico.

Desafortunadamente los semáforos convencionales tienen asignados tiempos fijos para el cambio de luces, esto a menudo ocasiona largos tiempos de espera innecesarios. Por ejemplo, cuando un semáforo asigna una fase verde con tiempo considerable a la avenida en que hay pocos o incluso, ningún automóvil.

En el presente documento se redacta el desarrollo de un sistema "inteligente" que hará uso de herramientas y técnicas de Inteligencia Artificial para la sincronización de los semáforos.

El sistema desarrollado será capaz de gestionar los semáforos de una intersección de 4 vías con doble sentido. Para esto el sistema reaccionará a variables de su entorno como:

- Cantidad de carriles de las avenidas.
- Congestión de toda la intersección.

Con este proyecto se espera alentar a los alumnos a que sigan investigando sobre el tema, ya que una buena gestión de tráfico no solo favorece la movilidad vehicular, sino también, disminuye la emisión de gases contaminantes (producto de los autos varados en los cruces), lo cual es un tema muy importante actualmente debido al calentamiento global.

1.2. Descripción de la empresa

En el presente capítulo se describen los datos del "Instituto Tecnológico de Tapachula" lugar en donde se desarrolló el proyecto, también se encuentra información acerca de la localización y el área en donde se elaboró el proyecto.

Nombre o Razón social

Tecnológico Nacional de México.

Breve descripción de la empresa

El instituto tecnológico de Tapachula es una institución educativa perteneciente al sistema nacional de institutos tecnológicos, que a su vez forma parte de la dirección general de educación superior tecnológica. Cuenta con las siguientes carreras:

- Ing. Civil.
- Ing. Industrial.
- Ing. Química.
- Ing. Electromecánica.
- Ing. En Sistemas Computacionales.
- Ing. En Gestión Empresarial.

Antecedentes del ITT

El Tecnológico Nacional de México Campus Tapachula pertenece al Tecnológico Nacional de México, que está integrada por 218 Institutos Tecnológicos y Centros Especializados, distribuidos en el territorio Mexicano. De ellos, 110 son de carácter federal, entre los que destacan 104 Institutos Tecnológicos Industriales, dos Centros Especializados y cuatro Centros de

Desarrollos Tecnológico. A lo mismo se unen 108 Tecnológicos Descentralizados, los cuales han servido al país durante más de 57 años de vida, siempre con el compromiso de hacer el mejor de sus esfuerzos; procurando que la educación que se imparten en dichas Instituciones Educativas responda a las exigencias de los más altos estándares de calidad educativa.

Atendiendo a las líneas de desarrollo regional para asegurar la pertenecía de los planes y programas de estudio; conscientes de que representan una vía de desarrollo, de esperanza, de inclusión y de movilidad social para los jóvenes de la provincia mexicana. El 16 de Mayo de 1983, el Instituto Tecnológico de Tapachula, abre sus puerta a la superación profesional a través de la carrera de Ingeniería Civil, además continua a 148 alumnos con nivel medio superior con una carrera terminal en tecnólogos en construcción y tecnólogos en electrotecnia, de igual manera absorbe a la población de nivel de licenciatura del CeRETI, en las carreras de Ingeniería Industrial en alimentos e Ingeniería Civil, permitiendo a los alumnos de ésta última cambiarse al plan de tecnológico.

Se autoriza el 15 de Noviembre de 1984; la apertura de la carrera de ingeniería Química, inscribiéndose para el semestre inicial, Septiembre 85 - Febrero 86, un total de 73 alumnos, en ese entonces el C. Ing. Jorge Elí Castellanos Martínez, como director del plantel, fue el encargado de darles la bienvenida.

El 29 de Mayo de 1985, siendo el director del plantel el C. Jorge Carlos García Revilla, se autoriza la carrera de Ingeniería Industrial, matriculando 54 alumnos para el semestre Septiembre 86 - Febrero 87. Uno de los objetivos primordiales del instituto es brindar a la juventud estudiosa del estado de Chiapas, la oportunidad de formación y superación profesional a través de las diferentes carreras que se imparten, ampliando la oferta educativa; es por ello que en 1990 se crea la carrera de licenciatura en Informática, con una población de 70 alumnos, y es el C. Ing. Víctor Manuel Ibarra Balderas, director del plantel, el encargado de darle la bienvenida a los alumnos de nuevo ingreso.

Un nuevo estudio sobre la demanda educativa en el estado, muestra la necesidad de proporcionar una nueva opción de formación profesional, en respuesta a ello el 28 de Enero de 1993, siendo el director el C. Ing. José Luis Méndez Navarro, se autoriza la carrera de Ingeniería

Electromecánica, iniciándose en el mes de Agosto del mismo año con una población de 29 alumnos. Las necesidades de la región, así como un nuevo estudio de expectativas, dieron como resultado que el 18 de Junio del 2002, autorizaran la carrera de Ingeniería en Sistemas computacionales. El C. M.A. Juan Amado Rueda Ibarra, como máxima autoridad de la institución les da la bienvenida, el 18 de Agosto de 2003, a los 45 miembros de la primera generación de esta carrera. También comparte el conocimiento científico y tecnológico con el público en general, a través de su programa de educación continua, el cual está compuesto por diferentes cursos de interés general, destacándose los del idioma de idioma de inglés en sus diferentes niveles. Los programas de servicio social y residencia profesional han permitido atender en las peticiones a más de 150 Instituciones municipales, estatales y federales de los sectores públicos y privado.

Como parte del compromiso que se tiene con la sociedad como institución educativa, el Instituto Tecnológico de Tapachula orgullosamente ha obtenido la certificación del proceso educativo de acuerdo a la norma ISO 9001:2000, cuyo certificado RSGC 247 le fue entregado el 2 de Octubre de 2006, y que en nombre de los trabajadores del Instituto Tecnológico lo recibió el Ing. Herman Calderón Pineda, en ese entonces director.

Tapachula, Abril 14.- Con entusiasmo fue recibida la noticia por la "comunidad tecnológica" la designación del maestro en ciencias de la administración, Miguel Cid del Prado Martínez, como nuevo Director del Instituto Tecnológico de Tapachula (ITT); designación realizada con fecha 24 de Marzo del año en curso por el doctor Carlos Alfonso García Ibarra, Director General de Educación Superior Tecnológica de la SEP.

Correspondió al Doctor Héctor Francisco Macías Díaz, Director de Capacitación y Desarrollo de la DGEST, quien en calidad de representante del Director General hizo la presentación del nuevo director, Del Prado Martínez, quien -dijo- venía fungiendo como subdirector de los servicios administrativos del Instituto Tecnológico de Tuxtla Gutiérrez, donde además desempeñó los cargos de profesor de licenciatura y posgrado, fue Jefe de la División de Estudios de Posgrado e Investigación, Jefe del Departamento de Ingeniería Química, Coordinador de la Especialización en Ingeniería Ambiental, Coordinador de la Maestría en Ciencias de la

Administración y Coordinador de Educación a Distancia.

El Ing. Pedro Ancheyta Bringas, en su calidad de Director del Instituto Tecnológico de Tapachula, manifestó el doble compromiso que para él representa la nueva encomienda: como lealtad y responsabilidad por ser egresado del Tecnológico de Tapachula, pero hoy asume dicha función con todos los deseos de sumarse al trabajo"

El 5 de Abril del 2017.- El Maestro Manuel Quintero Quintero Director General del Tecnológico Nacional de México (TecNM), nombró a la maestra Rosa Aidé Domínguez Ochoa como nueva Directora del Instituto Tecnológico de Tapachula, quien asumió sus funciones con esta fecha.

Actualmente, se cuenta con una población estudiantil de: 1 mil 801 alumnos, distribuidos en las diferentes carreras.

El Instituto Tecnológico de Tapachula Nº 51 se dedica a contribuir a la conformación de una sociedad más justa, humana y con amplia cultura científico-tecnológica, mediante un sistema integrado de educación superior tecnológica, equitativo en su cobertura y de alta calidad.

El Departamento de Sistemas y Computación del Instituto Tecnológico de Tapachula su giro es público.

Misión

Contribuir a la conformación de una sociedad más justa, humana y con amplia cultura científico-tecnológica, mediante un sistema integrado de educación superior tecnológica, equitativo en su cobertura y de alta calidad.

Visión

El Sistema Nacional de Institutos Tecnológicos se consolidará como un sistema de educación superior tecnológica de vanguardia, así como uno de los soportes fundamentales del desarrollo sostenido, sustentable y equitativo de la nación y del fortalecimiento de su diversidad cultural.

Valores

- El ser humano
- La calidad
- El trabajo en equipo
- El liderazgo
- El espíritu de servicio
- El alto desempeño

Organigrama

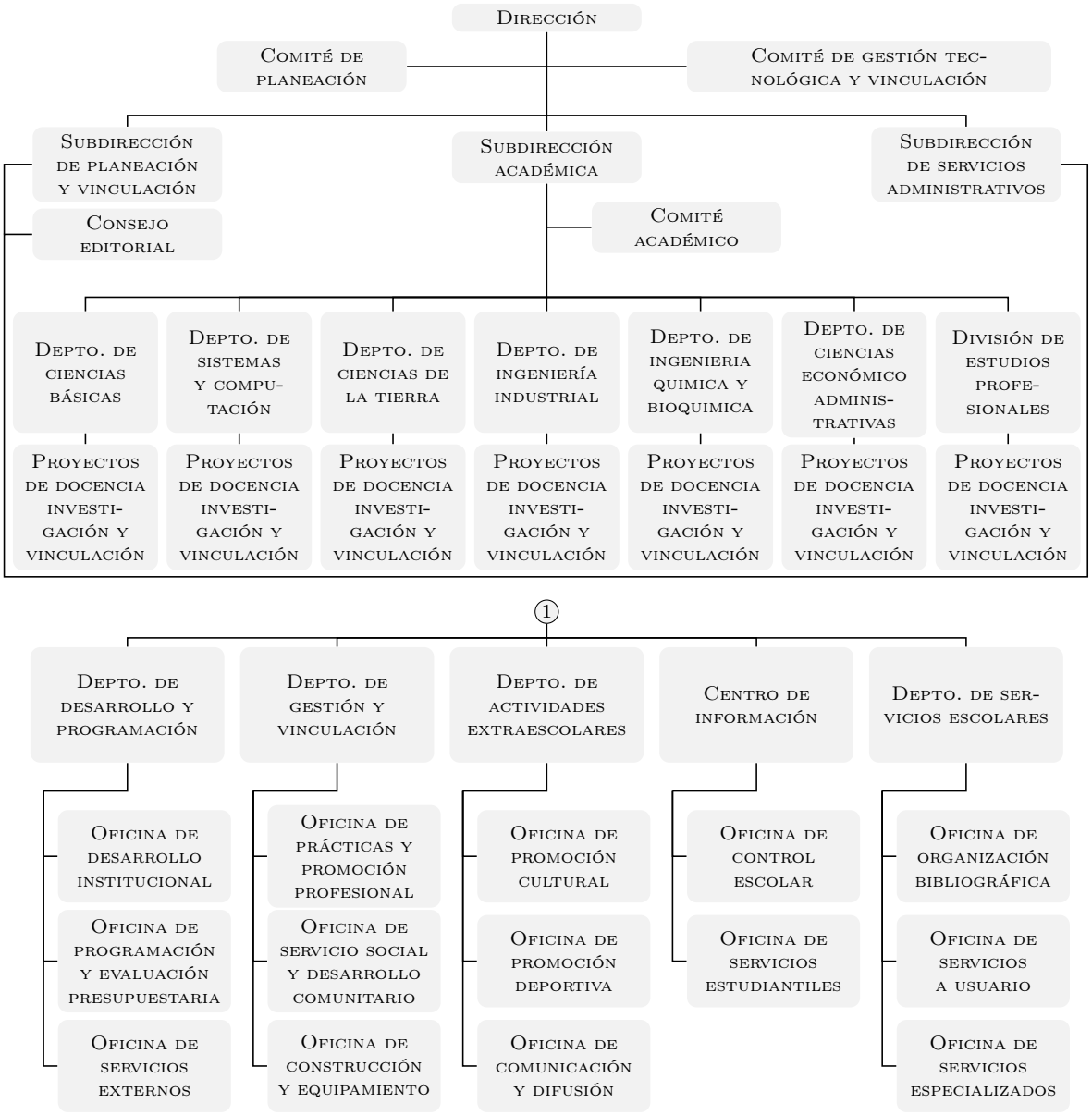


Figura 1.1: Organigrama del Instituto Tecnológico de Tapachula

Ubicación del Instituto Tecnológico

El Instituto Tecnológico de Tapachula, se encuentra ubicado en el Km. 2 Carretera a Puerto Madero. C.P. 30700 Tapachula, Chiapas.

Macro-localización

La figura 1.2 muestra una vista aérea de la Macro Localización.

Figura 1.2: Macro Localización del Instituto Tecnológico de Tapachula

Ubicación del Área en donde se elaboró el proyecto

El proyecto "implementación de un algoritmo de sincronización de semáforos usando inteligencia artificial" se desarrolla en el Departamento de Sistemas y Computación, en el área de Investigación y Desarrollo del Instituto Tecnológico de Tapachula, ubicado en el edificio "Centro de información".

Figura 1.3: Micro-Localización del Instituto Tecnológico de Tapachula

Características del Área o Departamento

Departamento de Investigación y Desarrollo

1. Planear, coordinar y evaluar las actividades de docencia, investigación y vinculación de las áreas correspondientes a sistemas y computación que se imparten en el Instituto Tecnológico de Tapachula, de conformidad a las normas y lineamientos establecidos por la secretaria de educación pública.

2. Elaborar el programa operativo anual y el anteproyecto de presupuesto y los procedimientos establecidos.

3. Aplicar la estructura orgánica autorizada para el departamento y los procedimientos establecidos

4. Coordinar con las divisiones de estudios profesionales y postgrado e investigación, y la aplicación de los programas de estudio y con el departamento de desarrollo académico los materiales y apoyo didáctico de las asignaturas correspondientes a las áreas de sistemas y computación que se imparten en el Instituto Tecnológico y controlar su desarrollo

5. Coordinar los proyectos de investigación educativa, científica y tecnológica en las áreas de sistemas y computación que se llevan a cabo en el Instituto Tecnológico y controlar su desarrollo

6. Coordinar los proyectos de producción académica y de investigación científica y tecnológica en las áreas de sistemas y computación relacionas con la vinculación del instituto tecnológico con el sector productivo de bienes, servicios de la región y controlar su desarrollo.

7. Proponer a la subdirección académica el desarrollo de cursos y eventos que propicien la superación y actualización profesional del personal docente de las áreas de sistemas y computación en el Instituto Tecnológico.

8. Apoya a la división de estudios profesionales en el proceso de titulación de los alumnos del instituto.

9. Supervisar y evaluar el funcionamiento del departamento y con base en los resultados, proponer las medidas que mejoren su operación.

10. Coordinar las actividades del departamento con las demás áreas den la subdirección académica

11. Presentar reportes periódicos de las actividades desarrolladas a la subdirección académica.

Funciones

Planear, organizar, dirigir, controlar y evaluar de acuerdo con las normas y lineamientos establecidos, las actividades de docencia, investigación y vinculación del instituto tecnológico. Elaborar el programa operativo anual y el anteproyecto propuesto a la subdirección del instituto tecnológico para lo conducente. Coordinar las actividades de la subdirección con las demás áreas para los cumplimientos de los objetivos del instituto tecnológico.

Capítulo 2

Descripción del proyecto

En este capítulo se presenta la descripción del proyecto "Implementación de un algoritmo de sincronización de semáforos usando inteligencia artificial", en donde se da a conocer la razón por la cual se decidió desarrollar la investigación, así como también, los objetivos que se deben cumplir, los problemas que resuelve el mismo, los alcances y limitaciones que se presentan.

2.1. Problemas a resolver

En la ciudad de Tapachula, no existen semáforos inteligentes que sean capaces de resolver los problemas que todos los días se presentan a consecuencia de la gran cantidad de automóviles que circulan en las calles.

Debido a ello en un futuro se pretende la creación de estos semáforos, pero para su desarrollo es necesario el uso de un algoritmo de sincronización que permita la comunicación entre semáforos de avenidas intersectadas y decida el estatus de cada uno de ellos.

2.2. Alcances y limitaciones

Alcances del proyecto

- El algoritmo contará con un tiempo de respuesta en el orden de milisegundos para su implementación en el control de semáforos.

Limitaciones del proyecto

- El algoritmo solo comunicará los semáforos de cuatro avenidas que se intersectan.

2.3. Objetivos

Objetivo general

Implementar un algoritmo de sincronización de semáforos utilizando inteligencia artificial.

Objetivos específicos

- Seleccionar la técnica de inteligencia artificial que resuelva la sincronización entre los semáforos.
- Implementar un algoritmo que use la técnica seleccionada y que determine el estatus de cada semáforo y el tiempo que permanecerá dicho estatus.

2.4. Justificación

Las ciudades son lugares en donde se realiza una alta actividad económica y en donde transportarse forma parte de la vida cotidiana de las personas; mientras que para las empresas representa una parte esencial de su operación diaria. Estos factores hacen que en las calles principales de la ciudad exista tráfico vehicular y aumente la contaminación del medio ambiente, lo cual es perjudicial para las personas y el ambiente mismo.

Tomando en cuenta los puntos antes mencionados, se considera importante, que en la ciudad de Tapachula, Chiapas, se debe de contar con un sistema inteligente que permita controlar de manera eficiente el flujo vehicular y peatonal, para lograr este objetivo será necesario que el sistema sea capaz de realizar el conteo de los automóviles de una forma rápida, logrando con ello tener un control que permita administrar el tiempo de espera para ceder el paso a los automóviles y así mismo a los peatones.

Capítulo 3

Marco teórico

Durante los últimos XX años se ha intentado mitigar el problema del alto congestionamiento vehicular mediante el uso de Semáforos Inteligentes. Las recientes investigaciones arrojan propuestas que hacen uso de técnicas de inteligencia artificial para resolver el problema.

A continuación, se muestra de manera resumida algunos de los trabajos de investigación más recientes, además, se hará una breve introducción a los conceptos necesarios, con el fin de facilitar y maximizar la comprensión del diseño del sistema propuesto para la optimización del tránsito.

3.1. Conceptos generales

La historia de los semáforos inicia con la aparición del automóvil, las carreteras y su inevitable congestionamiento [10].

3.1.1. Primeras carreteras

Desde la antigüedad, la construcción de carreteras ha sido uno de los primeros signos de civilización avanzada. Cuando las ciudades de las primeras civilizaciones empezaron a aumentar su tamaño y densidad poblacional, la comunicación con otras regiones se tornó necesaria para hacer llegar suministros alimenticios o transportarlos a otros consumidores.

Entre los primeros constructores de carreteras se encuentran los mesopotámicos, hacia el año 3500 A.C.; los chinos, que construyeron la Ruta de la Seda (la más larga del mundo) durante 2.000 años y desarrollaron un sistema de carreteras en torno al siglo XI A.C y; los incas de Sudamérica, que construyeron una avanzada red de caminos que no pueden ser considerados estrictamente carreteras, ya que los incas no conocían la rueda, esta red se distribuía por todos los Andes e incluía galerías cortadas en rocas sólidas.

3.1.2. Primeros automóviles

Puede afirmarse que el vehículo de motor de combustión interna en la forma que lo conocemos actualmente, forma parte y nació con el siglo XX. Al iniciar su vida y considerado como un artefacto de lujo y deporte, encontró serios obstáculos por los malos caminos y leyes anacrónicas, además de la natural oposición de las empresas y particulares habituados al ferrocarril y los carruajes tirados por animales, por lo que hubo que esperar para su florecimiento hasta principios del siglo XX.

Los grandes desarrollos en transporte han neutralizado relativamente el *"obstáculo espacio"* con la reducción de distancias expresada en disminución de tiempos de viaje, permitiendo la integración de las distintas zonas y funciones de la ciudad y de esta con áreas adyacentes e incluso distantes, lo cual influyó en la progresiva ampliación de las concentraciones urbanas.

3.1.3. La congestión aparece en escena

Después de la aparición del vehículo automóvil, las carreteras se proyectaban teniendo en cuenta únicamente el movimiento de vehículos aislados, debido a que circulaba un número muy bajo de ellos para entonces y bastaba que cada uno pudiera moverse a una velocidad razonable y segura para que la carretera cumpliera con todos sus objetivos. Pero ya hacia 1920 el número de vehículos en circulación era lo suficientemente elevado como para establecer medidas de regulación que evitasen las dificultades de circulación.

Actualmente el incremento en número y velocidad del tráfico motorizado contribuye a satisfacer los deseos y las necesidades de los habitantes de las ciudades, sin detenerse a analizar

que ese es también el causante de uno de los aspectos más conflictivos del sistema urbano en función a su sostenibilidad: la contaminación ambiental en sus diferentes formas, la ocupación extensiva del suelo y la seguridad del tráfico.

Se hace necesaria entonces la planeación integral del transporte: integración del transporte y los usos del suelo, la cual debe abordar la relación entre movilidad/accesibilidad y los modelos de crecimiento urbano. Por tanto se ve la necesidad de la realización de estudios, procedimientos de aplicación de las diferentes metodologías y desarrollos en este campo cuyo modelo de crecimiento urbano, se manifiesta en la *congestión del tráfico vehicular*.

