Universidad Técnica Particular de Loja

La Universidad Católica de Loja

ÁREA TÉCNICA

TÍTULO DE INGENIERO EN ELECTRÓNICA Y TELECOMUNICACIONES

Diseño e implementación de una solución para el conteo de vehículos a partir de imágenes aéreas.

TRABAJO DE TITULACIÓN.

AUTOR: Pérez Santín, Cristian Fernando.

DIRECTOR: Aguirre Reyes, Daniel Fernando, PhD.

LOJA – ECUADOR

2017

Septiembre, 2017

APROBACIÓN DEL DIRECTOR DEL TRABAJO DE TITULACIÓN

PhD.

Daniel Fernando Aguirre Reyes.

DOCENTE DE LA TITULACIÓN

De mi consideración:

El presente trabajo de titulación**: "Diseño e implementación de una solución para el conteo de vehículos a partir de imágenes aéreas"** realizado por **Pérez Santín Cristian Fernando** ha sido orientado y revisado durante su ejecución, por cuánto se aprueba la presentación del mismo.

Loja, abril de 2017.

f) ……

DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS

Yo, **Cristian Fernando Pérez Santín**, declaro ser autor del presente trabajo de titulación: “Diseño e implementación de una solución para el conteo de vehículos a partir de imágenes aéreas”, de la Titulación de Electrónica y Telecomunicaciones, siendo **Daniel Fernando Aguirre Reyes** director del presente trabajo; y eximo expresamente a la Universidad Técnica Particular de Loja y a sus representantes legales de posibles reclamos o acciones legales. Además, certifico que las ideas, conceptos, procedimientos y resultados vertidos en el presente trabajo investigativo, son de mi exclusiva responsabilidad.

Adicionalmente declaro conocer y aceptar la disposición del Art. 88 del Estatuto Orgánico de la Universidad Técnica Particular de Loja que en su parte pertinente textualmente dice: “Forman parte del patrimonio de la Universidad la propiedad intelectual de investigaciones, trabajos científicos o técnicos y tesis de grado o trabajos de titulación que se realicen con el apoyo financiero, académico o institucional (operativo) de la Universidad”.

f. ……

Autor: Cristian Fernando Pérez Santín

Cédula: 1104744071

DEDICATORIA

A mis padres Rodrigo y Germania que siempre han estado ahí apoyándome incondicionalmente para seguir adelante y por forjar en mí ese espíritu de perseverancia para culminar mi carrera universitaria.

A mis hermanos, la pequeña Belén e Israel por brindarme todo su cariño.

A mi abuelita que siempre está ahí para mí y a mis demás abuelitos que, aunque ya no estén aquí presentes, se que están muy orgullosos desde donde quiera que se encuentren.

A todos quienes de alguna manera supieron estar ahí brindándome sus consejos y todo su apoyo incondicional para salir adelante.

Cristian Fernando

AGRADECIMIENTO

A todos aquellos que han aportado con su granito de arena para la realización del presente trabajo, especialmente a Daniel Aguirre que, gracias a su paciencia y colaboración, ha permitido que esto sea posible.

A Belén, que siempre ha estado ahí en las buenas y las malas, brindándome todo su cariño e impulsándome a ser una mejor persona.

A doña Lucy, que me abrió las puertas de su hogar en toda esta etapa universitaria y por sus consejos.

A mis amigos, por estar presentes en todo momento.

Gracias totales.

ÍNDICE DE CONTENIDOS

CARÁTULA i
APROBACIÓN DEL DIRECTOR DEL TRABAJO DE FIN DE TITULACIÓN ii
DECLARACIÓN DE AUTORÍA Y CESIÓN DE DERECHOS iii
DEDICATORIA iv
AGRADECIMIENTO v
ÍNDICE DE CONTENIDOS vi
ÍNDICE DE FIGURAS x
ÍNDICE DE TABLAS xii
RESUMEN 1
ABSTRACT 2
INTRODUCCIÓN 3
CAPÍTULO I. ANTECEDENTES 4
1.1. Planteamiento del problema 5
1.2. Objetivos 5
1.2.1. Objetivo general. 5
1.2.2. Objetivos específicos. 5
1.3. Justificación 5
1.4. Alcance y limitaciones 6
1.5. Metodología 6
CAPÍTULO II. ESTADO DEL ARTE 8
2.1. Conceptos generales 9
2.2.1. Procesamiento de imágenes digitales. 9
2.2.1.1. Imagen digital. 9
2.2.1.2. Adquisición de imágenes. 10
2.2.1.3. Pre-procesamiento. 10
2.2.1.4. Segmentación. 10
2.2.1.5. Transformaciones morfológicas. 10
2.2.1.6. Representación y descripción. 11
2.2.1.7. Reconocimiento e interpretación. 11
2.2.2. Algoritmos detectores y descriptores. 11
2.2.2.1. SIFT: “Scale-invariant Feature Transform”. 12
2.2.2.1.1. Detección de extremos en el espacio-escala. 12
2.2.2.1.2. Localización de puntos de interés. 13