La congestión vehicular o vial se refiere tanto urbana como interurbanamente, a la condición de un flujo vehicular que se ve saturado debido al exceso de demanda de las vías, produciendo incrementos en los tiempos de viaje y atascamientos. Este fenómeno se produce comúnmente en las horas punta u horas pico, y resultan frustrantes para los automovilistas, ya que resultan en pérdidas de tiempo y consumo excesivo de combustible.

Las consecuencias de las congestiones vehiculares denotan en accidentes, a pesar de que los automóviles no pueden circular a gran velocidad, ya que el automovilista pierde la calma al encontrarse estático por mucho tiempo en un lugar de la vía. Esto también deriva en violencia vial, por otro lado, reduce la gravedad de los accidentes ya que los vehículos no se desplazan a una velocidad importante para ser víctima de daños o lesiones de mayor gravedad. También, los vehículos pierden innecesariamente combustible debido a que se está inactivo por mucho tiempo en un mismo lugar, sin avanzar en el trayecto de un punto a otro.

3.1.4. Semáforos

El primer semáforo de luces de tránsito que se instaló en la historia, fue en el exterior del parlamento británico de Westminster; obra del ingeniero J.P. Knight, especialista en señales de ferrocarril. Este aparato empezó a funcionar el 10 de Diciembre de 1868 e imitaba a las señales de ferrocarril y sólo usaba las luces de gas rojas y verdes por la noche. Dos zumbidos señalaban que el tráfico que podía avanzar era el de la avenida y un sólo zumbido indicaba

que era el tráfico de la calle. No tuvo una larga existencia dado un desafortunado accidente que provocó que explotase matando a un policía.

Debido a la proliferación de coches, el 4 de Agosto de 1914 se instaló el primer semáforo "moderno" en Estados Unidos, inventado por Garrett Augustus Morgan, gestionaba el tráfico entre la avenida Euclid y la calle 105. Contaba con luces rojas y verdes, colocadas sobre unos soportes con forma de brazo. Además incorporaba un emisor de zumbidos como su antecesor inglés. El sistema cambió pocos años después y se sustituyó el zumbador por una tercera luz de color ámbar. Los primeros semáforos de tres luces aparecieron en 1920 en las calles de Detroit, en semáforos de cuatro direcciones y en Nueva York, donde se pusieron a prueba en la Quinta Avenida.

En 1953 aparecieron los primeros semáforos eléctricos. Ocho años más tarde, en 1961 se introdujo en Berlín, el dispositivo regulaba la circulación de los peatones.

Ciclos y fases de un semáforo

El ciclo del semáforo comprende la sucesión cíclica de sus fases. Las longitudes del ciclo y los tiempos en verdes del semáforo pueden ser estimados utilizando las siguientes ecuaciones:

$$C = \frac{LX_c}{x_c - \sum_i (v/s)_a}$$

$$(verde.efect)_i = v_i C / s_i X_i = (v/s)_i (C/X_i)$$

Donde:

C es la longitud del ciclo en seg.

L es el tiempo perdido por ciclo.

X_c es la razón *volumen/capacidad* (v/c) crítica para la intersección.

X_i es la razón v/c para el grupo de carriles i.

$(v/s)_i$ es la proporción *volumen/saturación* para el grupo de carriles i.

$(verde.efect)$ es el verde efectivo para el grupo de carriles i, en segundos.

La longitud de ciclo que produce las demoras mínimas para la inserción se le conoce como **longitud del ciclo óptimo**, el cual se calcula utilizando la siguiente ecuación:

$$C_0 = \frac{1.5L + 5}{1 - \sum_i^n (v/s)_a}$$

Donde:

C_0 es la longitud del ciclo óptimo en seg.

L es el tiempo total perdido por ciclo.

n es el número de fases.

$(v/s)_i$ es la proporción *volumen/saturación* para el grupo de carriles i.

Por lo general, el ciclo óptimo para una intersección en particular se encuentra entre los siguientes límites:

$$0.75C_0 \leq C_0 \leq 1.50C_0$$

Las longitudes de ciclo deben estar entre 40 segundos y 120 segundos. Longitudes de ciclo fuera de estos valores son muy cortas o muy largas.

Las fases son la combinación de movimientos que operan simultáneamente, es decir, es la parte de un ciclo de un semáforo durante la cual uno o más movimientos reciben derecho de vía. Las fases se delimitarán en la vía cuando haya un cambio de derecho de paso, o sea, cuando un movimiento vehicular o peatonal es detenido y otro inicia, hay cambio de fase. El número de fases de un semáforo depende de la complejidad de la intersección. El número de fases tiene un rango que varía entre dos fases (el más simple) hasta ocho fases (el más complicado). La eficiencia de una intersección semaforizada decrece cuando el número de fases se aumenta.

En los arreglos de las fases para un semáforo se deben tener las siguientes consideraciones:

- El volumen del movimiento a la izquierda.
- El volumen del movimiento de frente que es opuesto al de *vuelta a la izquierda*.

- Accidentes.
- La disponibilidad de carriles exclusivos adecuados para vueltas a la izquierda.
- La operación del sistema, la forma en que los arreglos de las fases se relacionan con la operación coordinada con otras intersecciones semaforizadas.
- Actividad de peatones.

Recomendaciones para las fases:

- Usar el número mínimo de fases para cumplir con las necesidades del tráfico.
- Los ciclos prácticos están entre 40 seg. y 120 seg. Sin embargo, nunca exceder 180 seg. bajo condiciones de saturación ni 90 seg. con flujos bajos.
- Mantener el verde sin uso en un mínimo.

3.2. Conceptos técnicos

3.2.1. Inteligencia artificial

Lo que hoy se conoce como IA empezó hacia 1960 cuando en el Instituto Tecnológico de Massachusetts (MIT, por sus siglas en inglés), John McCarthy creó el LISP (el primer lenguaje de investigación dentro de la IA). Sin embargo, el término IA suele atribuírsele a Marvin Minsky, también del MIT, quien en 1961 escribió un artículo titulado "Hacia la Inteligencia Artificial" como miembro de la IRE (Institute of Radio Engineers).

Los años sesenta del siglo pasado fueron un intenso periodo de optimismo hacia la posibilidad de hacer que una computadora pensase. Después de todo, esos años contemplaron la primera computadora que jugaba ajedrez, las primeras pruebas matemáticas informatizadas, y el ya famoso e igualmente bien conocido Programa ELIZA que fue escrito en el MIT por Joseph Weisenbaum en 1964. El programa ELIZA actuaba como un psicoanalizador. En este tipo de análisis, el psiquiatra toma un papel pasivo generalmente repitiendo las propias declaraciones del paciente, en vez de llevar el peso de la conversación. Posteriormente, en la década de los años setenta se creó el PROLOG, obra de Alain Colmerauer, en Masella, Francia, en 1972.

PROLOG era un lenguaje diseñado para ayudar a resolver problemas relativos a la IA. Este lenguaje poseía un gran número de características especiales tales como una base de datos incorporada y una sintaxis bastante simple (Schildt, 1990).

A lo largo de la historia se han adoptado cuatro enfoques:

1. **Actuar como humano: el enfoque de la prueba de Turing.** Mediante la prueba de Turing, propuesta por Alan Turing (1950), se intenta ofrecer una satisfactoria definición operativa de lo que es la inteligencia. Turing definió una conducta inteligente como la capacidad de lograr eficiencia a nivel humano en todas las actividades de tipo cognoscitivo, suficiente para engañar a un elevador.

2. **Pensar como humano: el enfoque del modelo cognoscitivo.** Para poder afirmar que un programa determinado utiliza algún tipo de razonamiento humano, previamente habrá que definir cómo piensan los seres humanos. Habrá que penetrar en el funcionamiento de la mente humana.

3. **Pensar racionalmente: el enfoque de las leyes del pensamiento.** El filósofo griego Aristóteles fue uno de los primeros en intentar codificar "la manera correcta de pensar", es decir, establecer procesos de pensamiento irrefutables. Sus famosos silogismos son esquemas de estructuras de argumentación mediante las cuales siempre se llega a conclusiones correctas si se parte de premisas correctas. Este enfoque presenta dos obstáculos. En primer lugar, no es fácil recibir un conocimiento informal y expresarlo en los términos formales que exige la notación lógica, especialmente cuando el conocimiento tiene menos de 100 % de certidumbre.

4. **Actuar en forma racional: el enfoque del agente racional.** Actuar racionalmente implica actuar de manera tal que se logren los objetivos deseados con base en ciertos supuestos. Un agente es algo capaz de percibir y actuar. De acuerdo con este enfoque, se considera la IA como el estudio y construcción de agentes racionales.

3.2.2. Principales ramas de la I.A.

A continuación se ofrece un resumen de las principales ramas de la Inteligencia Artificial, a saber: Redes Neuronales, Algoritmos Genéticos, Sistemas Expertos; La Lógica Difusa, que también es una rama de la IA, será tratada con más detalle en la siguiente sección (sección 3.2.3).

Redes neuronales

El aparato de comunicación neuronal de los animales y del ser humano, está formado por el sistema nervioso y hormonal. Su misión es recoger informaciones, transmitirlas y elaborarlas, en parte también almacenarlas y enviarlas de nuevo en forma elaborada.

El elemento estructural y funcional más esencial, en el sistema de comunicación neuronal, es la célula nerviosa o neurona. La mayoría de las neuronas utilizas sus productos de secreción como señales químicas (transmisores) para la transmisión de información. Dicha información se envía, entre las distintas neuronas, a través de prolongaciones, formando redes en las cuales se elabora y almacena información.

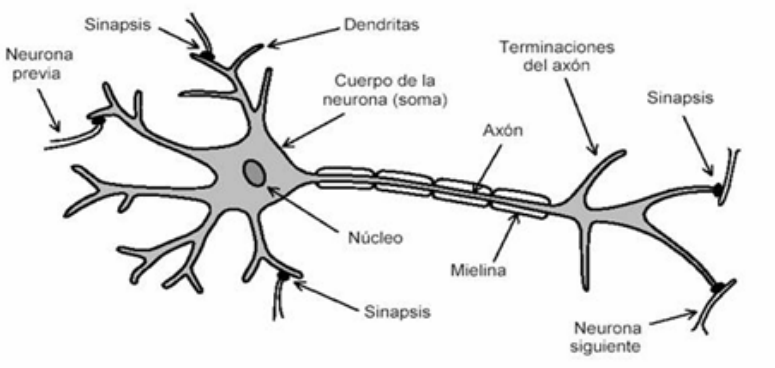


Figura 3.1: Neurona biológica

La misión de las neuronas comprende generalmente cinco funciones parciales:

- Las neuronas recogen la información que llega a ellas en forma de impulsos procedentes de otras neuronas o de receptores.
- La integran en un código de activación propio de la célula.
- La transmiten codificada en forma de frecuencia de impulsos a través de su axón.
- A través de sus ramificaciones el axón efectúa la distribución espacia de los mensajes.

Una Red de Neuronas Artificiales (en adelante, RNA) es un paradigma de procesamiento de información inicialmente inspirado en el modo en el que lo hace el cerebro. El elemento clave de este paradigma es su estructura. Las RNA están compuestas por un cierto número de elementos de procesamiento o neuronas que trabajan al unísono para resolver un problema específico.

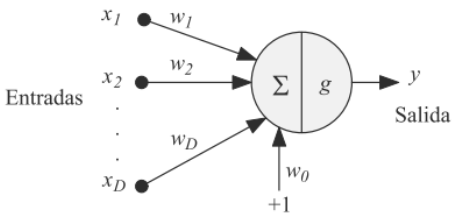


Figura 3.2: Modelo matemático de una neurona

Las redes neuronales actuales se basan en el modelo matemático de neurona propuesto por *McCulloch* y *Pitts* en 1943. En dicho modelo (véase Figura 3.2) cada neurona recibe un conjunto de entradas $x1, x2, ..., xD$ y devuelve una única salida . Además, dentro de una RNA existen numerosas conexiones entre las distintas neuronas que la forman. Estas conexiones simulan las conexiones neuronales del cerebro y, al igual que éstas, pueden establecerse con mayor o menor intensidad. En el caso de las RNA esta intensidad la determinan los pesos sinápticos (o simplemente pesos). De este modo, cada entrada x_i de una neurona se encuentra afectada por un peso w_i.

El primer paso para obtener la salida de la neurona es calcular la suma ponderada a de las entradas, llamada activación de la neurona:

$$a = \sum_{i=1}^{D} w_i x_i + w_0$$

Donde w0 es un umbral o sesgo que se utiliza para compensar la diferencia entre el valor medio de las entradas, sobre todo el conjunto de entrenamiento, y el correspondiente valor medio de las salidas deseadas. Posteriormente, a partir de este valor a se obtiene la salida y de la neurona mediante la aplicación de una función, llamada función de activación o de transferencia g(a), es decir:

$$y = g(a) = g\left(\sum_{i=1}^{D} w_i x_i + w_0\right) = g\left(\sum_{i=0}^{D} w_i x_i\right)$$

Donde, como se observa, es posible tratar el umbral w_0 como un peso más si se supone una entrada añadida x_0 con un valor fijo de 1. Finalmente, también es posible reescribir esta

ecuación en notación vectorial como $g(a) = g(w^T x)$, si tomamos w como el vector de pesos y x como el vector de entradas a la red. La función de transferencia empleada en este modelo básico de *McCulloch-Pitts* es la función escalón definida por la ecuación:

$$g(a) = \begin{cases} 0 \text{ cuando } x < 0 \\ 1 \text{ cuando } a > 0 \end{cases}$$

En los modelos actuales se escogen otro tipo de funciones, normalmente monótonas y derivables. A continuación se presentarán algunas de ellas.

- lineal $g(a) = a$
- sigmoidal $g(a) = \dfrac{1}{1 + e^{-a}}$
- tangente hiperbólica $g(a) = \dfrac{e^a - e^{-a}}{e^a + e^{-a}}$
- gaussiana $g(a) = e\left(-\dfrac{(a-\mu)^2}{2\sigma^2}\right)$

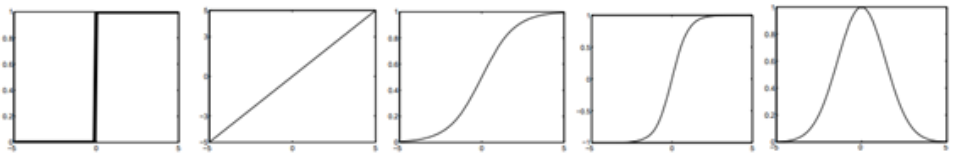


Figura 3.3: Funciones de activación

Algoritmos genéticos

Los Algoritmos Genéticos (AG) fueron introducidos inicialmente por Holland en 1975 para abstraer y explicar rigurosamente los procesos adaptativos de los sistemas naturales, así como para el diseño de sistemas artificiales de software que retengan los mecanismos importantes de los sistemas naturales. Fue unos años más tarde cuando su alumno, D. Goldberg, implementó el primer AG aplicado en problemas industriales. Estas y otras aplicaciones creadas por estudiantes de Holland convirtieron los AGs en un campo suficientemente aceptado.

En un AG, las soluciones potenciales al problema se representan normalmente mediante cadenas binarias de bits (0's y 1's) de una longitud determinada *long* que vendrá impuesta por el número de variables existentes en la solución y por el número de bits necesarios para codificarlas. Otros términos usados a menudo para denominar una solución del problema en un AG son string o estructura, y, siguiendo el vocabulario de los sistemas biológicos, cromosoma.

Así, los cromosomas están compuestos por unidades binarias que se denominan genes. Al valor de un gen determinado se le denomina alelo, y a su posición en el cromosoma locus. Al paquete genético total se le denomina genotipo, y a la interacción del genotipo con su entorno se le denomina fenotipo, que se traduce en la decodificación del cromosoma para la obtención de una solución alternativa (conjunto de parámetros particulares, o un punto en el espacio de búsqueda). De esta forma, podemos representar un cromosoma c_i^t en una generación (iteración) determinada t como:

$$c_i^t = (b_{i_1}^t \dots b_{i_l ong}^t)$$

Con $b_{i_j}^t \in \{0, 1\}$, $j = 1,\dots,$ *long*. El término individuo es frecuentemente utilizado para referirse al conjunto de información genotipo-fenotipo adecuación. Así, podemos representar un individuo X_i^t en una generación t, como la terna:

$$X_i^t = (c_i^t, x_i^t, f_i^t)$$

Donde X_i^t es la decodificación (fenotipo) del cromosoma c_i^t, y f_i^t es la adecuación de la solución al entorno o *fitness*.

Sistemas expertos

En la década de los setenta se inicia la explosión de aplicaciones de la IA en términos de sistemas basados en reglas a los que se les llama primero "Sistemas Expertos" y después "Sistemas Basados en Conocimiento". El reconocimiento de la insuficiencia de la lógica como herramienta única de representación da lugar al desarrollo de otras formas de representación e inferencia mediante redes causales y asociativas (semánticas, neuronales y bayesianas) y marcos, objetos y agentes.

El área de sistemas expertos es una aproximación muy exitosa a la solución de problemas clásicos de AI en la programación de inteligencia. El profesor Edward Feigenbaum de la Universidad de Stanford, pionero en la tecnología de los sistemas expertos, los ha definido como "un programa de computación inteligente que usa el conocimiento y procedimientos de inferencia para resolver problemas que son lo suficientemente difícil como para requerir significativa experiencia humana para su solución, es decir, un sistema experto es un sistema de cómputo que emula la habilidad de tomar decisiones de un especialista humano.

El conocimiento de un sistema experto puede representarse de varias maneras (puede estar encapsulado en reglas y objetos). Un método común de representar el conocimiento es en forma de reglas tipo SI...ENTONCES, como:

SI la luz es roja ENTONCES deténgase.

Aunque se trata de un ejemplo muy simple, se han construido muchos sistemas expertos significativos expresando en reglas el conocimiento de especialistas.

Los sistemas expertos se han aplicado casi a todos los campos del conocimiento, debido a eso a continuación se mencionarán algunos:

Nombre del S.E.	Área	Descripción
SPEX	Química	Planear experimentos de biología molecular.
PUFF	Medicina	Diagnosticar enfermedades de los pulmones.
LITHO	Geología	Interpretar los datos de registro de pozos petroleros.
TIMM	Informática	Diagnosticar computadoras DEC.

Cuadro 3.1: Ejemplos de Sistemas Expertos

3.2.3. Lógica difusa

Hace más de 50 años, en 1965 Lotfi A. Zadeh, en aquel entonces director del Departamento de Ingeniería Eléctrica de la Universidad de California en Berkeley, publicó *Fuzzy Sets*. Este artículo describe las matemáticas de los conjuntos difusos y por extensión de la lógica difusa, y este trabajo le dio nombre a su campo. Zadeh aplicó la lógica de Lukasiewicz a cada objeto en

un conjunto y creó un álgebra completa para conjuntos difusos. Esta teoría propone *funciones de pertenencia* (o los valores falso y verdadero) sobre el rango [0.0, 1.0].

El ser humano muestra dificultad para tomar decisiones cuando se tiene información imprecisa. La lógica difusa fue creada para emular la lógica humana y tomar decisiones acertadas a pesar de la información. Es una herramienta flexible que se basa en reglas lingüísticas dictadas por expertos. Por ejemplo, la velocidad de un automóvil es una variable que puede tomar distintos valores lingüísticos, como "alta", "media" o "baja". Estas variables lingüísticas están regidas por reglas que dictan la salida del sistema. En otras palabras, la lógica difusa es un conjunto de principios matemáticos basados en *grados de membresía o pertenencia*, cuya función es modelar información. Este modelado se hace con base en reglas lingüísticas que aproximan una función mediante la relación de entradas y salidas del sistema (composición). Esta lógica presenta rangos de membresía dentro de un intervalo entre 0 y 1, a diferencia de la lógica convencional, en la que el rango se limita a dos valores: el cero o el uno.

3.2.4. Conjuntos difusos

El concepto de conjunto difuso fue propuesto por Zadeh(1965): "Un conjunto difuso es una colección de objetos con un *grado de membresía* continuo. Un conjunto tal, es caracterizado por una *función de membresía* que asigna a cada objeto un grado de membresía en un rango de 0 a 1." (p.338).

En teoría clásica de conjuntos, un conjunto tiene unos límites *nítidos* bien definidos (límites *crisp*). Por ejemplo, el conjunto A de los números más grandes que 8 se representa como:

$$A = \{x \mid x > 8\}$$

Sin embargo, los conceptos manejados por el ser humano como *frío* y *caliente*, tienen una transición gradual. Así, un conjunto difuso, lidia con la vaguedad inherente de estos conceptos, conteniendo los elementos sólo con un cierto grado de pertenencia.

En teoría de conjuntos difusos, los conceptos se asocian a conjuntos difusos (asociando sus *valores de pertenencia*) en un proceso llamado *fuzzificación*. Una vez que tenemos los valores *fuzzificados* podemos trabajar con *reglas lingüísticas* y obtener una salida, que podrá seguir siendo difusa o *defuzzificada* para obtener un valor discreto.

Sea X el *Universo del discurso*, y sus elementos se denotan como x; En la teoría clásica de conjuntos se define un conjunto C sobre X mediante la *función característica* de C como f_c .

$$f_c(x) = \begin{cases} 1 \text{ cuando } x \in C \\ 0 \text{ cuando } x \notin C \end{cases}$$

Este conjunto mapea el universo X en un conjunto de dos elementos, donde la función $fc(x)$ es 1 si el elemento x pertenece al conjunto C y 0 si el elemento x no pertenece al conjunto C.