2.2.2.1.3. Asignación de orientación. 13
2.2.2.1.4. Descriptor de puntos clave. 14
2.2.2.2. SURF: “Speeded Up Robust Features”. 14
2.2.2.2.1. Detección de puntos de interés. 15
2.2.2.2.2. Asignación de la orientación. 16
2.2.2.2.3. Extracción de descriptores. 16
2.2.2.3. ORB: “Oriented FAST and Rotated BRIEF”. 17
2.2.2.3.1. oFAST: “FAST Keypoint Orientation”. 18
2.2.2.3.2. rBRIEB: “Rotation-Aware Brief”. 18
2.2.2.4. ASIFT. 19
2.2.2.4.1. Afinación de la aproximación local. 19
2.2.2.4.2. Descripción. 20
2.2.2.4.3. Correspondencia. 21
2.2.2.5. Detector de MSER. 21
2.2.3. Homografía. 22
2.2.4. Modelos de estimación de parámetros. 22
2.2.4.1. RANSAC. 22
2.2.4.2. MSAC. 22
2.2.4.3. MLESAC. 23
2.2.4.4. PROSAC. 23
2.2.4.5. MINPRAN. 23
2.2.5. Semáforos inteligentes. 24
2.2.5.1. Tecnologías. 24
2.2.5.1.1. Semáforo inteligente con RFID. 24
2.2.5.1.2. Semáforo inteligente con redes de sensores inalámbricos. 25
2.2.5.1.3. Semáforo inteligente mediante procesamiento de imágenes. 26
2.2.5.1.4. Semáforo inteligente basado en inteligencia artificial. 27
2.2.6. Algoritmos de búsqueda de la ruta más corta. 27
2.2.6.1. Algoritmo de Dijkstra. 27
2.2.6.2. Algoritmo BFS: “Breadth-First Search”. 28
2.2.6.3. Algoritmo Bellman-Ford. 28
2.2.6.4. Algoritmo DFS: “Depth-First Search”. 29
2.3. Trabajos relacionados. 29
2.3.1. Técnicas para la creación de vistas panorámicas tipo mosaico. 29
2.3.1.1. Creación de vistas panorámicas utilizando SIFT. 29

2.3.1.2. Creación de vistas panorámicas utilizando SURF. 34
2.3.2. Técnicas de detección de vehículos. 36
2.3.3. Técnicas para la implementación de semáforos inteligentes. 39
CAPÍTULO III. MATERIALES Y MÉTODOS 40
3.1. Análisis de requerimientos 41
3.1.1. Programas. 41
3.1.2. Equipos físicos. 41
3.2. Elección del programa 42
3.3. Elección de los elementos físicos 42
3.4. Ubicación del vehículo aéreo no tripulado y semáforos 43
CAPÍTULO IV. DESARROLLO E IMPLEMENTACIÓN 44
4.1. Descripción general del sistema 45
4.2. Construcción de una vista panorámica tipo mosaico 45
4.3. Diseño de una interfaz gráfica de usuario en MATLAB 48
4.3.1. Selección del escenario. 48
4.3.2. Segmentación de la imagen 50
4.3.3. Conteo vehicular. 54
4.3.4. Proceso de semaforización. 56
4.3.5. Cálculo y presentación del porcentaje de congestión en cada calle. 57
4.3.6. Acercamiento en cada intersersección. 59
4.3.7. Búsqueda de la ruta con menor congestión. 60
CAPITULO V. EXPERIMENTOS Y RESULTADOS 62
5.1. Descripción de los experimentos realizados 63
5.1.1. Resultados de la creación de la vista panorámica tipo mosaico. 63
5.1.2. Resultados de la contabilización de vehículos por cada calle. 66
5.2. Tablas y gráficas de resultados 71
5.2.1. Puntos característicos SIFT detectados. 71
5.2.2. Número de correspondencias calculadas. 72
5.2.3. Resultados con los tipos de interpolación. 74
5.2.4. Resultados del conteo vehicular. 79
5.2.5. Tiempo de procesamiento al realizar acercamiento en cada intersección. 82
5.2.6. Tiempo de procesamiento de la búsqueda la ruta con menor congestión. 82
CAPITULO VI. DISCUSIÓN 84
6.1. Análisis de la inversión y costos 85

6.1.1. Costos de implementación del proyecto. 85
6.1.2. Costos de inversión. 85
6.1.2.1. Costo de equipos. 85
6.1.2.1.1. Solución mediante la utilización del drone DJI Phanthom 3 Standard. 86
6.1.2.1.2. Solución mediante la utilización del drone DJI Mavic PRO. 87
6.1.2.1.3. Solución mediante la utilización del drone DJI Phanthom 4 Pro. 89
6.1.2.1.4. Solución mediante la utilización de globos cautivos con helio. 90
6.1.2.2. Costos de infraestructura. 92
6.1.2.3. Costos de instalación. 92
6.1.2.4. Costos de ingeniería. 92
6.1.3. Costos operativos. 93
6.1.4. Costos totales. 94
6.1.5. Análisis de las soluciones planteadas. 94
6.2. Factibilidad del proyecto 95
6.2.1. Técnica. 95
6.2.2. Económica. 96
6.3. Ventajas y desventajas del proyecto en su desarrollo 96
6.4. Objetivos futuros 97
CONCLUSIONES 98
RECOMENDACIONES 99
BIBLIOGRAFÍA 100
ANEXOS 105
ANEXO 1. CÓDIGO PARA LA CREACIÓN DE VISTAS PANORÁMICAS TIPO “MOSAICO” 106
ANEXO 2. CÓDIGO PARA LA IMPLEMENTACIÓN DE LA SOLUCIÓN PARA EL CONTEO VEHICULAR A PARTIR DE IMÁGENES AÉREAS 111