Si generalizamos esta función para que los valores asignados a los elementos del conjunto caigan en un rango particular y así indicar el grado de pertenencia de los elementos a ese conjunto, tendremos una **función de pertenencia** de un determinado conjunto difuso. La función de pertenencia μ_A por la que se define un conjunto difuso A, Sería:

$$\mu_A = X \to [0, 1]$$

Donde $\mu_A(x) = 1$ si x está totalmente en A, $\mu_A(x) = 0$ si x no está en A y $0 < \mu_A(x) < 1$ si x está parcialmente en A. Este valor entre 0 y 1 representa el grado de pertenencia μ al conjunto A.

Definición formal

Un conjunto difuso A definido en el universo de discurso U es caracterizado por una función de membresía $\mu_A : U \to [0, 1]$ que asocia a cada elemento u de U un valor $\mu_A(u)$ en el intervalo [0,1], con $\mu_A(u)$ representando el grado de membresía de u en A.
Algunas características:

El soporte de A son los puntos en U en los cuales $\mu_A(u)$ es positivo.

La altura de A es el valor máximo de $\mu_A(u)$ sobre U.

El punto de cruce de A es un punto en U donde el valor de membresía en A es 0.5

Principio isomorfo

Es bien conocido que la teoría de conjuntos, el *Álgebra booleana* y la Lógica tradicional son isomorfas, bajo transformaciones adecuadas. Esto significa que tienen una estructura subyacente similar, y que por tanto, las definiciones que se hagan en una de las tres teorías se pueden llevar a las otras dos, mediante las transformaciones adecuadas. En el siguiente cuadro se muestra la correspondencia de algunos operadores.

Teoría de conjuntos	Álgebra booleana	Lógica tradicional
Intersección	Conjunción	AND
Unión	Disyunción	OR
Complemento	Negación	NOT

Cuadro 3.2: Correspondencia entre operadores

Ahora bien, el razonamiento lógico consiste en la combinación de proposiciones para producir nuevas proposiciones; así, la combinación de las proposiciones *X es A* y *Y es B* mediante el operador *AND* da como resultado la proposición *X es A AND Y es B*. El cuadro 3.2 sugiere que puede representarse esta combinación mediante un operador análogo a la intersección de conjuntos.

Lo anterior es posible porque en lógica tradicional toda proposición puede tener uno de dos valores, *verdadero* o *falso*, lo que corresponde en la teoría de conjuntos discretos a los únicos dos valores que puede tomar la función de pertenencia para cualquier conjunto: 1 ó 0.

Ahora bien, en lógica difusa una proposición puede representarse por un conjunto difuso *X es A* corresponde a un conjunto A con función de pertenencia $\mu_A(x)$, mientras *Y es B* corresponde a un conjunto B con función de pertenencia $\mu_B(x)$, y la combinación de estas dos proposiciones con el operador *AND*, es decir, la proposición *X es A AND Y es B* corresponde

a un nuevo conjunto difuso $A\ AND\ B$ con función de pertenencia:

$$\mu_{A\cap B}(x, y) = \text{mín}(\mu_A(x), \mu_B(y))$$

En donde se ha utilizado el operador mín parara efectuar la intersección de los dos conjuntos, pero en general podría haberse utilizado cualquier norma T.

Nótese que los universos de discurso sobre los cuales están definidos los conjuntos A y B no son necesariamente el mismo; son, por ejemplo U y V respectivamente, mientras el conjunto $A \cap B$ está definido sobre el universo $U \times V$.

En forma análoga, al operador lógico OR puede hacerse corresponder a una *norma S*, mientras al operador lógico NOT puede hacerse corresponder el complemento.

3.2.5. Operaciones de conjuntos difusos

Las tres operaciones básicas entre conjuntos discretos: *unión, intersección y complemento*, se definen también para los conjuntos difusos, intentando mantener el significado de tales operaciones. La definición de estas operaciones se hace empleando el concepto de función de pertenencia de los conjuntos.

Intersección

Dado dos conjuntos difusos A y B con funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$, respectivamente. La intersección $A \cap B$ puede representarse en general como una función $T : [0, 1] \times [0, 1] \rightarrow [0, 1]$ o como un operador binario $\triangle$, tal que:

$$\mu_{A\cap B}(x, y) = T[\mu_A(x), \mu_B(y)] = \mu_A(x) \triangle \mu_B(y)$$

Donde T, debe satisfacer las siguientes propiedades:

1. *Elemento unidad:* $T(a, 1) = a$ y $T(a, 1) = T(1, a) = a$

2. *Conmutatividad:* $T(a,b) = T(b,a)$

3. *Monotonicidad:* Si $a \leq c$ y $b \leq d$ entonces $T(a,b) = T(c,d)$

4. *Asociatividad:* $T(T(a,b),c) = T(a,T(b,c))$

Todo operador que satisfaga las propiedades anteriores se conoce como una **Norma Triangular** o **Norma T** y representa la intersección de dos conjuntos difusos. Algunas *normas T* ampliamente utilizadas son:

- *Mínimo:* $T_{min}(a,b) = min(a,b)$
- *Producto algebraico:* $T_{ap}(a,b) = ab$
- *Diferencia limitada (o de Lukasiewick)*: $T_{bp}(a,b) = max(0, a+b-1)$

Unión

Dado dos conjuntos difusos A y B con funciones de pertenencia $\mu_A(x)$ y $\mu_B(x)$, respectivamente. La unión $A \cup B$ puede representarse en general como una función $S : [0,1] + [0,1] \rightarrow [0,1]$ o como un operador binario $\perp$, tal que:

$$\mu_{A \cup B}(x,y) = S\left[\mu_A(x), \mu_B(y)\right] = \mu_A(x) \perp \mu_B(y)$$

Donde S, debe satisfacer las siguientes propiedades:

1. *Elemento Neutro:* $S(1,1) = 1$ y $S(a,0) = S(0,a) = a$

2. *Conmutatividad:* $S(a,b) = S(b,a)$

3. *Monotonicidad:* Si $a \leq c$ y $b \leq d$ entonces $S(a,b) = S(c,d)$

4. *Asociatividad:* $S(S(a,b),c) = S(a,S(b,c))$

Todo operador que satisfaga las propiedades anteriores se conoce como una **Conorma T** o **Norma S** y representa la unión de dos conjuntos difusos. Algunas *normas S* ampliamente utilizadas son:

- *Máximo:* $S_{max}(a, b) = max(a, b)$
- *Producto:* $S_{as}(a, b) = (a + b) - (a \times b)$
- *Suma limitada (o de Lukasiewick)*: $S_{bs}(a, b) = min(a + b, 1)$

Complemento

Dado un conjunto A, con función de pertenencia $\mu_A(x)$. El complemento $\neg A$ puede generalizarse considerándolo como una función $N : [0, 1] \to [0, 1]$, tal que:

$$\mu_{\neg A}(x) = N(\mu_A(x))$$

La operación de complemento, a la que llamaremos **Norma N**, también debe cumplir ciertas propiedades:

1. *Condición limite o frontera:* $N(0) = 1$ y $N(1) = 0$.
2. *Monotonicidad:* si $a \leq b$ entonces $N(a) \geq N(b)$.
3. *Continuidad:* la función de complemento $N(a)$ es continua.
4. *Involutividad:* $N(N(a)) = a$

Al igual que con la unión y la intersección, también para el complemento, existen gran variedad de clases. Algunas *normas N* más utilizadas son:

1. *Clásico:* $N_c(a) = 1 - a$
2. *Sugeno:* $N_s(a) = \frac{1-a}{1+sa}$ con $s \in (-1, \infty)$
3. *Yager:* $N_w(a) = (1 - a^w)^{1/w}$ con $w \in (0, \infty)$

3.2.6. Funciones de membresía

Para definir un conjunto difuso A, hay que definir su *función de membresía* o *función de pertenencia*. Supongamos una función de pertenencia: $\mu_A(x)$, la imagen de la función es la curva que define el grado de pertenencia de cada elemento x al conjunto A.

Para la correcta representación de los grados de pertenencia de cada uno de los elementos que conforman el conjunto difuso, lo más natural es extraer los datos de los fenómenos que se va a representar y con ellos, elegir y ajustar una función de membresía adecuada. De otra manera existen metodologías que permiten asignar grados de membresía a cada uno de los elementos del conjunto.

Algunas de las funciones de membresía básicas son las siguientes.

Triangular. La imagen de la función, como su nombre lo indica, asemeja un triángulo. Realmente la función está compuesta por dos líneas rectas, una con pendiente positiva hasta alcanzar la unidad, y otra con pendiente negativa, hasta llegar a 0. Está definida por tres parámetros a, b, c donde $a \leq b \leq c$. Estos parámetros definen los vértices de la función.

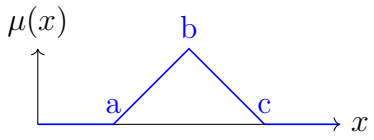


Figura 3.4: Función triangular

Su definición matemática es la siguiente:

$$f(x;a,b,c) = \begin{cases} 0 & \text{si} \quad x \leq a \\ \dfrac{x-a}{b-a} & \text{si} \quad a \leq x \leq b \\ \dfrac{c-x}{c-b} & \text{si} \quad b \leq x \leq c \\ 0 & \text{si} \quad x \geq c \end{cases}$$

La función triangular es adecuada para definir situaciones en las que se tiene un valor óptimo central, el cual se va perdiendo conforme se aleja de él. Un ejemplo de esta situación es la temperatura corporal, esta tiene un valor óptimo de 36 °C, pero por debajo de 35 °C o por encima de 37 °C se podría considerar peligrosa.

Trapezoidal. Una generalización de la función triangular es la función trapezoidal, que a diferencia de la triangular, tiene un núcleo más amplio. Esto es, el intervalo donde el valor de membresía es igual a 1 se extiende entre los vértices b y c. La función está definida por cuatro parámetros a, b, c, d donde $a \leq b \leq c \leq d$. En la figura 3.5 se aprecia el trapecio que forman estos cuatro vértices.

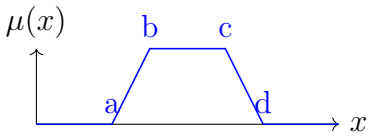


Figura 3.5: Función triangular

Su definición matemática es la siguiente:

$$f(x;a,b,c,d) = \begin{cases} 0 & \text{si} \quad x \leq a \\ \dfrac{x-a}{b-a} & \text{si} \quad a \leq x \leq b \\ 1 & \text{si} \quad b \leq x \leq c \\ \dfrac{c-x}{c-b} & \text{si} \quad c \leq x \leq d \\ 0 & \text{si} \quad x \geq d \end{cases}$$

La forma de esta función es utilizada cuando hay un rango de valores óptimos. Un ejemplo de esto es la temperatura ambiente. Hay un rango de temperaturas que podemos considerar adecuadas, pero por debajo de este rango las personas comienzan a sentir frío, y por encima de él se consideraría un ambiente caluroso.

Gaussiana. La *función gaussiana* es una función simétrica que juega el papel de la función triangular, solo que esta pertenece al grupo de las funciones con derivada continua. Si observamos la figura 3.6 podemos ver como la gráfica asemeja un triángulo suavizado, es decir, los vértices no tienen cambios abruptos sino graduales.

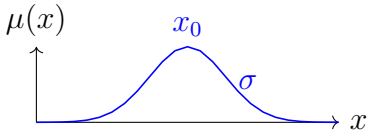


Figura 3.6: Función gaussiana

Su definición matemática es la siguiente:

$$f(x; a, x_0) = e^{-\frac{1}{2}\left(\frac{x-x_0}{a}\right)^2}$$

Donde

x_0 determina el centro o punto de cruce y,

a determina el ancho o la pendiente.

Campana generalizada. Esta función al igual que la anterior, es una función simétrica con derivada continua. Juega el papel de una función trapezoidal, si observamos la figura 3.7, podemos ver como la gráfica asemeja un trapecio suavizado, es decir, los vértices no tienen cambios abruptos sino graduales.

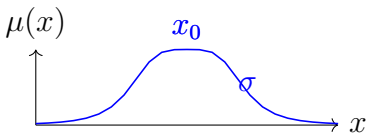


Figura 3.7: Función Campana generalizada

Su definición matemática es la siguiente:

$$f(x; a, b, x_0) = \frac{1}{1 + \left(\frac{x-x_0}{a}\right)^{2b}}$$

Donde

x_0 determina el centro o punto de cruce,

a determina el ancho y,

b determina la pendiente.

Sigmoidal. Una sigmoidal es una función con derivada continua y abierta. Su gráfica recuerda la forma de un escalón. Tiene un parámetro a que determina su pendiente, es decir, determina la suavidad de la transición entre los valores de membresía 0 a 1. Además el parámetro a también determina si la función

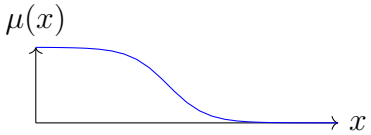


Figura 3.8: Función Sigmoidal

abrirá por la izquierda o por la derecha. Un valor positivo de a
genera un gráfica abierta por la derecha y un valor negativo genera que l gráfica se abra por la izquierda. En la imagen 3.8 podemos observar una función sigmoidal abierta por la izquierda.

Su definición matemática es la siguiente:

$$f(x; a, x_0) = \frac{1}{1 + e^{-a(x-x_0)}}$$

Donde

x_0 determina el centro o punto de cruce y,

a determina la pendiente.

3.2.7. Variables lingüísticas

Una variable lingüística adopta términos lingüísticos que permiten describir el estado de un objeto o fenómeno usando un lenguaje natural(o artificial); estos términos se pueden representar mediante conjuntos difusos. Una variable numérica toma valores numéricos, por ejemplo: *carros = 2*, mientras que una variable lingüística toma valores lingüísticos: carros es *"pocos"*.

De manera más formal: Los valores de una variable lingüística X son las etiquetas $T(x)$ de los subconjuntos difusos en U(conjunto universo). Las etiquetas pueden tener la forma de frases o sentencias en lenguaje natural o artificial. Por ejemplo, si U es la colección de enteros correspondiente a las edades de un grupo de personas:

$$U = 10 + 1 + 2 + ... + 70$$

y la edad es una variable lingüística llamada x definida en U, entonces los valores de x podrían ser *niño, joven, adulto joven, adulto, anciano*, etc. Donde los posibles valores son las etiquetas de los conjuntos difusos que caracterizan las edades (véase la fig. 3.9).

El significado de un término lingüístico $T(x)$ es caracterizado mediante una *función de compatibilidad o membresía*: $C : U \to [0, 1]$, la cual, asocia la compatibilidad (o grado de per-

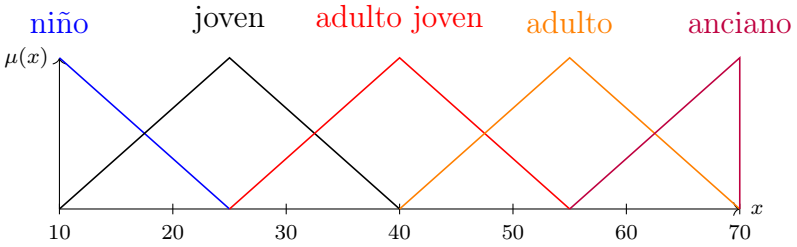


Figura 3.9: Variable lingüística *edad* llamada x

tenencia) de cada u en U, con $T(x)$. Así, la compatibilidad de una edad de *27 años* con el término lingüístico *joven* podría ser 0.7, mientras que una edad de *35 años* podría ser de 0.2.

El concepto de variable lingüística proporciona un medio para definir de manera aproximada los fenómenos que son demasiado complejos o muy imprecisos como para ser receptivos de una descripción convencional en términos cuantitativos.

Una variable lingüística está caracterizada mediante la quíntupla:

$$(X, T(x), U, G, M)$$

Donde

X es el nombre de la variable.

$T(X)$ es el conjunto de términos o valores lingüísticos.

U es el universo de discurso.

G es la regla sintáctica que genera los términos en $T(x)$; y,

M es una regla semántica que socia cada valor lingüístico X con su correspondiente significado $M(X)$, siendo $M(X)$ un conjunto difuso en X.

Hay varios aspectos básicos que es necesario tomar en cuenta al momento de definir de una variable lingüística.

Primero, Es importante entender que la noción de membresía es distinta a la de probabilidad. Así, la declaración de que la membresía de la edad de *28 años* al conjunto difuso *joven* es de 0.7, no tiene relación con la probabilidad de que la edad sea o no 28. La interpretación

correcta es que el valor de membresía 0.7, es simplemente una indicación subjetiva de *qué tanto encaja* la edad de "28 años" (*en una escala de 0 a 1*) con el concepto "joven".

Segundo, normalmente asumiremos que una variable lingüística está *estructurada* tomando en cuenta dos reglas:

Regla (i). La regla sintáctica, esta especifica la manera en que se generan los términos lingüísticos de la variable. En relación a esta regla, normalmente asumiremos que los términos de la variable son generados por una gramática de contexto libre.

Regla(ii). La regla semántica, que especifica el proceso para computar el significado de cualquier término lingüístico dado. Aquí, observamos que un valor típico de una variable lingüística involucra lo que llamamos términos primarios (joven y viejo) que son a su vez, subjetivos y dependientes de contexto. Por lo que se asume que el significado de dichos términos es conocido *a priori*.

En resumen, al momento de definir una variable lingüística $(X, T(x), U, G, M)$, se asume que G es una gramática libre de contexto, además que, cualquier significado $M(x)$ generado por M es subjetivo y conocido *a priori*. Por lo que normalmente M y G se omitirán en la definición.

Tomando en cuenta lo anterior, una variable lingüística se define básicamente por la siguiente tripla.

$$(X, T(x), U)$$

Donde:

X es el nombre de la variable.

$T(x)$ es el conjunto de términos o valores lingüísticos.

U es el universo de discurso.

3.2.8. Sistema difuso

Un sistema de inferencia difuso (*FIS*) consta, conceptualmente, de tres etapas. En la primer etapa se transforman (fuzzifican) las variables de entrada obteniendo valores de difusos. Posteriormente dichos valores son manejados por un sistema de inferencia que genera una salida, también difusa, a partir de las reglas establecidas y los propios valores de entrada. Finalmente el resultado pasa por un proceso llamado defuzzificación, a través del cual, se obtiene la salida real del sistema ya en valores concretos. La figura 3.10 muestra el diagrama esquemático del controlador difuso.

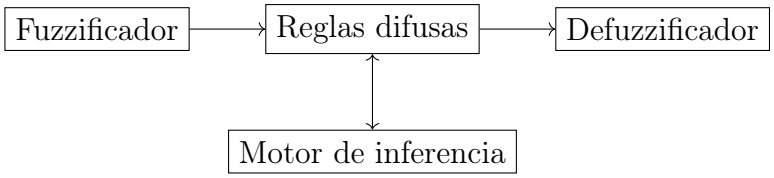


Figura 3.10: Diagrama esquemático de un sistema de inferencia difuso

Fuzzificación

La primera etapa se basa en un proceso donde las variables tienen un grado de incertidumbre metalingüístico. Por lo tanto, el rango de valores (*universo de discurso*) de cada variable puede clasificarse por conjuntos difusos[1], por ejemplo *baja, media, alta*. Cuando el sistema obtiene variables, pasan por un proceso de *fuzzificación* que consiste en pasar dichos valores a un rango de pertenencia entre cero (0) y uno (1). Se busca determinar el grado de pertenencia del valor de entrada a un conjunto difuso. Los conjuntos difusos son caracterizados mediante funciones de membresía ajustadas a las necesidades del sistema. La imagen 3.11 muestra la interpretación grafica de la fuzzificación.

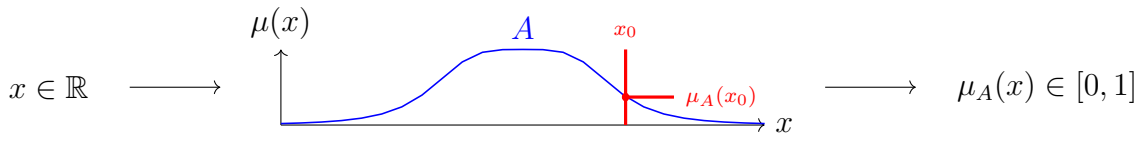


Figura 3.11: Fuzzificación de un valor concreto

[1]Para más información véase la sección 3.2.6 *Funciones de membresía*.

Reglas difusas

En lógica difusa las proposiciones del tipo "IF *temperatura* es *alta* THEN *ventilador* es *máximo*" son usadas para modelar el conocimiento del experto sobre un fenómeno en un lenguaje natural y cotidiano. A estas proposiciones que operan sobre conjuntos difusos se les conoce como reglas difusas.

El objetivo de las reglas difusas es capturar el conocimiento empírico del experto y al conjunto de reglas se le denomina Base de Conocimientos.

Supongamos una proposición sencilla:

$$\text{IF } x \text{ es } a \text{ THEN } y \text{ es } b.$$

En los sistemas de reglas clásicos, si el antecedente es cierto, el consecuente también lo es. En un sistema difuso donde el antecedente es difuso, todas las reglas se ejecutan parcialmente, el consecuente es cierto, pero solo en cierto grado.