ÍNDICE DE FIGURAS

Figura 1. Fases para el desarrollo del proyecto. 7
Figura 2. Detección de extremos del espacio-escala 13
Figura 3. a) Gradiente de la imagen y b) Descriptor del punto clave 14
Figura 4. Derivada parcial de segundo orden del filtro gaussiano discretizado y aproximación de la derivada por el descriptor SURF 15
Figura 5. Vector de orientación y sus respuestas obtenidas a partir de las funciones de Haar. 16
Figura 6. Composición de la característica del descriptor 17
Figura 7. Esquema de funcionamiento ASIFT. 20
Figura 8. Muestreo espacial óptimo para ASIFT. 21
Figura 9. Esquema de un semáforo inteligente con tecnología RFID. 25
Figura 10. Esquema de un semáforo inteligente con tecnología WSN. 26
Figura 11. a) Características para la correspondencia luego de aplicar RANSAC y b) Mosaico de imágenes. 30
Figura 12. Secuencia de imágenes para formar el mosaico. 31
Figura 13. a) Correspondencias con SIFT. b) Mosaico resultante. 31
Figura 14. Mosaico con el método SIFT mejorado. 32
Figura 15. Mosaico con el método SIFT adaptativo. 32
Figura 16. a) Puntos característicos extraídos mediante L^2-SIFT b) Correspondencias entre ambas imágenes. 33
Figura 17. a) Par de imágenes y b) Mosaico creado con SURF. 35
Figura 18. Resultado final de la detección de vehículos. 37
Figura 19. Detección de vehículos usando: a) Método V-J original b) Método mejorado V-J 38
Figura 20. Detección de vehículos usando bajas condiciones de iluminación. 38
Figura 21. Fases del proyecto. 41
Figura 22. Método para la elaboración del mosaico. 48
Figura 23. Aplicativo GUI en MATLAB 49
Figura 24. Mapa del casco urbano (escenario 4) y semáforos en la GUI de MATLAB. 49
Figura 25. Mosaico de imágenes seleccionado y cargado en la GUI. 50
Figura 26. Mosaico de imágenes pre-procesado. 51
Figura 27. Uso de la apertura morfológica aplicado al mosaico de imágenes. 51
Figura 28. Aumento del contraste de la imagen 52
Figura 29. Imagen binaria sin procesar. 52
Figura 30. Imagen binaria sin procesar. 53
Figura 31. Imagen erosionada. 54
Figura 32. Imagen procesada 54
Figura 33. Conteo de vehículos de la ROI seleccionada 55
Figura 34. Vehículos a contabilizar por cada calle 55
Figura 35. Proceso de semaforización en cada intersección de calles del casco urbano 56
Figura 36. Proceso de presentación del porcentaje de congestión en cada calle. 58
Figura 37. Presentación del porcentaje de disponibilidad en cada calle. 59
Figura 38. Vista aérea de cada intersección de calles. 60
Figura 39. Presentación del mensaje de rutas disponibles en la GUI de MATLAB. 61
Figura 40. Presentación de rutas con menor congestión cargado en la GUI de MATLAB. 61
Figura 41. a) Puntos característicos detectados en la primera y segunda imagen. b) Correspondencias entre ambas imágenes. 63

Figura 42. Primera fusión de imágenes para formar el mosaico. 64
Figura 43. Puntos característicos detectados en la primera fusión y tercera imagen. 64
Figura 44. Segunda fusión de imágenes para formar el mosaico. 65
Figura 45. Puntos característicos detectados en la segunda fusión y tercera imagen. 65
Figura 46. Mosaico final resultante. 66
Figura 47. Disponibilidad de calles en el primer escenario. 66
Figura 48. Disponibilidad de calles en el segundo escenario. 67
Figura 49. Disponibilidad de calles en el tercer escenario. 67
Figura 50. Disponibilidad de calles en el cuarto escenario. 68
Figura 51. Disponibilidad de calles en el quinto escenario. 68
Figura 52. Disponibilidad de calles en el sexto escenario. 69
Figura 53. Disponibilidad de calles en el séptimo escenario. 69
Figura 54. Disponibilidad de calles en el octavo escenario. 70
Figura 55. Disponibilidad de calles en el noveno escenario. 70
Figura 56. Disponibilidad de calles en el décimo escenario. 71
Figura 57. Número de correspondencias por canal de color. Imagen reducida al 30 %. 73
Figura 58. Número de correspondencias por canal de color. Imagen reducida al 50 %. 73
Figura 59. Número de correspondencias por canal de color. Imagen reducida al 70 %. 74
Figura 60. Número de correspondencias por canal de color. Imagen reducida al 90 %. 74
Figura 61. Tiempo de procesamiento para la creación del mosaico con interpolación bicúbica. 76
Figura 62. Tiempo de procesamiento para la creación del mosaico con interpolación bilineal. 77
Figura 63. Tiempo de procesamiento para la creación del mosaico con interpolación vecino más cercano 77
Figura 64. Comparación tiempo de procesamiento entre los dos ordenadores con 78
Figura 65. Comparación tiempo de procesamiento entre los dos ordenadores con 78
Figura 66. Comparación tiempo de procesamiento entre los dos ordenadores con interpolación del vecino más cercano. 79
Figura 67. Comparación tiempo de procesamiento para el conteo vehicular. 80
Figura 68. Comparación tiempo de procesamiento entre los ordenadores al realizar el acercamiento al mapa. 82
Figura 69. Tiempo de procesamiento entre los dos ordenadores al presentar ruta descongestionada. 83
Figura 70. Arquitectura primera solución con drone DJI Phanthom 3. 86
Figura 71. Arquitectura primera solución con drone DJI Mavic Pro. 88
Figura 72. Arquitectura primera solución con drone DJI Phanthom 4 Pro. 89
Figura 73. Arquitectura primera solución con globo cautivo. 91

ÍNDICE DE TABLAS

Tabla 1. Resultados de la creación del mosaico de imágenes aplicando RANSAC. 30
Tabla 2. Número de puntos característicos. 30
Tabla 3. Resultados de la creación del mosaico basado en SIFT. 31
Tabla 4. Resultados del algoritmo SIFT mejorado y el algoritmo SIFT tradicional. 32
Tabla 5. Resultados del número de correspondencias calculadas. 33
Tabla 6. Análisis de la precisión del algoritmo L^2-SIFT. 34
Tabla 7. Comparación del rendimiento computacional de SURF. 35
Tabla 8. Comparación del método tradicional SURF con el algoritmo mejorado. 36
Tabla 9. Puntos SIFT detectados para cada par de imágenes 72
Tabla 10. Número de correspondencias calculadas. 72
Tabla 11. Resultados de las pruebas realizadas con interpolación bicúbica. 75
Tabla 12. Resultados de las pruebas realizadas con interpolación bilineal. 75
Tabla 13. Resultados de las pruebas realizadas con interpolación vecino más cercano. 76
Tabla 14. Tiempo de procesamiento para el conteo vehicular 79
Tabla 15. Resultados de precisión del algoritmo en la detección de vehículos 81
Tabla 16. Tiempo de procesamiento para el acercamiento del mapa. 82
Tabla 17. Tiempo de procesamiento para la presentación de la ruta con menor congestión. 83
Tabla 18. Costo de los equipos y accesorios para la primera solución 87
Tabla 19. Costo de los equipos y accesorios para la segunda solución. 88
Tabla 20. Costo de los equipos y accesorios para la tercera solución 89
Tabla 21. Costo de los equipos y accesorios para la cuarta solución 91
Tabla 22. Costos de infraestructura. 92
Tabla 23. Costos de instalación 92
Tabla 24. Costos de ingeniería 93
Tabla 25. Costos operativos para la solución con drones. 93
Tabla 26. Costos operativos para la solución con globos cautivos. 93
Tabla 27. Costos de inversión por cada solución. 94
Tabla 28. Costos operativos anuales por cada solución. 94
Tabla 29. Comparación de características entre las cuatro posibles soluciones. 95

RESUMEN

El presente trabajo tiene como finalidad el diseño e implementación de una solución para el conteo vehicular en horas pico en el casco urbano de la ciudad de Loja, mediante la creación de una vista panorámica tipo mosaico a partir imágenes aéreas captadas a 100 metros de altura usando la cámara que viene incorporada en el vehículo aéreo no tripulado y utilizando el algoritmo descriptor SIFT y el algoritmo RANSAC. Para el diseño del algoritmo de visión por computador se realizó una aplicación en el programa MATLAB (R2016a) con la finalidad de detectar y contabilizar vehículos en diez escenarios distintos mediante visión por computador; permitiendo controlar el tráfico vehicular a través de semaforización inteligente, como también la posibilidad de presentarle al conductor la ruta con menor congestión y de esta manera agilizar el tráfico de las vías públicas.

PALABRAS CLAVES: drone, VANT, procesamiento de imágenes, mosaico, MATLAB, SIFT, RANSAC.

ABSTRACT

The present work has as an aim to design and implement a solution to count vehicles on pick hours at the urban area of Loja's city, using a panoramic mosaic view from aerial imagery captured at 100 meters in height by a camera that is already incorporated in an air vehicle drone using the algorithm descriptor SIFT and the algorithm RANSAC. The algorithm has been made on MATLAB (R2016a) with the purpose of detecting and counting vehicles in ten different scenarios letting it to control the vehicle traffic through a group of smart semaphore, as well the possibility of introduce to the driver a route with less congestion and in this way speed up the traffic of the public ways.

KEYWORDS: drone, VANT, image processing, mosaic, MATLAB, SIFT, RANSAC

INTRODUCCIÓN

En la actualidad se produce un excesivo tráfico vehicular en horas pico en la parte céntrica de la ciudad de Loja, causando un gran embotellamiento en esta zona (Guamán, 2012; Ruiz, Sánchez, & Arteaga, 2016). A su vez, la detección de vehículos mediante visión por computador se está convirtiendo en un tema cada vez más importante dentro de la investigación, ya que puede ser aplicado en campos como el control de tráfico vehicular (Santamaría & Moscol, 2015). Por ello este trabajo tiene por objetivo el desarrollo e implementación de una solución que permita detectar y contabilizar el número de vehículos que circulan por cada calle a través de imágenes aéreas, mediante algoritmos de visión por computador y con ello implementar un sistema de semaforización inteligente en nuestra ciudad para controlar la circulación de vehículos y agilizar el tráfico de las vías públicas.

El siguiente trabajo está dividido en los siguientes capítulos:

En el primer capítulo se presenta el planteamiento del problema, los objetivos que se pretenden alcanzar, la metodología a emplearse, como también la justificación, exponiendo todas las razones que motivaron a la realización del presente trabajo.

El segundo capítulo aborda el estado del arte dónde presenta algunos conceptos generales para un mejor entendimiento y los trabajos relacionados publicados en revistas científicas, documentos técnicos, conferencias, páginas web con información acerca de los algoritmos de visión por computador y las distintas técnicas de procesamiento de imágenes digitales.

El tercer capítulo abarca los materiales y métodos, dónde se realiza un análisis de requerimientos tanto en equipo físico como en programas, herramientas y algoritmos de visión por computador necesarios para el desarrollo del proyecto.

En el cuarto capítulo se explica todos los distintos procedimientos realizados durante el desarrollo y la implementación del trabajo de titulación.

En el quinto capítulo se evalúa el funcionamiento del sistema. Se presentan todos los resultados obtenidos a partir de cada una de las pruebas realizadas para obtener las conclusiones y recomendaciones acerca del presente proyecto de titulación.