Inferencia

En la segunda etapa se proponen reglas lingüísticas (*inferencia*) que servirán para inferir la salida del sistema. El grado de pertenencia de cada una de las variables se evalúa en un conjunto de reglas de inferencia. Dichas reglas de inferencia fueron determinadas con ayuda de un experto. El conjunto de reglas de inferencia determina una consecuencia, es decir, asigna un grado de pertenencia a un conjunto difuso que caracteriza a las salidas.

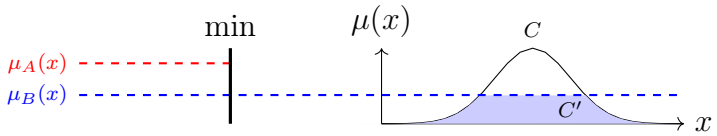


Figura 3.12: Inferencia del conjunto resultado C

En la imagen 3.11 se aprecia gráficamente un ejemplo de un proceso de inferenciación, a saber, de tipo mamdani. Existen varios mecanismos de inferencia, la elección de uno de ellos dependerá de la situación. Algunos de los métodos de inferencia más usados son:

Mamdani.

Sugeno.

Tsukamoto.

Defuzzificación

Una vez obtenidas las consecuencias, la tercera etapa es un proceso para determinar los valores óptimos de salida, conocido como defuzzificación, y que consiste en pasar el grado de pertenencia, proveniente de la consecuencia de las regla de inferencia (*conjunto consecuencia*), a un valor nítido o real. Para hacer eso, previamente se sintonizaron funciones de membresía de cada una de las salidas con el fin de obtener un valor cuantificable.

Al final el control entregará valores nítidos o reales, consecuencia de las reglas lingüísticas previamente estructuradas, con lo cual este sistema interpretará las órdenes y realizará las acciones pertinentes.

3.3. Estado del arte

Artículo	Descripción	Resultado
Un enfoque de semáforo inteligente utilizando algoritmos de visión computacional en una intersección aislada para optimizar el flujo vehicular.[1] *(08/02/2017)*	El sistema consta de una serie de cámaras que son colocadas en una intersección vehicular, mismas que detectan el movimiento de los autos mediante algoritmos de visión computacional. Posteriormente, la información recabada por las cámaras es enviada a un sistema de control inteligente, el cual se encarga de analizar y determinar cuándo cambiar la luz del semáforo.	Solo se evaluó la detección de cantidad de carros de una imagen y se logró tener el resultado correcto. El siguiente paso es la evaluación de toma de decisiones.
Control de tráfico vehicular automatizado utilizando Lógica Difusa.[2] *(2008)*	Se ha utilizado la librería de Matlab Fuzzy. Se analizó una intersección de dos avenidas que cuentan con tres periodos de semáforos. Todos los diseños del programa de control fueron implementados inicialmente en Matlab utilizando la librería Fuzzy, como referencia. En base a ello se ha realizado la implementación del algoritmo final en un dispositivo de hardware: PIC, para realizar el procesamiento de la información utilizando el lenguaje de programación PIC basic.	Presenta muchos altibajos en la eficiencia llegando en el peor de los casos a una eficiencia de menos del 30 %, lo cual es inaceptable.
Semáforos inteligentes para la regulación del tráfico vehicular.[3] *(07/2014)*	La Investigación desarrolla un Sistema de Semáforo Inteligente (SSI), basado en lógica difusa, que según la densidad vehicular capturada por cámaras web, permiten organizar los cambios de luces en función de las condiciones que se presenten en la zona. Este trabajo es una aplicación de lógica difusa, y está basado en visión por computador, cámaras web que permiten la entrada de datos, lenguaje de programación Python, para el procesamiento de imágenes algoritmos de visión, como es OpenCV y Highgui, así como del Microcontrolador PIC 18F2550.	Mediante el desarrollo de un prototipo se simuló el flujo de vehículos que cruzan por dos intersecciones. El uso del sistema de semáforos inteligentes con lógica difusa ha permitido regular el tráfico vehicular, obteniendo resultados muy favorables, en donde los semáforos permiten dar tiempos variables dependiendo de la densidad vehicular en tiempo real, logrando así mayor fluidez del flujo vehicular.

Cuadro 3.3: Continúa en la siguiente página...

Cuadro 3.3: Continúa de la página anterior

Articulo	Descripción	Resultado
Control de tráfico vehicular por medio de semáforos inteligentes.[4] *07/2013*	Se desarrollo un sistema de semáforos inteligentes para el control del tráfico vehicular basado en hardware programado en lenguajes de alto nivel compilados. se utilizó la metodología *open up*, se establece el sistema a desarrollar tomando en cuenta los problemas y soluciones, se definen las expectativas y se establecen los requerimientos en torno al hardware y el software los cuales son Sensores de ultrasonido, Kit *Arduino*, Sistema Operativo Windows o Linux, Lenguaje Todos los soportados por el Arduino.	Se hicieron una serie de pruebas en un ambiente de intersecciones simulado para determinar si se han alcanzado las expectativas del proyecto. Culminado el proyecto, se emiten recomendaciones basándose en las pruebas, para ayudar a mantener el sistema funcionando óptimamente y para su futura mejoría.
Modelo de Semaforización Inteligente para la Ciudad de Bogotá. [5] *(27/09/2007)*	Analiza una corriente vehicular (varios vehículos) a través de una secuencia de semáforos que pueda presentar distancias variables entre ellos, y luego controlar estos semáforos para pretender mantener la velocidad máxima de la corriente vehicular permitida en la vía. La aplicación presentada en este artículo se fundamenta en una herramienta basada en el modelo ANFIS (Adaptive Neuro-Based Fuzzy Inference System ó Sistema adaptativo de deducciones borrosas basado en neuronas) la cual está disponible en lenguaje MATLAB, con utilidad en el pronóstico de series de tiempo.	El control de los semáforos con el modelo ANFIS es más óptimo ya que la densidad vehicular se reduce, permitiendo atender una mayor cantidad de vehículos en una misma distancia al compararse con el Sistema de temporización fija.

Cuadro 3.3: Continúa en la siguiente página...

Cuadro 3.3: Continúa de la página anterior

Articulo	Descripción	Resultado
Selección de un algoritmo de visión de computadora para la detección de vehículos. [6] *(7/12/2017)*	Este trabajo es una aplicación dentro del campo de la Inteligencia Artificial, específicamente dentro de Lógica difusa, está basado en visión por computador, cámaras web que permiten la entrada de datos, lenguaje de programación Python, para el procesamiento de imágenes algoritmos de visión, como es OpenCV y Highgui, así como del Microcontrolador PIC 18F2550 que permiten en gran medida disminuir la congestión como principal propósito de la investigación.	se hicieron pruebas con diferentes escenarios y con diversas tomas las cuales nos dieron muchos resultados pero llegando a la conclusión que el mejor escenario para una buena detección conteo de los vehículos es cuando el día está totalmente despejado y con una buena cantidad de luz solar y se obtienen resultados de hasta un 95 % de efectividad.

Cuadro 3.3: Estado del arte

Capítulo 4

Desarrollo

En este capítulo se describe el desarrollo del sistema en general: la selección de la técnica, el desarrollo de la técnica seleccionada y la selección del lenguaje de programación.

A continuación, en la **sección 4.2** se describe, a manera de resumen: las diferentes técnicas tomadas en cuenta durante la investigación y los porqués de la técnica seleccionada.

Después en la **sección 4.3**, se describe el proceso de diseño del *Sistema de Inferencia Difuso* bajo una metodología de mejora iterativa. El sistema tiene como objetivo calcular el tiempo de la fase en verde de un semáforo, que mediante la estructura propuesta y un preprocesamiento de los datos, permitirá gestionar diferentes tipos de intersecciones.

Posteriormente en la **sección 4.4** se modela, mediante diagramas UML, la arquitectura del software desarrollado. El resultado es una micro-librería que permite *expresar las variables y reglas del sistema difuso en notación de objetos*. Además, se modela la estructura propuesta para integrar la técnica seleccionada, a un sistema de control (a un algoritmo de sincronización) de semáforos y así, posibilitar la aplicación del sistema a una intersección con número no-estático de avenidas y fases.

4.1. Preliminares

Antes de continuar con el diseño del sistema, se considera adecuado aclarar algunos puntos acerca de:

El enfoque presentado en este proyecto difiere con el resto en que, fue diseñado para ser embebido dentro de un algoritmo de sincronización de semáforos más general, de esta manera la tarea del *FIS* se centra en determinar el tiempo en verde, dejando los detalles de sincronización en manos de un segundo algoritmo (también propuesto y desarrollado en este proyecto).

El pre-procesamiento de los datos permite, mediante un proceso matemático sencillo, adaptar el *FIS* propuesto a cruces con un número arbitrario de avenidas y fases, por ejemplo, intersecciones de: 2 avenidas y 2 fases, 4 avenidas y 2 fases o 4 avenidas y 4 fases. De esta manera se logra integrar una técnica de inteligencia artificial con un segundo algoritmo para la sincronización de semáforos.

La elección del lenguaje de programación para la implementación del algoritmo fue bastante sencilla. Se requería un lenguaje eficiente que permitiera tener un control fino del proceso de inferencia, además de que permitiera que el algoritmo fuese portable entre plataformas. Por ello, se optó por el lenguaje de programación *C++*, en específico, el estándar *ISO C++11*.

C++ es uno de los más eficientes, con herramientas como punteros y referencias que evitan el desperdicio de memoria además de la *semántica de movimiento* incluida en el estándar C++11 que, aligera el paso de objetos de gran tamaño entre funciones (como los vectores usados en el proceso de inferencia). Por lo dicho anteriormente y por su amplia disponibilidad de compiladores en casi todas las plataformas, C++ resulta la elección perfecta para este proyecto.

C++ es un lenguaje **multiparadigma**, permite trabajar usando programación estructurada, orientada a objetos e incluso funcional. En este proyecto se decidió hacer uso del paradigma orientado a objetos, debido a que se busca crear piezas de software altamente reutilizables y expresivas.

4.2. Selección de la técnica

Después de investigar acerca de las principales ramas de la Inteligencia Artificial (véase sección 3.2.2), se decidió optar por el uso de una técnica de Lógica Difusa; a saber, un *Sistema Inferencia Difusa (FIS por sus siglas en inglés)*. Cabe mencionar que esta técnica está siendo ampliamente usada en la optimización del tráfico vehicular (véase la tabla 3.3), esto se debe en parte, a que permite razonar sobre términos imprecisos como "*muchos carros*" o "*poca congestión*".

La elección de esta técnica sobre las otras se debe a varios factores:

a) Algoritmos Genéticos. Son una buena opción ya que podrían permitir probar diferentes asignaciones de tiempo, evaluar su desempeño y, así, ir depurando hasta encontrar la configuración optima de tiempos. Sin embargo, debido que la evaluación del desempeño podría ocasionar una congestión bastante alta, no es factible evaluarlo en un entorno real. Otra opción es evaluar el algoritmo en un entorno simulado, por desgracia, el tiempo requerido para la desarrollo del AG, del entorno de simulación y su correspondiente modelo matemático, va más allá del disponible para este proyecto.

b) Redes Neuronales Artificiales. Estas son de gran ayuda en tareas como reconocimiento de voz, clasificación automática, tratamiento de imágenes entre otros. Sin embargo, desarrollar de una RNA para determinar el cambio de fase de un semáforo, podría llevar a lidiar con dificultades innecesarias, dificultades inherentes al entrenamiento de la propia red.

c) Sistemas Expertos. A primera vista puede parecer una buena idea, sin embargo, los problemas vienen al momento de expresar el conocimiento del experto en términos cuantitativos ya que no es muy fácil decidir en qué punto se le debe ceder el paso a una avenida u otra.

Una variante, son los sistemas difusos. Aquí sucede algo importante: los controladores difusos y los sistemas expertos difusos son en esencia lo mismo: Sistema de Inferencia Difusa. Es decir, ambos se basan en la inferencia difusa, por ejemplo: la inferencia de mamdani.

d) Sistema de Inferencia Difusa. La gran ventaja de estos sistemas es que, gracias a que

usan lógica difusa, permiten expresar el conocimiento (reglas) en términos coloquiales que poseen cierto grado de incertidumbre. Así, es posible modelar un sistema que de preferencia a las avenidas con muchos carros y, definir el término muchos carros con una función de membresía centrada en 7. De esta manera, entre más cercano sea a 7 el valor evaluado, mayor peso tendrá en la decisión tomada por el sistema.

Estos sistemas permiten lidiar con la información que no es tan precisa o, en el caso de este proyecto, permite razonar sobre términos como *"muchos carros"* o *"poca congestión"*. Términos para los cuales, no es fácil asignar un intervalo de valores. Es por eso que se decidió usar un Sistema de Inferencia Difusa para dotar de cierto grado de "inteligencia" a un algoritmo de control de semáforos.

4.3. Diseño del sistema de inferencia

En esta sección se expone de manera metodológica las diferentes configuraciones, resultados y conclusiones que se obtuvieron en el desarrollo del sistema. Para la configuración se sigue una estrategia de mejora iterativa hasta alcanzar el resultado deseado.

4.3.1. Configuraciones aplicadas de manera general

Selección del método de inferencia

Se optó por el método de **Mamdani** debido a que emula mejor el razonamiento humano.

Selección de operadores

Debido a la variedad de operadores que pueden ser usados en la inferencia difusa, a continuación se muestran los operadores usados en este proyecto.

- **Método AND:** operador mínimo *min*.
- **Método OR:** operador máximo *max*.
- **Implicación:** operador mínimo *min*.
- **Agregación:** operador máximo *max*.
- **Defuzzificación:** centroide $\frac{\sum \mu(x).x}{\sum \mu(x)}$.

Selección de variables

Como se mencionó, el diseño del sistema difuso, se enfoca en el cálculo de tiempos óptimos para las fases verdes de los semáforos, para ello, se consideran tres aspectos, a saber: la cantidad de autos a los que se les cederá el paso, la cantidad de autos que quedarán en espera y, el número de carriles por avenida. Estos datos permitirán al sistema de inferencia dar preferencia a aquellas avenidas con un mayor número de carriles y/o mayor número de autos.

La selección de variables de entrada y salida del sistema, debe ser cuidadosa para no elevar la complejidad del sistema ni caer en redundancias. En el desarrollo de este sistema se ha seguido la filosofía KISS[1] que plantea el desarrollo de sistemas sencillos, flexibles y escalables.

Al seleccionar las variables hay que cuidar que estás estén correlacionadas con las variables de salida, además, se debe mantener el número de variables y particiones al mínimo necesario.

Habiendo dicho lo anterior, se presenta la selección de variables, así, como una breve justificación de su elección.

Variables de entrada

a) Vehículos. La variable vehículos indica la cantidad de autos que se encuentran en la avenida a la que se le dará el paso.

b) Congestión. La variable congestión se propone como un indicador que sume de manera ponderada la cantidad de automóviles y carriles. A saber, la suma ponderada se define como:

$$\frac{\sum_{i=1}^{n} x_i * w_i}{\sum_{i=1}^{n} w_i}$$

Donde:

x es un vector con la cantidad de autos de las avenidas.

w es un vector de pesos que describe la prioridad de cada avenida.

[1] Keep it simple, Stupid!

Además, se propone el cálculo de los pesos de las avenidas como una función directamente proporcional al número de carriles de la avenida en cuestión e inversamente proporcional al número total de carriles , i,e:

$$w_i = \frac{c_i}{\sum_{j=1}^{j=n} c_j}$$

Donde:

C es un vector con la cantidad de carriles de cada avenida.

i es la avenida a evaluar.

De esta manera el sistema resulta lo suficientemente flexible para ser aplicado a intersecciones de 2, 4, 8 o cualquier numero n de avenidas (siempre y cuando tenga sentido).

Variables de salida

a) Tiempo. Con el fin de hacer el sistema fácilmente escalable, se propone como variable de salida la cantidad de segundos de la fase verde del semáforo en cuestión. Esto permitirá integrar el sistema de inferencia dentro de un algoritmo clásico de control de semáforos. Así, la *inteligencia* se encuentra en la optima asignación de tiempos.

4.3.2. Configuración A

La configuración presentada a continuación mantiene al mínimo el número de términos dentro de las variables de entrada y salida.

Variable de entrada Vehículos. Referente a la cantidad de vehículos en la avenida a la cual se le asignará la fase verde. La variable se encuentra definida en un universo de discurso $U = [0, 12]$.

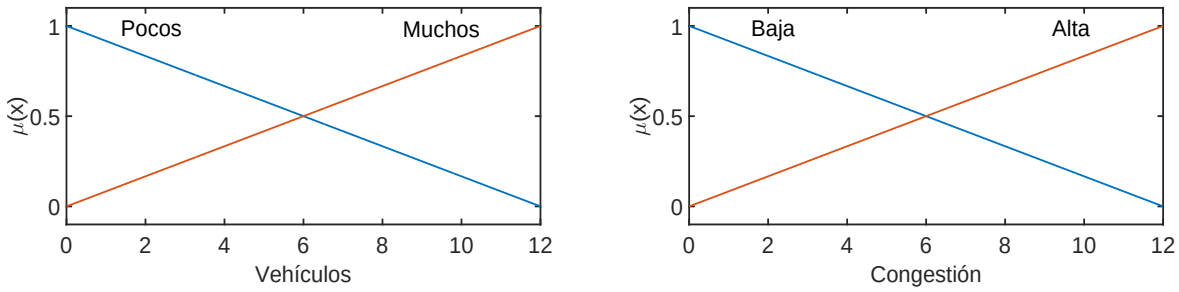
Termino lingüístico	Función de membresía	Parámetros
Pocos	Triangular	[0, 0, 12]
Muchos	Triangular	[0, 12, 12]

Cuadro 4.1: Variable lingüística *Vehículos*

Variable de entrada Congestión. Se refiere al índice ponderado de la cantidad de vehículos en las avenidas que quedaran a la espera durante la fase verde actual. La variable se encuentra definida en un universo de discurso $U = [0, 12]$.

Termino lingüístico	Función de membresía	Parámetros
Baja	Triangular	[0, 0, 12]
Alta	Triangular	[0, 12, 12]

Cuadro 4.2: Variable lingüística *Congestión*



(a) Variable lingüística Vehículos

(b) Variable lingüística Congestión

Figura 4.1: Representación gráfica de las variables lingüísticas Vehículos y Congestión

Variable de salida Tiempo. Se refiere al tiempo en segundos que le será asignada a la fase verde actual. La variable se encuentra definida en un universo de discurso $U = [0, 90]$.

Termino lingüístico	Función de membresía	Parámetros
Bajo	Triangular	[0, 0, 30]
Medio	Triangular	[0, 30, 60]
Moderado	Triangular	[30, 60, 90]
Alto	Triangular	[60, 90, 90]

Cuadro 4.3: Variable lingüística *Tiempo*

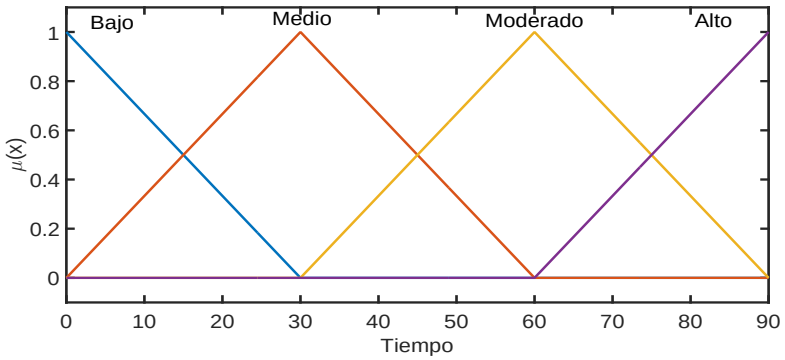


Figura 4.2: Representación gráfica de la variable lingüística Tiempo

Base de conocimientos

La siguiente tabla muestra las reglas difusas empleadas en esta configuración.

Antecedente				Consecuente
Vehículos Pocos	Y	Congestión Baja	→	Verde Medio
Vehículos Pocos	Y	Congestión Alta	→	Verde Bajo
Vehículos Muchos	Y	Congestión Baja	→	Verde Alto
Vehículos Muchos	Y	Congestión Alta	→	Verde Moderado

Cuadro 4.4: Reglas difusas para la configuración *A*

Resultados

Una vez diseñado e implementado el sistema de inferencia, se le suministraron valores de prueba para evaluar su desempeño. Los resultados de dicha evaluación se reflejan en la tabla siguiente:

$V \backslash C$	0	3	6	9	12
0	30.00	28.99	26.21	20.86	09.70
3	39.02	39.03	37.69	35.58	33.96
6	46.40	45.83	45.00	44.16	43.59
9	56.03	54.41	52.30	50.96	50.97
12	80.29	69.13	63.79	61.00	59.99

Cuadro 4.5: Resultados de la evaluación para la configuración

Donde: la columna V son los valores de prueba de la variable Vehículos, la fila C son los valores de prueba de la variable Congestión y las celdas son los tiempos (en segundos).

Observaciones

Después de evaluar el desempeño de la primera configuración se observa como efectivamente el tiempo asignado varía de acuerdo a las variables de entrada; sin embargo, el tiempo asignado para las entradas Vehículos = 0 y Congestión = 6, resulta excesivo puesto que la cantidad de vehículos a los que se les cederá el paso es mínimo (0) mientras que el índice de vehículos (6) que se mantendrán a la espera es bastante considerable.