Finalmente, se presenta la discusión, donde se expone los trabajos futuros, ventajas y desventajas del presente trabajo y los costos de implementación mediante la evaluación de las posibles soluciones que se puede tener para este trabajo.

CAPÍTULO I.
ANTECEDENTES

1.1. Planteamiento del problema

El excesivo tráfico vehicular que se presenta en horas pico es uno de los mayores inconvenientes en nuestra ciudad (AGN, 2010), y no solo en Loja, sino en varios lugares alrededor del mundo. Con el aumento de la demanda del parque automotor en la ciudad (Ruiz et al., 2016), la necesidad de las personas para movilizarse en el menor tiempo posible de un lugar a otro para cumplir con sus actividades diarias en sus vehículos y el cierre de vías generadas debido a la regeneración urbana que es un problema latente en Loja (AVB, 2012), constituyen factores que producen una gran acumulación vehicular y con ello molestias para los conductores, que buscan conocer una ruta descongestionada para llegar a su destino en el menor tiempo posible.

Es por esto que se desarrolla el presente trabajo de titulación con la finalidad de reemplazar el sistema de semaforización tradicional por un sistema inteligente mediante la ayuda de vehículos aéreos no tripulados y de algoritmos de visión por computador que permitan detectar y contabilizar el número de vehículos por cada calle a partir de las imágenes aéreas captadas, para que los semáforos ubicados en cada intersección comparen el número de vehículos obtenidos por cada calle y con ello controlar el tránsito de manera más eficiente en función de los vehículos contabilizados mediante el control de los tiempos de duración de las fases del semáforo, dando una solución óptima a este problema con el fin de reducir de manera considerable la gran congestión vehicular.

1.2. Objetivos

1.2.1. Objetivo general.

Diseñar e implementar una solución para el conteo de vehículos a partir de imágenes aéreas.

1.2.2. Objetivos específicos.

- Revisar el estado del arte.
- Diseñar la arquitectura funcional.
- Diseñar funcional y operativamente la solución.
- Implementar la solución.
- Evaluar el desempeño de la solución desde una perspectiva técnica y económica.

1.3. Justificación

La finalidad de la presente investigación se enfoca en estudiar los distintos algoritmos de visión por computador existentes en la actualidad, mediante la ayuda de los mismos y de

cámaras montadas en vehículos aéreos no tripulados, solucionar uno de los mayores problemas que nuestra ciudad afronta, como lo es la congestión vehicular.

Se busca entonces, mediante esta solución, reducir los embotellamientos producidos a diario por los vehículos en la parte céntrica de la ciudad a través de un sistema automático de detección y contabilización de vehículos que sea viable desde una perspectiva técnica como económica. Asimismo, en cierta parte, este trabajo busca disminuir la contaminación ambiental que producen los vehículos en las vías, siendo también una alternativa ecológica.

1.4. Alcance y limitaciones

La solución que se pretende desarrollar es una aplicación que por medio de algoritmos detectores y descriptores pueda crear un panorama a partir de las imágenes aéreas captadas del casco urbano de la ciudad de Loja gracias a la ayuda de vehículos aéreos no tripulados.

Se busca procesar la imagen o el mosaico resultante mediante algoritmos de visión por computador en el programa MATLAB, realizando la detección y el conteo vehicular en cada calle origen, para que cada semáforo tome una decisión y con ello permitir el paso a la calle objetivo con el mayor número de vehículos, así como también recomendar a los conductores las calles menos congestionadas para que los mismos puedan llegar a su destino mucho más rápido.

1.5. Metodología

En el presente trabajo de titulación se emplea una metodología que busca realizar avances por fases, y que se describe cada una a continuación:

La primera fase consiste en recolectar y buscar información sobre temas relacionados para poder realizar el presente trabajo de titulación. Se procedió a analizar publicaciones científicas, documentos técnicos, códigos relacionados y libros acerca de procesamiento de imágenes con la ayuda de MATLAB; todo esto con el fin de obtener la mayor cantidad de información relevante para proceder a estudiar todas estas posibilidades y seleccionar la que más se adecúe para realizar los objetivos propuestos.

En la segunda fase consta el desarrollo de la solución mediante el mejoramiento de algoritmos de visión por computador antes ya diseñados en el programa MATLAB, para adecuarlos e implementarlos según la aplicación propuesta en este trabajo de titulación.

La tercera fase corresponde a la evaluación del funcionamiento del sistema mediante pruebas desarrolladas en distintos escenarios, documentando todos los resultados obtenidos en cada uno de ellos, para obtener las conclusiones y las recomendaciones que se pueda dar al finalizar el proyecto de titulación.

La metodología antes descrita se presenta en la Figura 1:

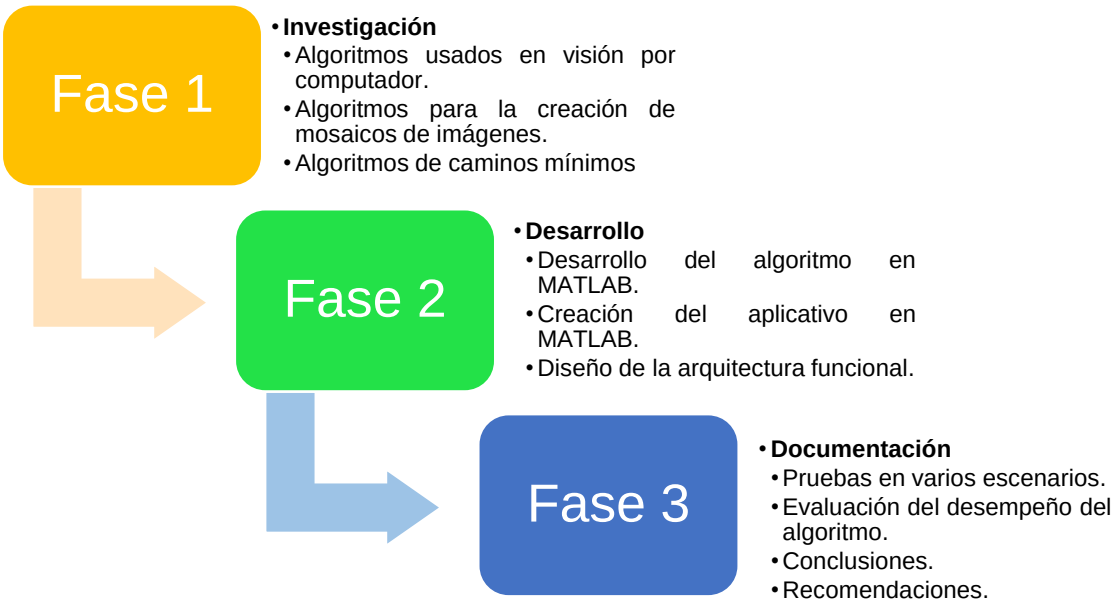


Figura 1. Fases para el desarrollo del proyecto.
Fuente: Imagen propia del autor.
Elaboración: Autor.

CAPÍTULO II.
ESTADO DEL ARTE

2.1. Conceptos generales

Antes de empezar con la realización del presente trabajo, se debe tener claro ciertos conceptos relevantes para un mejor entendimiento del mismo.

2.2.1. Procesamiento de imágenes digitales.

El procesamiento digital de imágenes se refiere al tratamiento de las imágenes que son captadas en el mundo real de una manera digital por medio de un dispositivo. Estudia los fundamentos conceptuales de la adquisición y despliegue de las imágenes, con detalle de los fundamentos teóricos y algorítmicos del procesamiento como tal (García, 2008).

Los procedimientos que se siguen para el procesamiento digital de imágenes generalmente se subdividen en seis áreas que pueden ser: captura y adquisición, pre-procesamiento, segmentación, descripción, reconocimiento e interpretación. Sin embargo, no todas las aplicaciones requieren de todos los procesos. Esto depende de la complejidad del problema a resolver.

2.2.1.1. Imagen digital.

Es una representación bidimensional definida como $f(x,y)$ donde x e y son coordenadas en el plano y la amplitud de la función f se la denomina como la intensidad o nivel de gris en ese punto (González & Woods, 1996).

Una imagen digital está compuesta por un número finito de elementos llamados píxeles; donde el término píxel (del inglés “picture element”), es la unidad mínima de información de una imagen, siendo estas unidades empleadas en cualquier dispositivo de adquisición de imágenes de tipo digital (García, 2008).

Dentro del procesamiento digital de imágenes se maneja básicamente cuatro tipos de imágenes, las cuáles son descritas en breves rasgos a continuación:

- **Imágenes RGB (“Red-Green-Blue”):** este tipo de imágenes emplea tres canales para imprimir el color en la pantalla, utilizando ocho bits por canal, es decir, 24 bits de color por cada píxel y reproduciendo hasta 16.7 millones de colores.
- **Imágenes indexadas:** reducen los colores de la imagen a un máximo de 256 y de esta manera disminuye el tamaño del archivo, teniendo un byte por píxel.
- **Imágenes en escala de grises:** utilizan 256 tonalidades distintas de gris, dónde cada píxel tiene un valor de intensidad de brillo comprendido entre 0 (negro) y 255 (blanco).
- **Imágenes binarias:** este tipo de imágenes poseen una profundidad de color de 1 bit, para lo cual utiliza 0 (negro) y 1 (blanco) para representar los píxeles de una imagen (García, 2008).

2.2.1.2. Adquisición de imágenes.

Este proceso consiste en la obtención de una imagen digital a través de un dispositivo de captura como una cámara digital, cámara de vídeo, escáner, y otro tipo de cámara, que permiten obtener una imagen del mundo físico.

2.2.1.3. Pre-procesamiento.

En la adquisición de imágenes existen factores que pueden alterar la calidad de la imagen, por esta razón es necesario un mejoramiento de la imagen. Para ello, esta etapa incluye técnicas como la reducción del ruido presente en la imagen, el realce del contraste, y el suavizado mediante la utilización de filtros lineales y no lineales (García, 2008).

2.2.1.4. Segmentación.

Consiste en la subdivisión de una imagen en sus partes constituyentes u objetos, con el fin de separar las partes que sean de interés del resto de la imagen. Para la detección de las partes en una imagen se identifican los bordes de una imagen o se segmenta la imagen en regiones, líneas o curvas (Nora, Serna Palomino, Ulises, & Concha, 2009).

Los algoritmos de segmentación por lo general se basan en una de las dos propiedades básicas de los valores del nivel de gris que son: discontinuidad y similitud.

- **Discontinuidad:** consiste en la división de la imagen basándose en los cambios bruscos del nivel de gris.
- **Similitud:** se divide la imagen basándose en la búsqueda de zonas que presentan regularidad en los valores del nivel de gris.

2.2.1.5. Transformaciones morfológicas.

Radica en una serie de transformaciones con el objetivo de facilitar la búsqueda de información y obtención de características de una imagen. Este proceso va desde operaciones aritmético-lógicas y geométricas entre píxeles, hasta filtros para la detección de bordes (González et al., 2006).

Las operaciones morfológicas están basadas en la teoría de conjuntos. Dentro de la morfología matemática, las imágenes de tipo binario son un conjunto de puntos en dos dimensiones que representan los puntos activos en una imagen; mientras que, las imágenes en escala de gris, son conjuntos en tres dimensiones, dónde la tercera componente es el nivel de intensidad de gris (González et al., 2006).

Las distintas operaciones morfológicas que se pueden realizar se detallan a continuación:

- **Dilatación:** esta operación morfológica aumenta el tamaño de un objeto, dependiendo de la elección del elemento estructural. Una de las aplicaciones es la unión de píxeles relacionados.
- **Erosión:** es el proceso inverso a la dilatación, el cuál combina dos conjuntos usando el concepto de inclusión. Esta transformación disminuye el tamaño de los objetos, siendo usado para la eliminación de detalles que son irrelevantes en la imagen.
- **Apertura:** es la operación de erosionar y luego dilatar una imagen. La apertura no es una operación inversa a la erosión.
- **Cierre:** es el proceso de dilatar y luego erosionar una imagen, siendo una operación inversa a la apertura (González et al., 2006).

2.2.1.6. Representación y descripción.

En esta etapa se obtienen características convenientes a partir de la salida del proceso de segmentación, donde se tienen los datos de píxeles en bruto que pueden constituir el contorno de una región o a su vez todos los puntos de una región determinada. En ambos casos es necesario convertir estos datos a una forma adecuada para el procesamiento (Molina, 1998).

El primer caso consiste en la toma de decisiones para saber si lo datos se pueden representar como un contorno o una región completa, teniendo en cuenta esta representación es la más adecuada cuando el interés se establece en las características externas, como esquinas e inflexiones. En cambio, la representación regional (segundo caso) es más adecuada cuando el interés se centra en propiedades internas, como la textura o estructura de un objeto (Molina, 1998).

2.2.1.7. Reconocimiento e interpretación.

Esta es la última etapa en cuanto a procesamiento digital de imágenes se refiere. En este proceso se asigna una etiqueta a cada objeto basándose en la información obtenida en los descriptores (Molina, 1998). En sí, la interpretación significa asignar significado al conjunto de objetos reconocidos.

2.2.2. Algoritmos detectores y descriptores.

Uno de los aspectos más estudiados en la visión por computador es el uso de algoritmos detectores y descriptores que ayudan a la extracción de información importante de archivos multimedia como imágenes. Se puede hacer uso de estos algoritmos en diversas aplicaciones como: fotogrametría, visión en robótica industrial, calibración de cámaras, reconocimiento de objetos o modelado 3D.

Los algoritmos que se han visto convenientes para estudiarlos y seleccionar el más conveniente para la implementación en el presente trabajo son presentados en la sección 2.2.2.1.

2.2.2.1. SIFT: "Scale-invariant Feature Transform".

SIFT es un algoritmo detector y descriptor de características locales en una imagen, siendo invariante a la traslación, rotación, escalamiento y robusto frente a variaciones de iluminación donde cada punto invariante de la imagen posee una serie de valores que describen al punto (Lowe, 2004).

Para la selección de los puntos clave se elimina los puntos con bajo contraste y también los puntos que se encuentran en los bordes, ya que estos pueden crear confusión para las correspondencias siguientes. El tamaño del descriptor es un vector de 128 elementos con información referente a las características invariantes tales como: posición, escala, orientación y otros valores descriptivos.

Este algoritmo se realiza básicamente mediante cuatro pasos que son:

2.2.2.1.1. Detección de extremos en el espacio-escala.

Para empezar, se encuentra una función continua denominada espacio-escala $L(x, y, \sigma)$ mediante la convolución ($*$) entre la función gaussiana $G(x, y, \sigma)$, con valores distintos en la escala σ y la función $I(x, y, \sigma)$ que representa la coordenada de cada píxel en la imagen (Ecuación 1):

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y, \sigma) \tag{1}$$

donde, x e y son las coordenadas en la imagen, $\sigma = \sqrt{2}$ es el factor del espacio-escala y la función gaussiana es representada en la siguiente expresión (Ecuación 2):

$$G(x, y, \sigma) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{2}$$

Seguidamente, para la detección eficiente de la localización de los puntos clave en el espacio-escala, David Lowe (Lowe, 1999) detecta los extremos del espacio-escala en la diferencia de gaussianas (DoG) convolucionada con la imagen $I(x, y, \sigma)$, detectando de manera rápida los puntos característicos más estables mediante la diferencia de dos escalas cercanas separadas por un factor multiplicativo constante k (Ecuación 3).

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma) \tag{3}$$

El proceso anterior se lo puede observar en la Figura 2.

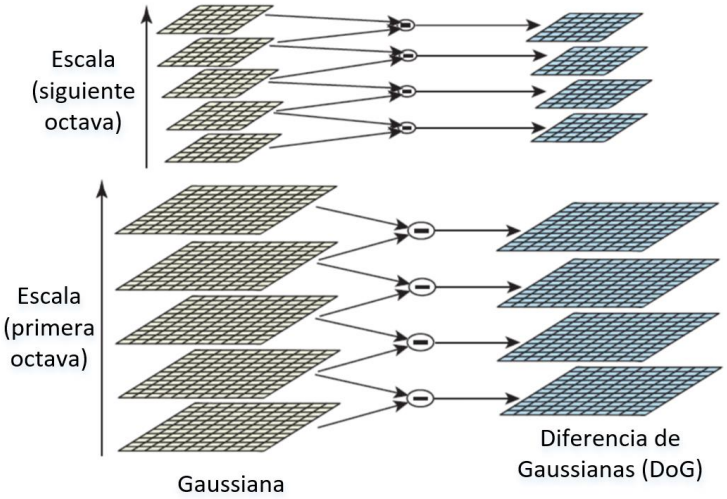


Figura 2. Detección de extremos del espacio-escala
Fuente: Tomado de (Lowe, 2004).
Elaboración: David Lowe (Lowe, 2004).