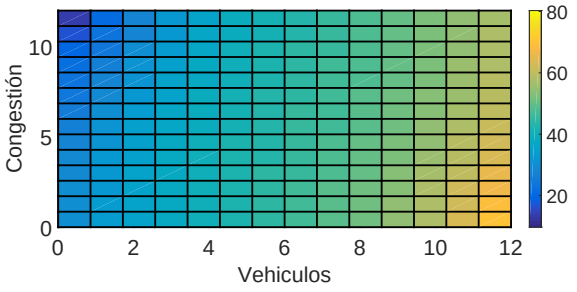


Figura 4.3: Superficie de control

4.3.3. Configuración B

La configuración presentada en esta sección, ajusta los parámetros de las funciones de membresía de la variable de salida Tiempo. La configuración pasa a ser la siguiente:

Variable de entrada Vehículos. Se mantiene sin cambios.

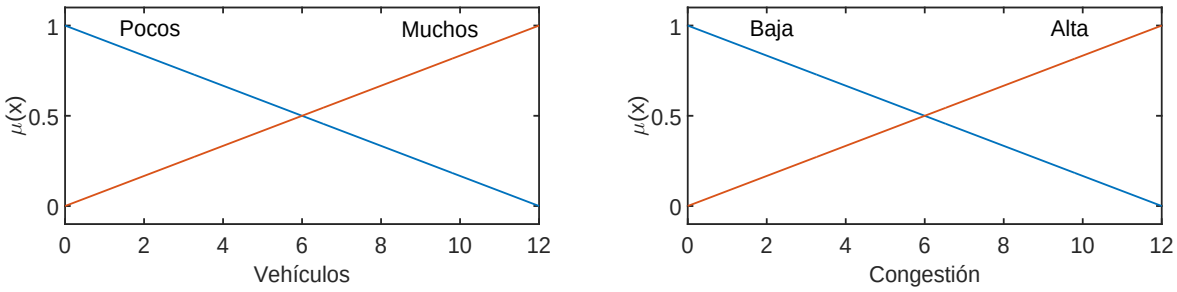
Termino lingüístico	Función de membresía	Parámetros
Pocos	Triangular	[0, 0, 12]
Muchos	Triangular	[0, 12, 12]

Cuadro 4.6: Variable lingüística *Vehículos*

Variable de entrada Congestión. Se mantiene sin cambios desde la configuración anterior.

Termino lingüístico	Función de membresía	Parámetros
Baja	Triangular	[0, 0, 12]
Alta	Triangular	[0, 12, 12]

Cuadro 4.7: Variable lingüística *Congestión*



(a) Variable lingüística Vehículos

(b) Variable lingüística Congestión

Figura 4.4: Representación gráfica de las variables lingüísticas Vehículos y Congestión

Variable de salida Tiempo. En esta ocasión los parámetros de las funciones de membresía fueron ajustados para generar tiempos más acertados. La nueva configuración de esta variable se muestra a continuación:

Termino lingüístico	Función de membresía	Parámetros
Bajo	Triangular	[0, 0, 20]
Medio	Triangular	[0, 20, 45]
Moderado	Triangular	[20, 45, 70]
Alto	Triangular	[45, 70, 70]

Cuadro 4.8: Variable lingüística *Tiempo*

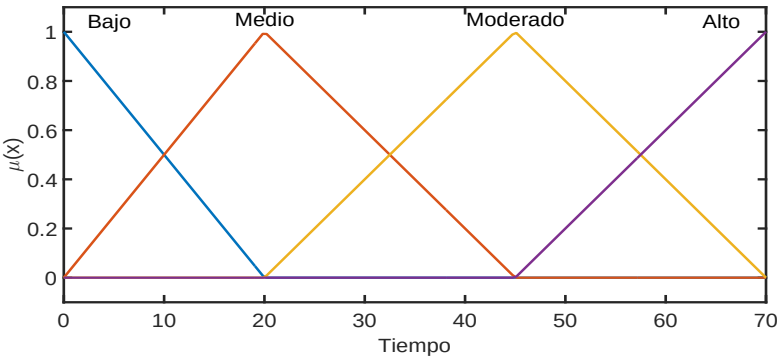


Figura 4.5: Representación gráfica de la variable lingüística Tiempo

Base de conocimientos

La siguiente tabla muestra las reglas difusas empleadas en esta configuración que se mantiene sin cambios.

Antecedente				Consecuente
Vehículos Pocos	Y	Congestión Baja	→	Verde Medio
Vehículos Pocos	Y	Congestión Alta	→	Verde Bajo
Vehículos Muchos	Y	Congestión Baja	→	Verde Alto
Vehículos Muchos	Y	Congestión Alta	→	Verde Moderado

Cuadro 4.9: Reglas difusas para la configuración *B*

Resultados

Una vez reajustados los parámetros del sistema de inferencia, se le suministraron lo mismos valores de prueba para evaluar su desempeño respecto a la configuración anterior. Los resultados de dicha evaluación se reflejan en la tabla siguiente:

$V \backslash C$	0	3	6	9	12
0	21.67	21.09	19.37	15.68	06.44
3	29.56	29.69	29.16	28.26	27.11
6	35.87	35.51	35.00	34.49	34.13
9	43.75	42.58	40.44	39.14	39.18
12	61.90	52.60	48.15	45.83	45.00

Cuadro 4.10: Resultados de la evaluación para la configuración B

Donde: la columna V son los valores de prueba de la variable Vehículos, la fila C son los valores de prueba de la variable Congestión y las celdas son los tiempos (en segundos) obtenidos por la configuración actual.

Observaciones

Después de evaluar el desempeño de la segunda configuración se observa como efectivamente el tiempo asignado es más acorde a la cantidad de vehículos; sin embargo, el tiempo asignado para las entradas Vehículos = 0 y Congestión = 0, resulta generoso puesto que la cantidad de vehículos a los que se les cederá el paso es mínimo (0) mientras al igual que el índice de vehículos (6); Sería mejor que en estos casos el tiempo fuera más corto para acelerar el ciclo del semáforo y reducir la espera de las otras avenidas. En otro casos como V = 12 y C = 12, el tiempo podría resultar insuficiente para desahogar el tráfico, por lo que se ajustará en la siguiente configuración.

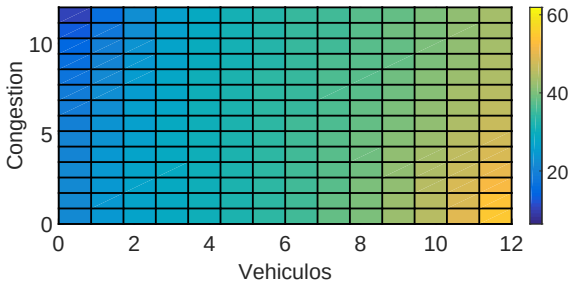


Figura 4.6: Superficie de control

4.3.4. Configuración C

En está ocasión los reajustes serán mayores a los de la configuración anterior. Puesto que se agregaran un nuevo termino lingüístico a las variables Vehículos y Tiempo.

Variable de entrada Vehículos. Se agrega un nuevo término lingüístico para aumentar la flexibilidad del sistema.

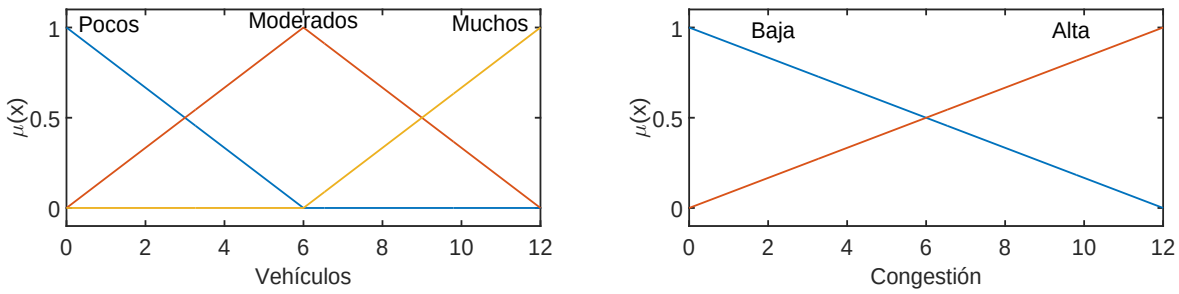
Termino lingüístico	Función de membresía	Parámetros
Pocos	Triangular	[0, 0, 6]
Moderados	Triangular	[0, 6, 12]
Muchos	Triangular	[6, 12, 12]

Cuadro 4.11: Variable lingüística *Vehículos*

Variable de entrada Congestión. Se mantiene sin cambios desde la configuración anterior.

Termino lingüístico	Función de membresía	Parámetros
Baja	Triangular	[0, 0, 12]
Alta	Triangular	[0, 12, 12]

Cuadro 4.12: Variable lingüística *Congestión*



(a) Variable lingüística Vehículos

(b) Variable lingüística Congestión

Figura 4.7: Representación gráfica de las variables lingüísticas Vehículos y Congestión

Variable de salida Tiempo. Al igual que la variable vehículos, se ha agregado un nuevo término lingüístico para aumentar la flexibilidad. Además los parámetros de las funciones fueron reajustados ligeramente al igual que sus etiquetas.

Termino lingüístico	Función de membresía	Parámetros
Mínimo	Triangular	[0, 10, 20]
Bajo	Triangular	[0, 20, 40]
Medio	Triangular	[20, 40, 60]
Alto	Triangular	[40, 60, 80]
Extra	Triangular	[60, 80, 80]

Cuadro 4.13: Variable lingüística *Tiempo*

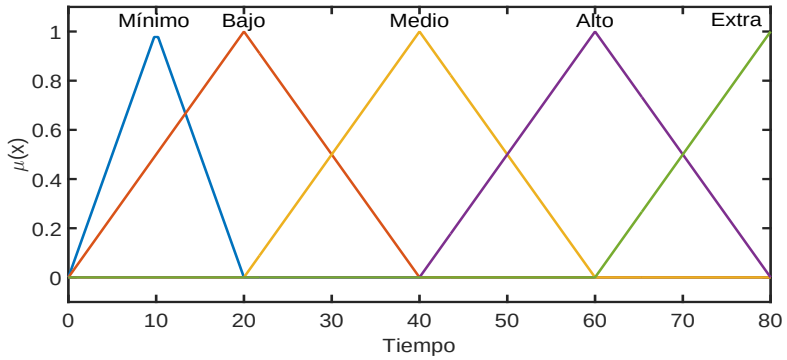


Figura 4.8: Representación gráfica de la variable lingüística Tiempo

Base de conocimientos

La siguiente tabla muestra las reglas difusas empleadas en esta configuración que se mantiene sin cambios.

Antecedente				Consecuente
Vehículos Pocos			→	Verde Mínimo
Vehículos Moderados	Y	Congestión Baja	→	Verde Medio
Vehículos Moderados	Y	Congestión Alta	→	Verde Bajo
Vehículos Muchos	Y	Congestión Baja	→	Verde Extra
Vehículos Muchos	Y	Congestión Alta	→	Verde Alto

Cuadro 4.14: Reglas difusas para la configuración *C*

Resultados

Una vez reajustados los parámetros del sistema de inferencia, se le suministraron lo mismos valores de prueba para evaluar su desempeño respecto a la configuración anterior. Los resultados de dicha evaluación se reflejan en la tabla siguiente:

$V \backslash C$	0	3	6	9	12
0	10.00	10.00	10.00	10.00	10.00
3	30.00	29.63	28.81	25.59	18.85
6	40.00	34.21	30.00	25.79	20.00
9	59.74	44.49	42.44	40.75	40.00
12	73.33	65.87	62.38	60.59	60.00

Cuadro 4.15: Resultados de la evaluación para la configuración C

Donde: la columna V son los valores de prueba de la variable Vehículos, la fila C son los valores de prueba de la variable Congestión y las celdas son los tiempos (en segundos) obtenidos por la configuración actual.

Observaciones

En esta tercera configuración se aprecia que hubo mejoras considerables en los tiempos inferidos por el sistema. Si se observa la salida para V = 12 y C = 12, cuya salida es 60, se nota que es un tiempo bastante acertado que ayudaría a desahogar la congestión. Por otro lado cuando V = 12 y C = 0, el sistema otorga 13 segundos extra. En el caso contrario, cuando V = 0 y C = 12, el sistema asigna el tiempo mínimo de 10 segundos. Hace falta un nuevo ajuste para saber si los tiempos pueden mejorar aún más.

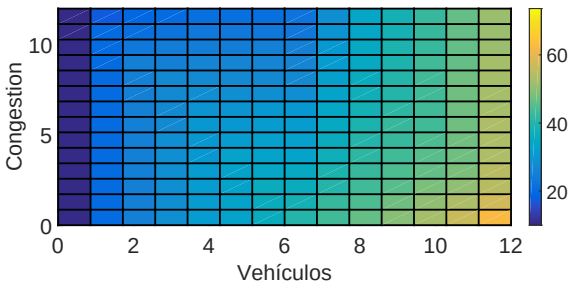


Figura 4.9: Superficie de control

4.3.5. Configuración D

Existen funciones de membresía que modelan mejor el modo de clasificación que realiza la mente humana, estas funciones son las curvas tales como: *Sigmoidales, Gaussianas y Campanas Generalizadas*; en esta configuración se explota el potencial de las primeras dos.

Variable de entrada Vehículos. Las funciones de membresía *triangulares* se remplazan por funciones *Sigmoidales y Gaussianas*, estás modelan de manera más eficiente la manera en que el ser humano suele clasificar los fenómenos.

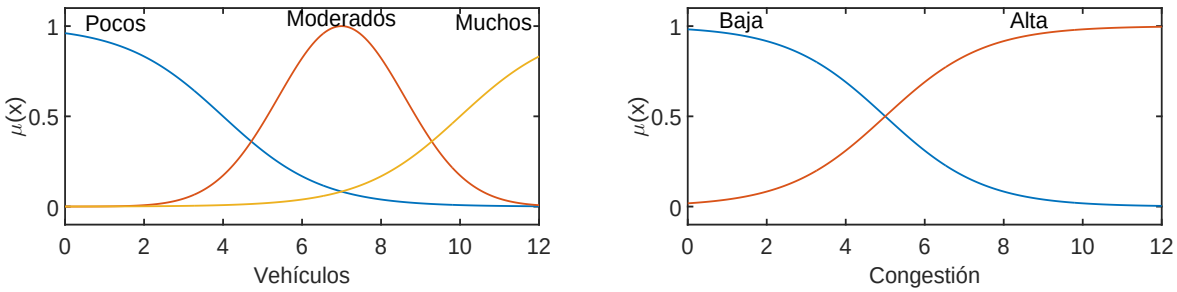
Termino lingüístico	Función de membresía	Parámetros
Pocos	Sigmoidal	[-0.8, 4]
Moderados	Gaussiana	[1.6, 7]
Muchos	Sigmoidal	[0.8, 10]

Cuadro 4.16: Variable lingüística *Vehículos*

Variable de entrada Congestión. También se remplaza las funciones triangulares por *funciones Sigmoidales.*

Termino lingüístico	Función de membresía	Parámetros
Baja	Sigmoidal	[-0.8, 5]
Alta	Sigmoidal	[0.8, 5]

Cuadro 4.17: Variable lingüística *Congestión*



(a) Variable lingüística Vehículos

(b) Variable lingüística Congestión

Figura 4.10: Representación gráfica de las variables lingüísticas Vehículos y Congestión

Variable de salida Tiempo. Esta variable únicamente tiene un ajuste menor en el término lingüístico *Mínimo*, el resto de términos permanece intacto.

Termino lingüístico	Función de membresía	Parámetros
Mínimo	Triangular	[0, 0, 25]
Bajo	Triangular	[0, 25, 40]
Medio	Triangular	[20, 40, 60]
Alto	Triangular	[40, 60, 80]
Extra	Triangular	[60, 80, 80]

Cuadro 4.18: Variable lingüística *Tiempo*

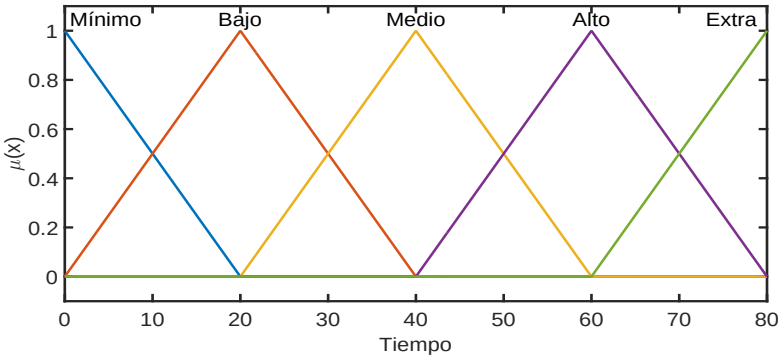


Figura 4.11: Representación gráfica de la variable lingüística Tiempo

Base de conocimientos

La siguiente tabla muestra las reglas difusas empleadas en esta configuración que se mantiene sin cambios.

Antecedente				Consecuente
Vehículos Pocos			→	Verde Mínimo
Vehículos Moderados	Y	Congestión Baja	→	Verde Medio
Vehículos Moderados	Y	Congestión Alta	→	Verde Bajo
Vehículos Muchos	Y	Congestión Baja	→	Verde Extra
Vehículos Muchos	Y	Congestión Alta	→	Verde Alto

Cuadro 4.19: Reglas difusas para la configuración *D*

Resultados

Después de remplazar algunas de las funciones de membresía y realizar los ajustes necesarios, se le suministró los mismos valores de prueba para evaluar su desempeño respecto a la configuración anterior. Los resultados de dicha evaluación se reflejan en la tabla siguiente:

$V \backslash C$	0	3	6	9	12
0	08.41	08.41	08.41	08.41	08.41
3	13.25	13.25	13.25	12.98	10.85
6	36.56	36.54	28.81	24.08	24.02
9	47.81	43.60	39.23	37.65	37.62
12	70.83	66.58	60.27	59.40	59.39

Cuadro 4.20: Resultados de la evaluación para la configuración D

Donde: la columna V son los valores de prueba de la variable Vehículos, la fila C son los valores de prueba de la variable Congestión y las celdas son los tiempos (en segundos) obtenidos por la configuración actual.

Observaciones

Definitivamente, los resultados obtenidos tras remplazar las funciones triangulares por funciones sigmoidales y gaussianas, son mucho más acertadas. Al principio, los tiempos asignados incrementan más rápido conforme más se eleva el número de autos, después, el ritmo de crecimiento de los tiempos de asignación desacelera al acercarse al valor máximo para las variables de entrada. Se concluye que la configuración actual será la usada en la implementación.

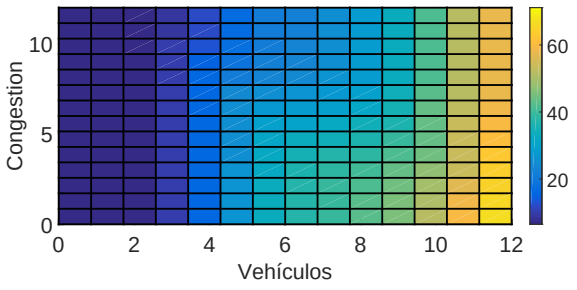


Figura 4.12: Superficie de control

4.4. Desarrollo del algoritmo

Después de haber culminado el desarrollo del sistema de inferencia, el cual representa el componente principal del proyecto, ahora el desarrollo continúa con el algoritmo de sincronización de semáforos.

El modelo propuesto, es a su vez un marco de trabajo para la implementación última, por parte del usuario final.

A lo largo de las siguientes páginas se modela el desarrollo del algoritmo mediante diagramas UML, empleando solo aquellos diagramas que ayuden a tener una perspectiva general del proyecto:

- **Diagrama de actividades** del flujo general del sistema.
- **Diagrama de clases** de todo el sistema.
- **Diagrama de secuencia** del bucle principal.

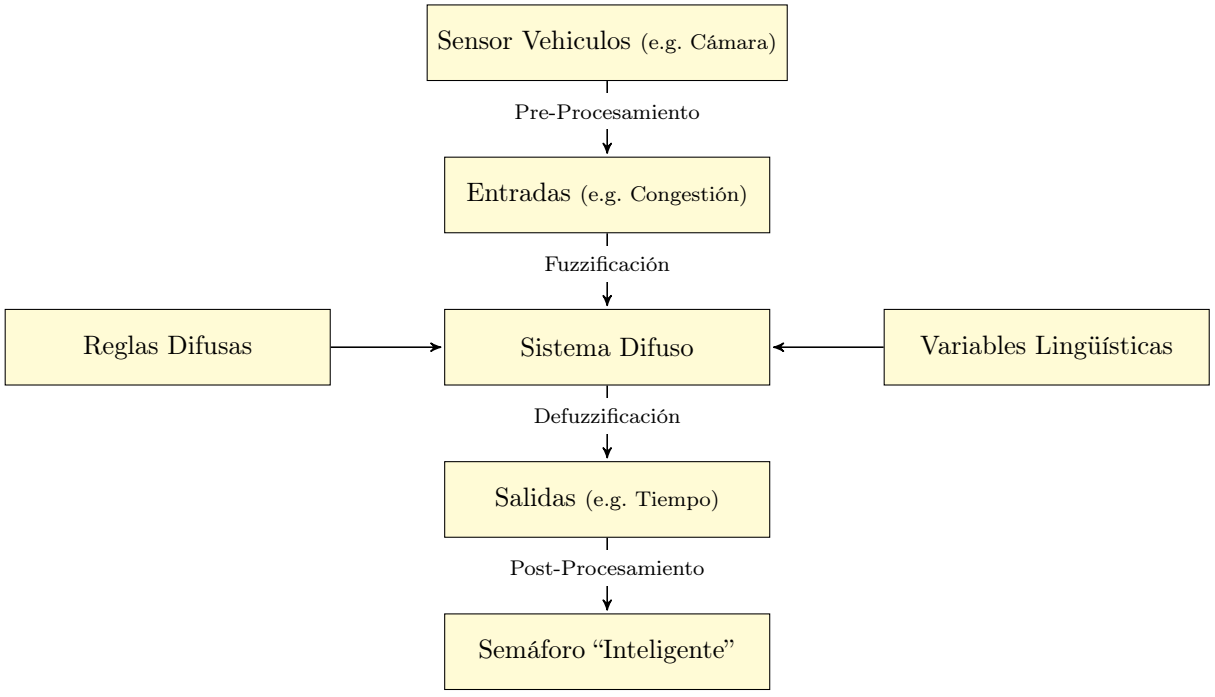


Figura 4.13: Diagrama general del sistema

4.4.1. Diagrama de actividades

El siguiente diagrama muestra a grandes rasgos, el flujo de ejecución general del algoritmo. En él se puede observar el proceso de inicialización y configuración del semáforo y del sensor; además se muestra el bucle principal "*repetir siempre*" que se encarga de obtener los datos del sensor, procesarlos y establecer el cambio de fase.

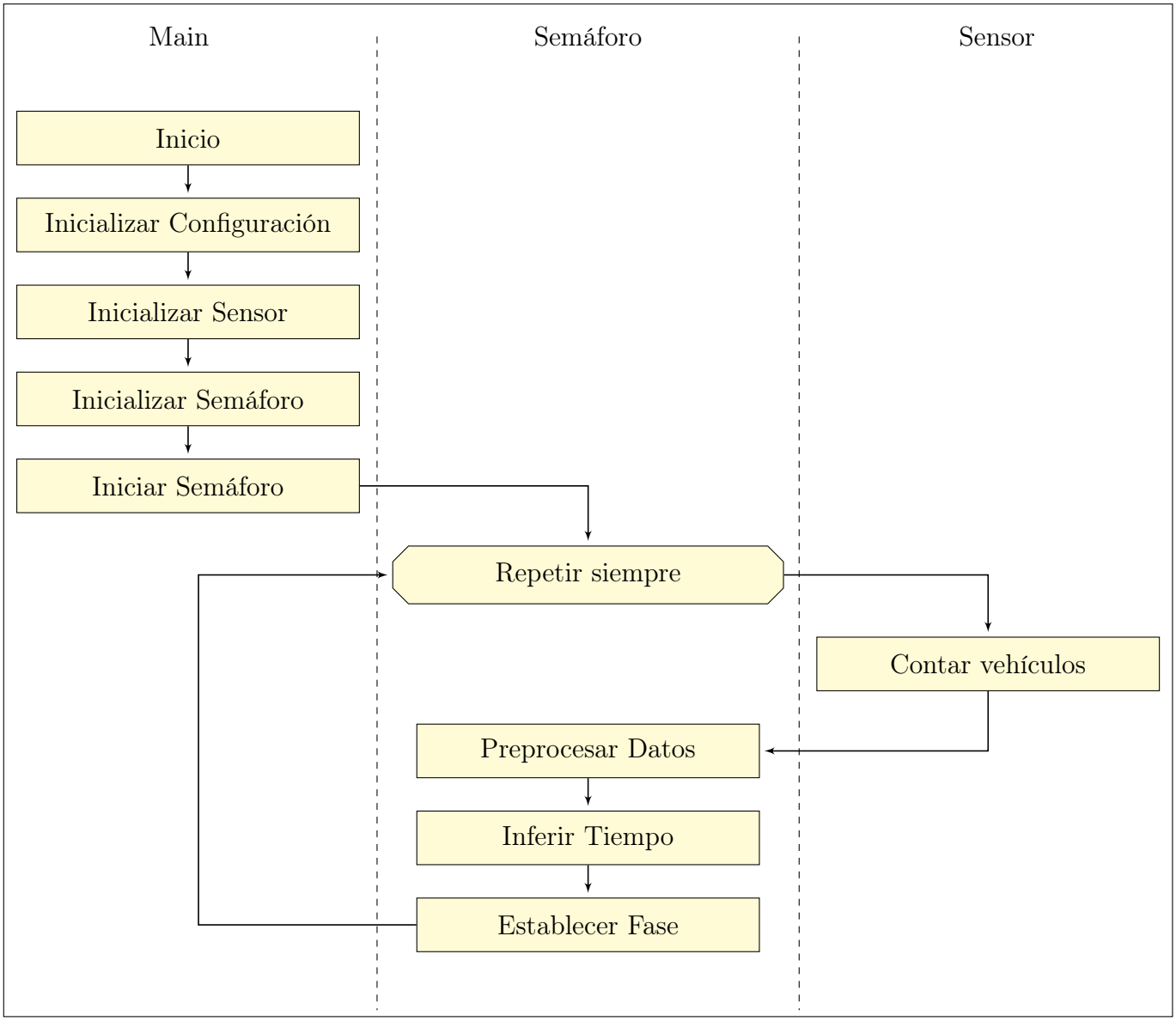


Figura 4.14: Diagrama de actividades general

4.4.2. Diagrama de secuencia

Diagrama de secuencia del bucle de control del semáforo

En el diagrama anterior se muestra el bucle principal del sistema, ahora mediante un diagrama de secuencia se detallarán dicho bucle debido a que es la parte central del sistema.

Para fines de legibilidad, en el siguiente diagrama se omiten los nombres de las clases (*FuzzySemaforo y SensorVehiculos*). Las funciones que se muestran pertenecen a la clase *FuzzySemaforo* excepto una, la cual está indicada en el diagrama.

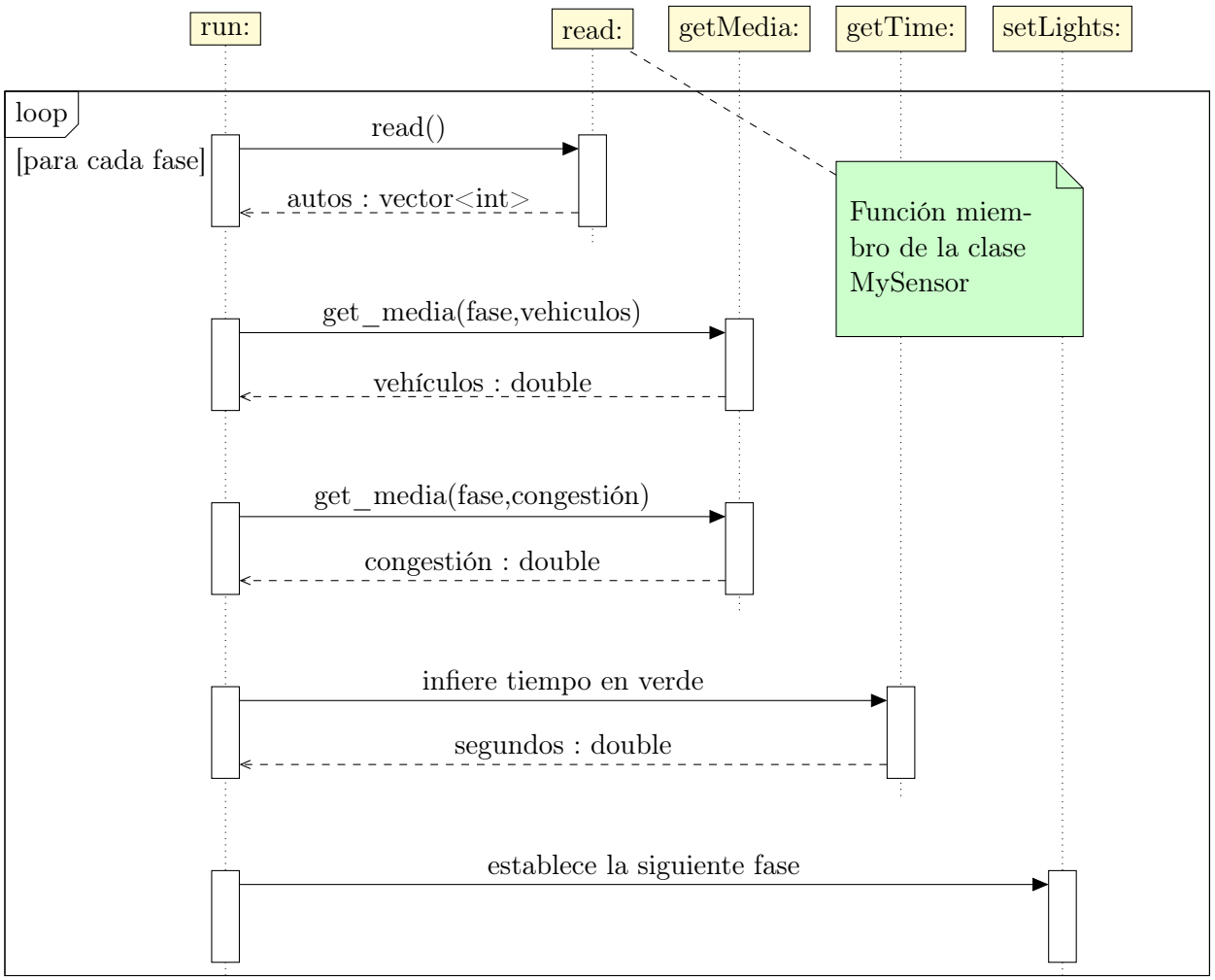


Figura 4.15: Diagrama de secuencia del bucle principal

4.4.3. Diagrama de clases

Modelado de las *relaciones de clase* del sistema

En el siguiente diagrama se modela las clases que constituyen el sistema además, se muestran las diferentes relaciones que existen entre dichas clases.

Para mayor legibilidad, el diagrama no modela atributos ni comportamientos de las clases; sin embargo, en el apéndice A se modelan los elementos omitidos.

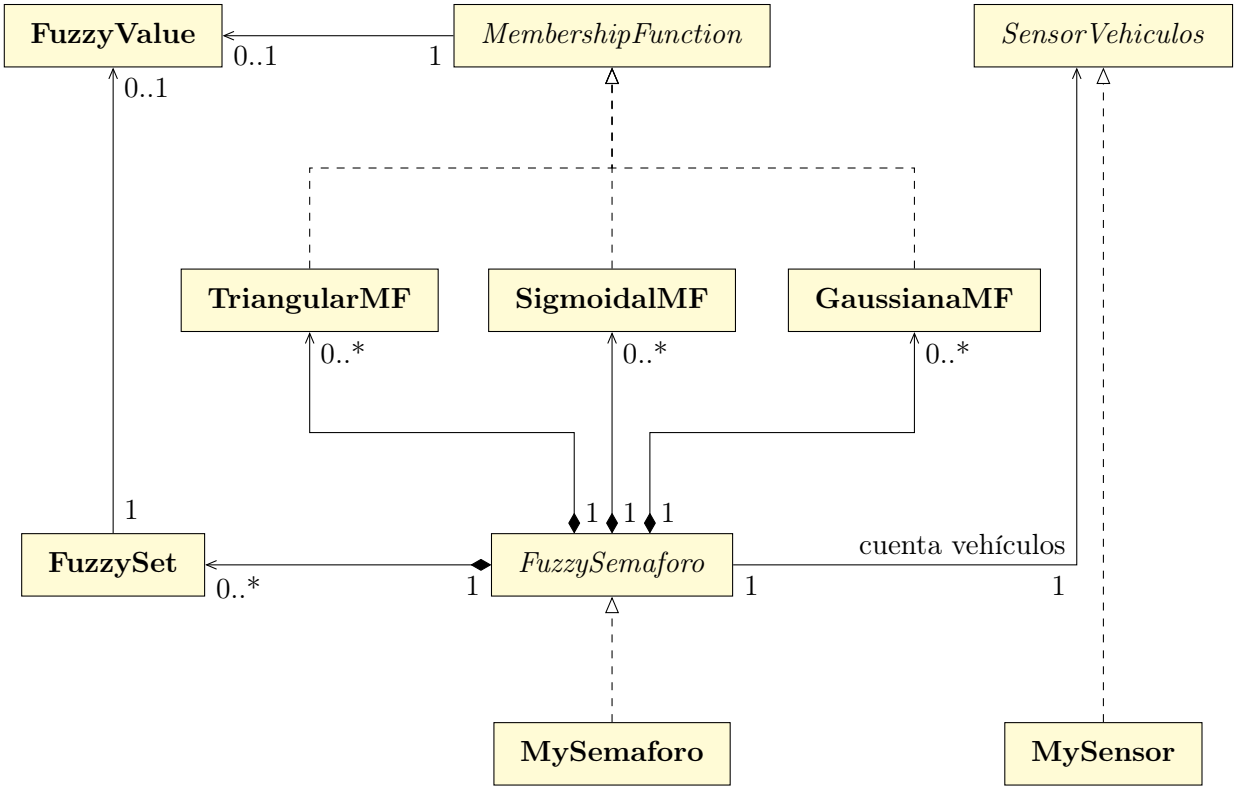


Figura 4.16: Diagrama de clases del sistema

Capítulo 5

Análisis de resultados

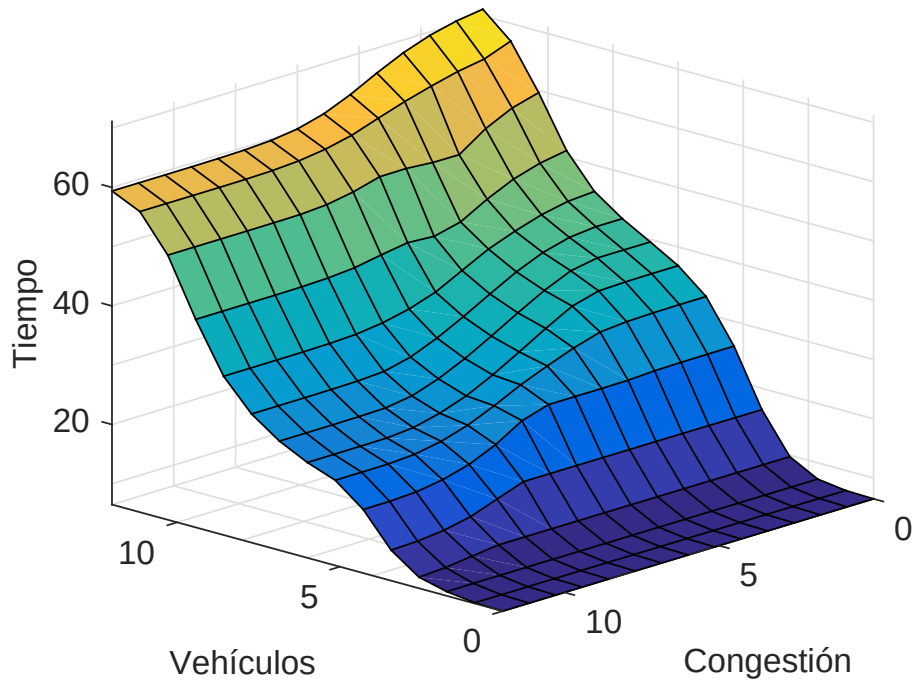
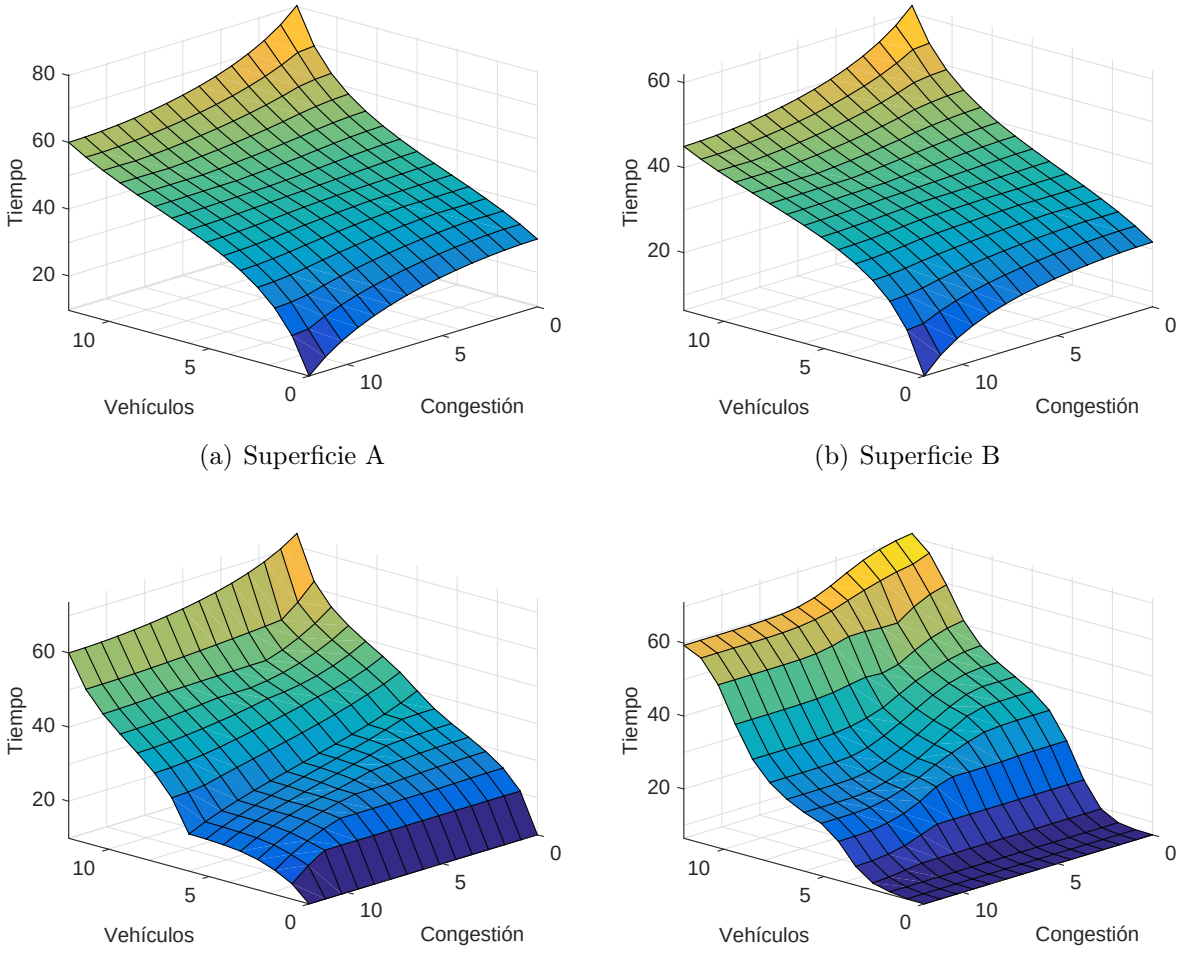


Figura 5.1: Superficie de control del sistema de inferencia

5.1. Comparativa de las superficies de control

Las siguientes figuras son las curvas de control generadas por cada una de las configuraciones probadas en la sección 4.3, cada una de las curvas representa, mediante una gráfica de 3 dimensiones (una superficie), los valores de salida obtenidos por las distintas configuraciones.



(a) Superficie A

(b) Superficie B

(c) Superficie C

(d) Superficie D

Figura 5.2: Superficies de control

Las superficies mostradas arriba permiten ver el cambio gradual de las asignaciones de tiempo, de igual manera permiten apreciar la suavidad de la transición de los valores, además de que facilitan tener una perspectiva general de los valores de salida del sistema de inferencia.

Al analizar las gráficas se aprecia como las superficies A y B se ven afectadas de manera casi lineal por las variables de entrada. La diferencia más evidente es su rango de valores, pues la configuración A arroja valores de hasta 80s mientras que la B apenas rebasa los 60s.

En la superficie C se observa que existe un área donde la transición se vuelve un poco brusca e incluso toma una pendiente negativa. Este tipo de situaciones es más fácil observarlas cuando

se recurre a una gráfica.

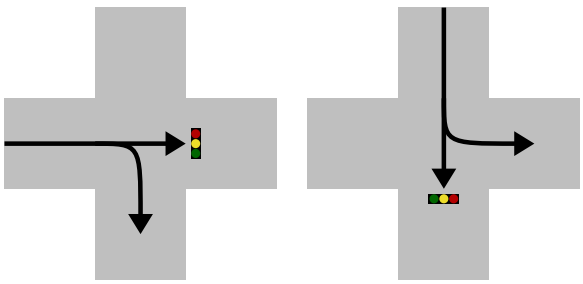
La superficie D, por otro lado, muestra una transición más limpia. También se observa como los valores de salida para congestiones altas se mantiene entre los 60s y 70s, mientras que para congestiones bajas se mantiene en un mínimo de alrededor de 10s. Para valores de congestión entre 5 y 10, se observa como con los mismo valores para la variable vehículos, aumenta hasta alcanzar un máximo que se mantiene.

5.2. Análisis de los repartos de tiempo

Después de haber desarrollado y configurado la técnica seleccionada, ahora, en este capítulo se analizará el desempeño del algoritmo. Como se mencionó anteriormente, el algoritmo puede ser configurado para controlar intersecciones de diferente número de avenidas, carriles y avenidas. Con el fin de evaluar el desempeño del algoritmo frente a diferentes escenarios, se realizaron pruebas para los siguientes tipos de intersecciones que resultan ser las más comunes.

- A: Intersecciones de 2 avenidas con 2 fases,
- B: Intersecciones de 4 avenidas con 2 fases y,
- C: Intersecciones de 4 avenidas con 4 fases.

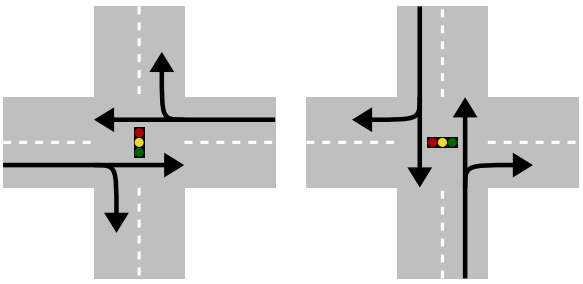
Intersección A: 2 avenidas con 2 fases Se entiende un cruce de dos calles de un solo sentido, donde el semáforo se encarga de ceder el paso de una sola de ellas en cada una de las fases (2 fases) tal y como se ve en la figura siguiente.



(a) Fase 1 (b) Fase 2

Figura 5.3: Intersección de 2 avenidas y 2 fases

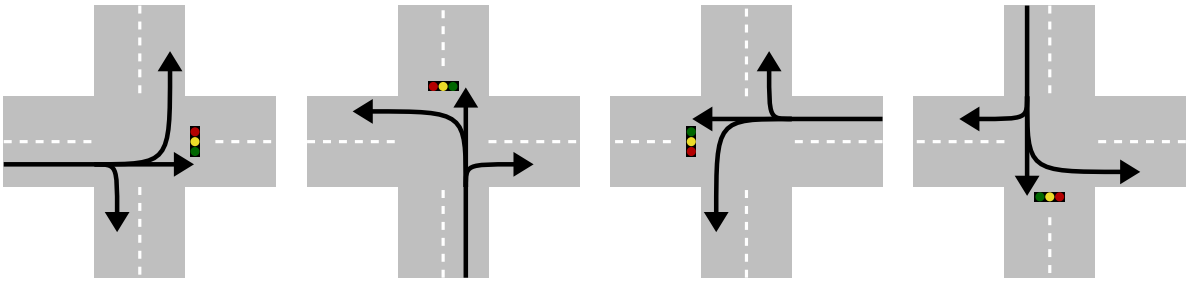
Intersección B: 4 avenidas con 2 fases. Se entiende un cruce de dos calles con doble sentido, donde el semáforo se encarga de ceder el paso a dos de ellas a la vez en cada fase, tal y como se ve en la figura siguiente. Normalmente se permite la vuelta a la izquierda con precaución.



(a) Fase 1 (b) Fase 2

Figura 5.4: Intersección de 4 avenidas y 2 fases

Intersección A: 4 avenidas con 2 fases. Se entiende un cruce de dos calles con doble sentido, donde el semáforo se encarga de ceder el paso a solo una de ellas en cada fase, tal y como se ve en la figura siguiente.



(a) Fase 1 (b) Fase 2 (c) Fase 3 (d) Fase 4

Figura 5.5: Intersección de 4 avenidas y 4 fases.

Para cada una de las intersecciones se suministra un conjunto de valores de prueba. Los valores inferidos por el algoritmo son presentados a manera de tabla en las siguientes secciones.

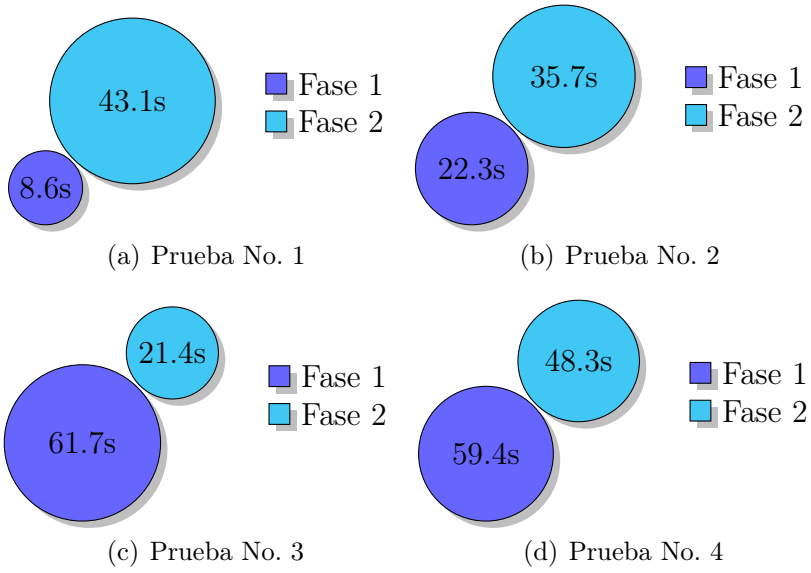
5.2.1. Intersección de 2 avenidas y 2 fases

La siguiente tabla muestra los valores (aleatorios) de entrada suministrados para cada una de las 4 pruebas realizadas, cada uno de los valores representa la cantidad de autos por avenida. Además, se observa el número de carriles de cada avenida y las avenidas que intervienen en cada *fase en verde*.

No	**Vehículos**		**Tiempos asignados**	
	Avenida 1 (1 carril)	Avenida 2 (1 carril)	Fase 1 (av 1)	Fase 2 (av 2)
1	1	8	8.58	43.09
3	4	7	22.29	35.74
2	12	5	61.69	21.40
4	12	10	59.39	48.29

Cuadro 5.1: Valores de prueba para la intersección A

Las siguientes gráficas representan los repartos de tiempo para cada conjunto de valores de prueba. A diferencia de una asignación estática, aquí se observa como los tiempos varían en función de la cantidad de vehículos.



(a) Prueba No. 1 (b) Prueba No. 2

(c) Prueba No. 3 (d) Prueba No. 4

Figura 5.6: Repartos de tiempo de la intersección A

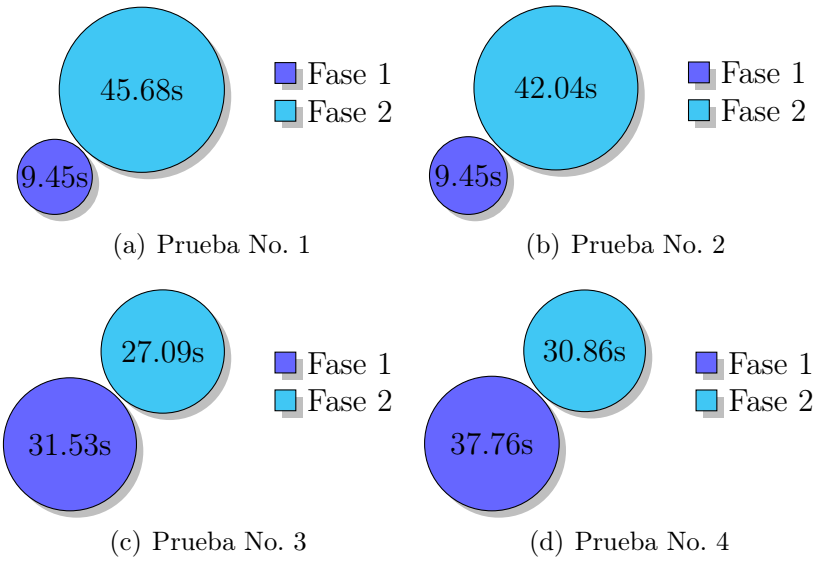
5.2.2. Intersección de 4 avenidas y 2 fases

Al igual que en la sección anterior, se observa los valores de prueba suministrados y los tiempos inferidos por el sistema de inferencia, también se observa el número de carriles de cada avenida y las avenidas que intervienen en cada *fase en verde*.

No	**Vehículos**				**Tiempos asignados**	
	Avenida 1 (2 carriles)	Avenida 2 (1 carril)	Avenida 3 (2 carriles)	Avenida 4 (1 carril)	Fase 1 (avs 1 y 3)	Fase 2 (avs 1 y 4)
1	1	8	8	10	9.45	45.68
3	2	4	8	12	9.45	42.04
2	20	10	12	5	31.53	27.09
4	30	12	7	5	37.76	30.86

Cuadro 5.2: Valores de prueba para la intersección B

Las siguientes gráficas representan los repartos de tiempo para cada conjunto de valores de prueba.



(a) Prueba No. 1 (b) Prueba No. 2

(c) Prueba No. 3 (d) Prueba No. 4

Figura 5.7: Repartos de tiempo de la intersección B

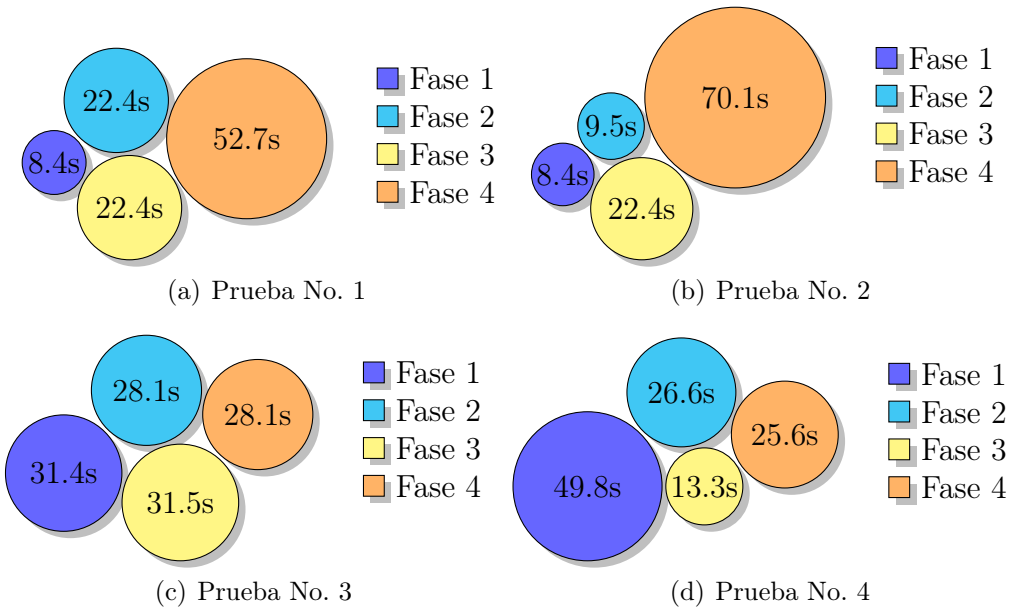
5.2.3. Intersección de 4 avenidas y 4 fases

Al igual que en las secciones anteriores, se observa los valores de prueba suministrados y los tiempos inferidos, también se observa el número de carriles de cada avenida y las avenidas que intervienen en cada *fase en verde*.

No	**Vehículos**				**Tiempos asignados**			
	Avenida 1 (3 carriles)	Avenida 2 (2 carriles)	Avenida 3 (2 carriles)	Avenida 4 (1 carril)	Fase 1 (av 1)	Fase 2 (av 2)	Fase 3 (av 3)	Fase 4 (av 2)
1	2	8	8	10	08.44	22.43	22.43	52.65
3	2	4	8	12	08.44	09.45	22.43	70.07
2	20	10	12	5	31.42	28.09	31.52	28.09
4	30	12	7	5	49.76	26.63	13.28	25.59

Cuadro 5.3: Valores de prueba para la intersección C

El reparto de tiempos es el siguiente:



(a) Prueba No. 1

(b) Prueba No. 2

(c) Prueba No. 3

(d) Prueba No. 4

Figura 5.8: Repartos de tiempo de la intersección C

Capítulo 6

Conclusiones

6.1. Conclusiones del proyecto

La Inteligencia Artificial juega un papel cada vez mas importante en nuestras vidas, desde los asistentes personales en nuestros móviles, hasta la optimización de procesos en la industria. Dentro de las diversas técnicas de IA, la Lógica Difusa, permite modelar situaciones del mundo real de una manera elegante, sencilla y fácil.

En este proyecto se abordó el problema de la congestión en los cruces semáforizados. Una de las principales causas de la congestión es la configuración de tiempos estáticos que a menudo "desperdicia el tiempo" en avenidas donde la cantidad de autos es nula o mínima.

La solución propuesta fue un Sistema de Inferencia Difusa que asignara tiempos de acuerdo a la cantidad de vehículos en las avenidas que se intersectan. El sistema fue capaz de reaccionar de manera adecuada frente a diversas situaciones.

Cabe destacar que durante el diseño del FIS, no fue esencial conocer a fondo el modelo matemático de los semáforos, siendo esto uno de los principales atractivos de esta técnica de IA. Además, permitió definir el problema en jerga común, esto es, en términos coloquiales donde la imprecisión dificulta la asignación de rangos bien definidos a dichos términos.

En conclusión, la técnica seleccionada permitió dar solución al problema de una manera acertada.

6.2. Recomendaciones

Aún queda un largo camino por recorrer, un proyecto multidisciplinario como este, necesita de diferentes expertos en diferentes áreas para tratar algunos detalles finos. Las recomendaciones que emitimos en pro de mejorar la eficacia del algoritmo son las siguientes:

Acerca del ciclo del semáforo. Si bien, el sistema es capaz de asignar tiempos de manera dinámica en respuesta a la cantidad de vehículos en la intersección, sugerimos analizar la posibilidad de añadir una capa extra al sistema. Una capa encargada de determinar la longitud del ciclo. La salida de dicha capa extra sería un factor de multiplicación para alargar o acortar la duración del ciclo, y sus entradas podrían ser la longitud media de la cola de vehículos u otros factores como la hora pico o el clima, factores que afectan el tiempo de reacción de los automovilistas.

Acerca del algoritmo de visión computacional. Para el conteo de automóviles en las avenidas, se recomienda el uso del algoritmo propuesto en [6] cuya implementación en C++ facilitamos en el apéndice B por motivos de eficiencia y compatibilidad.

Dicho trabajo arroja un algoritmo implementado en *Python* que, según los resultados de la propia investigación, ha probado tener una eficacia (con buena iluminación) de hasta un 95 %.

El tiempo de ejecución del algoritmo sugerido por ESTRADA & RECINOS & VIDAL (2017): "Detección de vehículos mediante imagen de fondo " es de 1.5 segundos (implementado en Python), sin embargo al implementarse en C++ alcanza un tiempo de 0.03 segundos, es decir, 3 centésimas de segundo. Por lo que se recomienda el uso de este algoritmo implementado en C++.

Mejoras: A tal algoritmo se le recomienda agregar una fase donde se compense la perspectiva desde la cual se tome la foto, ya que debido a esto, los autos más lejanos podrían no cubrir la cuota de pixeles mínima y por ende pasarían desapercibidos para el semáforo.

Apéndice A

Diagramas de clases UML

En la sección 4.4.3, página 69 se mostró el siguiente diagrama UML con elementos omitidos, ahora se presentarán el resto de diagramas sin elementos omitidos.

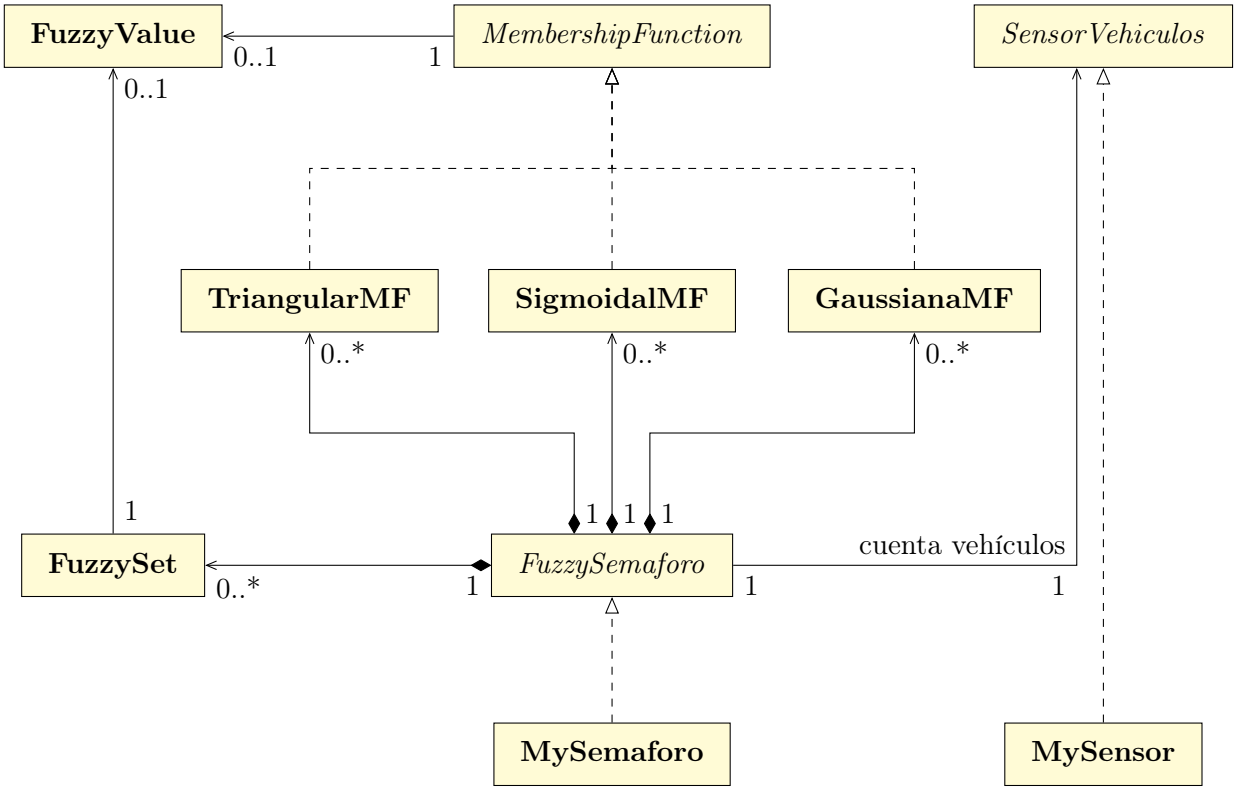


Figura A.1: Diagrama de clases que muestra las relaciones del sistema

A.1. Diagramas de las clases *FuzzySet y FuzzyValue*

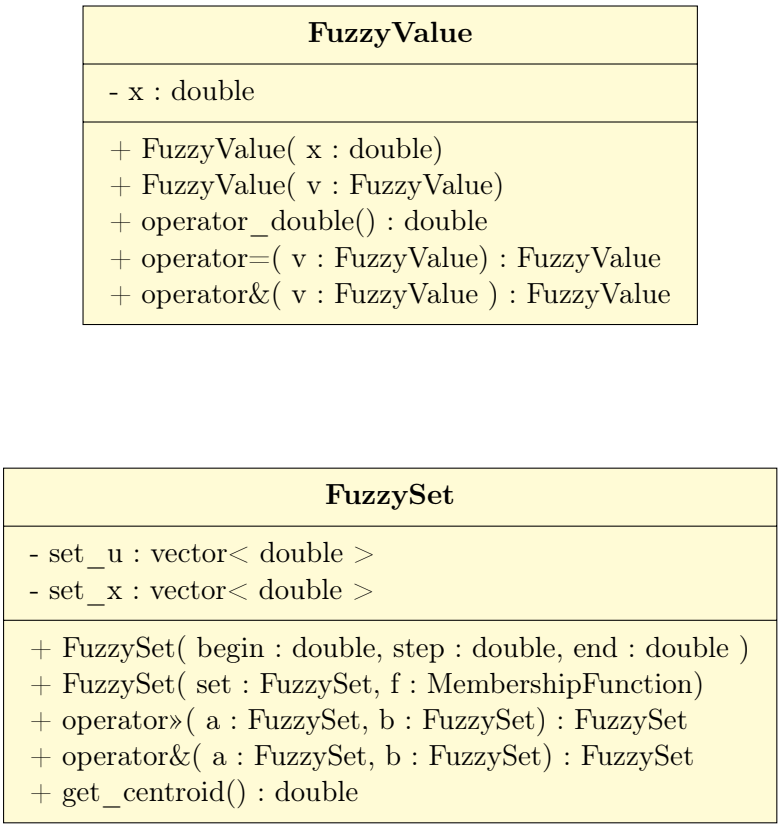


Figura A.2: Diagrama de las clases *FuzzySet* y *FuzzyValue*

A.2. Jerarquía de herencia *MembershipFunction*

El siguiente diagrama UML modela los miembros de las clases *MembershipFunction, TriangularMF, SigmoidalMF, GaussianaMF*, además muestra su relación de herencia.

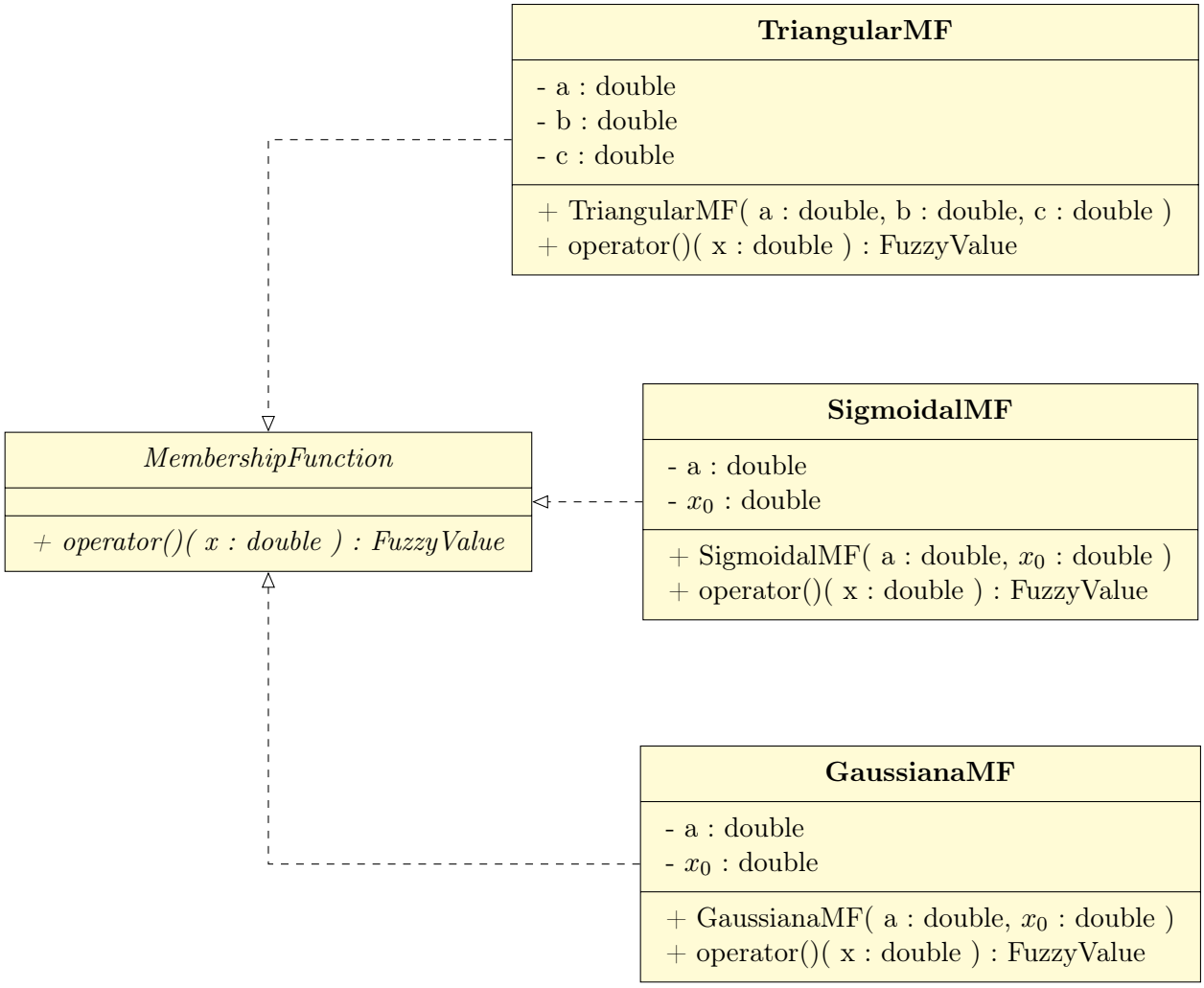


Figura A.3: Diagrama de clases que modela la jerarquía de herencia MembershipFunction

A.3. Realización de la clase *SensorVehiculos*

El siguiente diagrama modela los elementos omitidos de las clases *SensorVehiculos* y *MySensor* del diagrama 4.16, además, se modela su relación de herencia y *realización*.

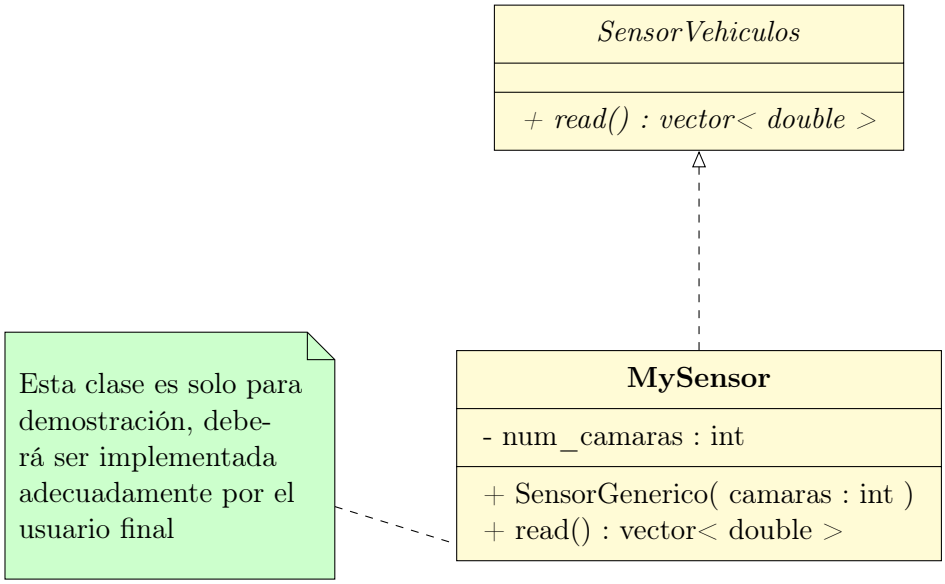


Figura A.4: Diagrama de clases que modela la implementación de *SensorVehiculos:read()*

A.4. Realización de la clase *FuzzySemaforo*

El siguiente diagrama modela los elementos omitidos de las clases *FuzzySemaforo* y *MySemaforo* del diagrama 4.16, además, se modela su relación de herencia y *realización*.

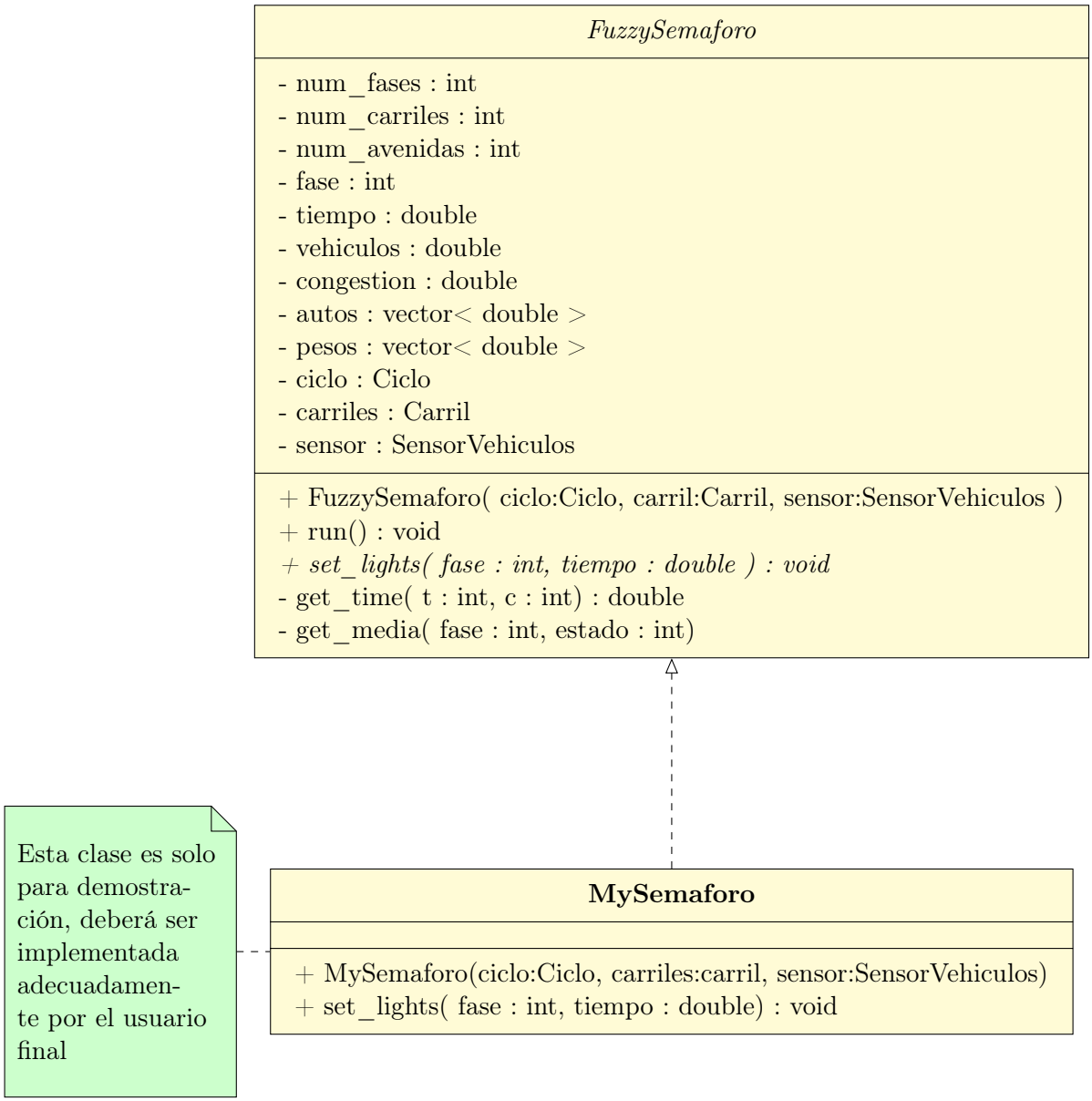


Figura A.5: Diagrama de clases que modela la implementación de *set_lights: FuzzySemaforo*

Apéndice B

Implementación en C++ de un algoritmo de detección de vehículos

Del resultado de la investigación realizada en [6] se rescata el algoritmo de visión computacional seleccionado para la detección de automóviles. Dicho trabajo arroja un algoritmo implementado en *Python* que, según los resultados de la propia investigación, ha probado tener una eficacia (con buena iluminación) de hasta un 95 %.

El tiempo de ejecución del algoritmo sugerido por *Estrada & Recinos & Vidal (2017)*: "Detección de vehículos mediante imagen de fondo " es de 1.5 segundos (implementado en Python), sin embargo al implementarse en C++ alcanza un tiempo de 0.03 segundos, es decir, 3 centésimas de segundo.

Requerimientos

El algoritmo en su versión C++ requiere de un compilador que soporte el estándar ISO C++11 además de tener compilado e instalado la librería para visión por computadora OpenCV en su versión 2.x.

Código Fuente

```
1 #include <opencv2/highgui/highgui.hpp>
2 #include <opencv2/imgproc/imgproc.hpp>
3 #include <iostream>
4 #include <ctime>
```

```
5  #include <windows.h>
6
7  //#define DEBUG
8
9
10 #define IMGW 640
11 #define IMGH 480
12
13 #define BLUR 3
14 #define THRESHOLD 37
15 #define DILATE 4
16
17
18 using namespace std;
19 //using namespace cv;
20 int main( int argc, char** argv ) {
21
22
23   /*
24   * Variables para la medición del tiemp de ejecución
25   */
26
27   double    time = 0;
28   unsigned long  tick = 0;
29   unsigned long tock = 0;
30
31
32
33   /*
34   * Variables necesarias para la manipulacion de las imagenes
35   */
36
37   cv::Mat foreground;  // Captura actual
38   cv::Mat background;  // Carretera de fondo
39
40   cv::Mat fg_rsz;      // Captura escalada
41   cv::Mat bg_rsz;      // Carretera escalada
42
43   cv::Mat fg_blr;      // Captura desenfocada
44   cv::Mat bg_blr;      // Carretera desenfocada
45
46   cv::Mat img_dif;     // Resultado de aplicar diferencia absoluta a las
       tomas
47   cv::Mat img_thd;     // Resultado de aplicar threshold
48   cv::Mat img_dlt;     // Resultado de la dilatacion
49
50
51   /*
52    * Variables necesarias para la detección y procesamiento de los
        contornos
53    */
54
```

```
55  vector< vector< cv::Point > > contours;
56  vector< cv::Vec4i > hierarchy;
57  cv::Mat element;
58
59  // toma de tiempo inicial
60  tick = clock();
61
62  // Carga de la imagen y transformacion a escala de grises
63  // Para obtener las imagenes desde una camara se recomienda usar:
64  // cv::createFileCapture( ID_CAMARA )
65  foreground = cv::imread("foreground.png" , CV_LOAD_IMAGE_GRAYSCALE);
66  background = cv::imread("background.png" , CV_LOAD_IMAGE_GRAYSCALE);
67
68  if( !foreground.data || !background.data )
69  {
70      std::cout <<  "Alguna de las imagenes no se encuentra o no puede
        abrirse" << std::endl ;
71      return -1;
72  }
73
74  #ifdef DEBUG
75  cv::namedWindow( "Display window", cv::WINDOW_AUTOSIZE );
76  #endif
77
78  // Redimencionamiento de la imagen
79  cv::resize(foreground, fg_rsz, cv::Size(IMGW,IMGH));
80  cv::resize(background, bg_rsz, cv::Size(IMGW,IMGH));
81
82  #ifdef DEBUG
83  cv::imshow( "Display window", fg_rsz ); cv::waitKey(0);
84  #endif
85
86  // Desenfoque gausiano para suavisar los bordes
87  cv::blur(fg_rsz, fg_blr, cv::Size(BLUR,BLUR) ); //cv::GaussianBlur(
        fg_rsz, fg_blr, cv::Size(21,21), 0);
88  cv::blur(bg_rsz, bg_blr, cv::Size(BLUR,BLUR) ); //cv::GaussianBlur(
        bg_rsz, bg_blr, cv::Size(21,21), 0);
89
90  #ifdef DEBUG
91  cv::imshow( "Display window", fg_rsz ); cv::waitKey(0);
92  #endif
93
94
95  // Diferencia absoluta para detectar las direnecias entre las dos
        imagenes
96  cv::absdiff(fg_blr, bg_blr, img_dif);
97
98  #ifdef DEBUG
99  cv::imshow( "Display window", img_dif ); cv::waitKey(0);
100 #endif
101
102
```

```
103   // Aplica threshold para crear una imagen binaria sin escalas de grises
104   cv::threshold( img_dif, img_thd, THRESHOLD, 255, cv::THRESH_BINARY);
105
106   #ifdef DEBUG
107   cv::imshow( "Display window", img_thd ); cv::waitKey(0);
108   #endif
109
110
111   // Dilata la imagen para unificar los contornos adyacentes
112   element = cv::getStructuringElement( cv::MORPH_RECT, cv::Size(DILATE
          *2+1,DILATE*2+1), cv::Point(DILATE,DILATE) );
113   cv::dilate(img_thd, img_dlt, element);
114
115   #ifdef DEBUG
116   cv::imshow( "Display window", img_dlt ); cv::waitKey(0);
117   #endif
118
119
120   // Busca los contornos presentes en la imagen
121   cv::findContours( img_dlt, contours, hierarchy, cv::RETR_TREE, cv::
          CHAIN_APPROX_SIMPLE, cv::Point(0,0) );
122
123   #ifdef DEBUG
124   vector< vector< cv::Point > > contours_poly( contours.size() );
125   vector< cv::Rect > boundRect( contours.size() );
126   #endif
127
128   //vector<Point2f> center( contours.size() );
129   //vector<float> radius( contours.size() );
130   int cont = 0;
131
132   for( int i = 0; i < contours.size(); ++i )
133   {
134
135     //cv::minEnclosingCircle( (Mat)contours_poly[i], center[i], radius[i]
             );
136
137     auto c = contours[ i ];
138
139     if( cv::contourArea( c ) > 12000 )
140     {
141       ++cont;
142
143       #ifdef DEBUG
144
145       cv::Scalar color( 255, 000, 000);
146       //cv::drawContours( fg_rsz, contours, i, color, 2, 8, hierarchy, 0 ,
              Point() );
147
148       cv::approxPolyDP( cv::Mat(contours[i]), contours_poly[i], 3, true);
149       boundRect[ i ] = cv::boundingRect( cv::Mat(contours_poly[i]) );
150       cv::rectangle( fg_rsz, boundRect[i].tl(), boundRect[i].br(), color,
```

```
            2, 8, 0);
151
152         //cv::circle( fg_rsz, center[i], (int)radius[i], color, 2, 8, 0);
153         #endif
154
155       }
156
157     }
158
159
160     tock = clock();
161     time = ((double)(tock-tick)/CLOCKS_PER_SEC);
162
163     std::cout << "time: " << time << ", clocks per sec: "<< CLOCKS_PER_SEC
          <<std::endl;
164     std::cout << "contornos: " << cont << std::endl;
165
166
167
168     #ifdef DEBUG
169     cv::imshow( "Display window", fg_rsz );
170     cv::waitKey(0);
171     #endif
172     //cv::imshow( "Display window", dst );
173     //cv::waitKey(0);
174
175
176
177     return 0;
178 }
```

Listing B.1: Implementación en C++ del algoritmo de detección de vehículos

Bibliografía

[1] García, G. (2017). Un enfoque de semáforo inteligente utilizando algoritmos de visión computacional en una intersección aislada para optimizar el flujo vehicular (Tesis de maestría). Centro de Investigación en Inteligencia Artificial (CIIA) de la Universidad Veracruzana, Xalapa de Enríquez, Veracruz, México.

[2] Alvaro E., R. De Somocurcio S. (2008). Control de tráfico vehicular automatizado utilizando Lógica Difusa. Universidad Ricardo Palma, Lima, Perú.

[3] Bances, M., Ramos, M. (2014). Semáforos Inteligentes para la regulación del tráfico vehicular. Rev. Ingeniería: Ciencia, Tecnología e Innovación. Vol. 1 (No. 1). pp. 37-45. .

[4] Morales, L. Rafael., Gonzáles, S. Juan. (2013) *Control de tráfico vehicular por medio de semáforos inteligentes*. Universidad de Rafael Urdaneta, República Bolivariana de Venezuela.

[5] Hernández, C.A., Salcedo, O., & Pedraza, L.F. (2007). Modelo de Semaforización Inteligente para la Ciudad de Bogotá. Revista Científica y Tecnológica de la Facultad de Ingeniería, Universidad Distrital Francisco José de Caldas, Vol. 11(No.2). 61-69.

[6] Estrada, E.K., Recinos, H., & Vidal, G. Y. (2017). Selección de un algoritmo de visión por computadora para la detección de vehículos. Instituto Tecnológico de Tapachula, Tapachula, Chiapas, México.

[7] Oscar G. Duarte (1999). Sistemas de lógica difusa. Fundamentos. *Revista Ingeniería e Investigación, No. 42, pp. 22.*

[8] Secretaría de Desarrollo Social (SEDESOL). (1994). Programa de Asistencia técnica en transporte urbano para las ciudades medias mexicanas. Manual Normativo, Tomo XII. Estudios de Ingeniería de Tránsito. México.

[9] Ministerio de Transporte e Infraestructura. (2008). Manual para la Revisión de Estudios de Tránsito. Realización de Manuales Técnicos para la Revisión y Aprobación de Estudios y Diseños de Carreteras. Managua, Nicaragua.

[10] Juan Gabriel Tapia Arandia, Romel Daniel Veizaga Balta (2006). *Apoyo didáctico para la enseñanza y aprendizaje de la asignatura de ingeniería de tráfico*. (Trabajo para optar al diploma de Licenciatura en Ingeniería Civil). Universidad Mayor De San Simón, Cochabamba, Bolivia.

[11] Zadeh, A.L. (1965). Fuzzy sets *Information and Control*, vol. 8, pp. 338-353.

[12] Zadeh, A.L. (1973). Outline of a New Approach to the Analysis of Complex Systems and Decision Processes. *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-3, (No. 1). 28-44.

[13] Zadeh, A.L. (1975). The Concept of a Linguistic Variable and its Application to Approximate Reasoning-I. *Information Sciences*. Vol. 8. 199-249.

[14] Ponce Cruz P. (Primera Edición). (2010). *Inteligencia Artificial con Aplicaciones a la Ingeniería*. México: Alfaomega.

[15] Amador Hidalgo L. (Primera Edición). (1997). *Inteligencia Artificial y Sistemas Expertos*. Córdoba: Universidad de Córdoba.

[16] Alfonso V. Rivera. (2013). *Controladores difusos aplicados a convertidores DC/DC* (Tesis de maestría). Universidad Autónoma de Aguascalientes, Aguascalientes, Ags.

[17] Carlos G. Morcillo. *Lógica Difusa, una introducción práctica*. E-mail: Carlos.Gonzales@uclm.es

[18] José Carlos. *Control Neuro-Difuso Aplicado a una Grúa Torre*. Recuperado de Tesis Digitales UNMSM

[19] Kevin M. Passino, Stephen Yurkovick, (1997). *Fuzzy Control*. California, Berkeley: ADDISON-WESLEY

[20] Marín, M. R., Palma, M. J. (2008). Inteligencia Artificial: Métodos, técnicas y aplicaciones. España: Mcgraw-Hill/Interamericana De España, S. A. U.

[21] Isasi, V. P., Galván, L. I. (2004). Redes de Neuronas Artificiales: Un enfoque práctico. Madrid, España: Pearson Educación, S.A.