2.2.2.1.2. Localización de puntos de interés.

La detección de extremos en el espacio-escala produce varios candidatos, entre ellos, puntos con bajo contraste los cuales no son estables a cambios de iluminación y al ruido. Para descartar estos puntos se estima la función DoG en torno a un punto (x_0, y_0, σ_0) mediante una serie de Taylor de segundo grado con el fin de obtener el punto extremo. Si el valor del punto extremo encontrado es menor que determinado umbral, el punto es descartado de la lista de puntos clave.

2.2.2.1.3. Asignación de orientación.

Aquí se calcula la orientación de cada punto característico seleccionado en el paso anterior, basándose en la dirección del gradiente de la imagen. Los siguientes procesos se realizan con los datos de la imagen que ha sido transformada según su orientación, escala y localización asignada para cada característica y así obtener invariancia ante estas transformaciones. Por cada punto de la imagen $L(x, y, \sigma)$ es posible determinar el módulo de su gradiente $m(x, y)$ presentada en la Ecuación 4 y la fase del mismo $\theta(x, y)$ utilizando diferencias entre píxeles como se muestra en la Ecuación 5.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x+1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \quad (4)$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x+1,y)}\right) \quad (5)$$

Realizado todo esto, se obtiene información de la localización, escala y orientaciones principales de cada región importante de la imagen.

2.2.2.1.4. Descriptor de puntos clave.

A cada punto característico se le asigna una escala, una localización en x e y y una orientación. Luego se determina para cada punto un descriptor invariante ante cambios de iluminación, el cual está basado en el entorno del mismo.

Una vez determinada la fase y la magnitud del gradiente en torno al punto clave, se pondera en una ventana gaussiana los valores del módulo y fase, con subregiones de tamaño 4×4 píxeles en la vecindad del mismo y nuevamente se genera para cada subregión un histograma de 8 direcciones distintas. Finalmente se obtiene un descriptor para cada punto clave, de 4×4×8 = 128 valores (Figura 3).

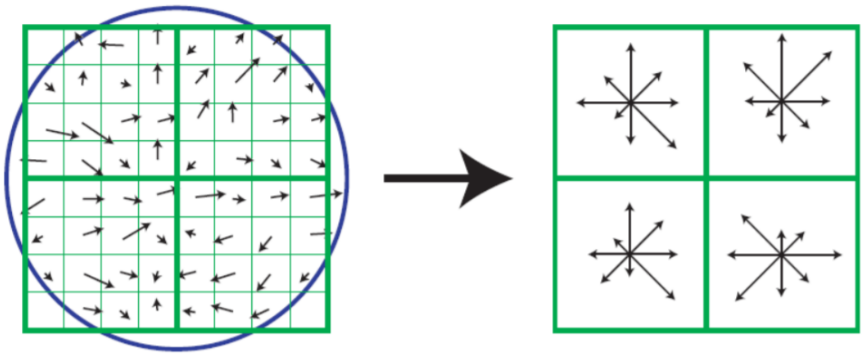


Figura 3. a) Gradiente de la imagen y b) Descriptor del punto clave
Fuente: Tomado de (Lowe, 2004).
Elaboración: Lowe, D.

En consecuencia, el algoritmo determina un número de puntos clave con descriptores altamente distintivos, consiguiendo que cada característica sea correctamente correspondida, haciendo de este un algoritmo altamente robusto (Lowe, 2004).

2.2.2.2. SURF: "Speeded Up Robust Features".

Inspirado en su antecesor SIFT (Lowe, 2004), el algoritmo descriptor y detector SURF diseñado por Bay, Tuytelaars y Van Gool (Bay, Tuytelaars, & Van Gool, 2006), trabaja con el detector de BLOB (Bay et al., 2006) para la transformación de imágenes, ofreciendo una velocidad mejorada para la detección de características y siendo invariante frente a rotaciones, cambios de escala y cambios en la iluminación.

En cuanto a la descripción de características, SURF se ha mejorado para altas tasas de reconocimiento con la ayuda de la Transformada de Haar-wavelet (Stanković & Falkowski, 2003) y con ello proveer una alternativa robusta al descriptor SIFT.

2.2.2.2.1. Detección de puntos de interés.

El descriptor SURF está basado en la determinante de una matriz Hessiana para la localización y la escala de los puntos. Esto resulta muy interesante, ya que no se usa una métrica para la posición y otra para la escala de los puntos de interés, sino que, al contrario, emplea la misma medida en todos los casos.

Donde, dado un punto $X = (x, y)$ en una imagen I, la matriz Hessiana $H(X, \sigma)$ en X y con escala σ, es definida en la Ecuación 6:

$$H(X,\sigma) = \begin{bmatrix} L_{xx}(X,\sigma) & L_{xy}(X,\sigma) \\ L_{xy}(X,\sigma) & L_{yy}(X,\sigma) \end{bmatrix} \quad (6)$$

siendo $L_{xx}(X, \sigma)$ la convolución de segundo orden derivativo de la Gaussiana $\frac{\partial^2}{\partial \chi^2} g(\sigma)$, con la imagen I en el punto x, y, y de la misma manera para $L_{xy}(X, \sigma)$ y $L_{yy}(X, \sigma)$.

Para conseguir eficiencia, se suprimen las segundas derivadas mediante una aproximación de filtros tipo caja, denotando las derivadas de segundo orden como D_{xx}, D_{xy}, D_{yy}.

SURF transforma la imagen original en una imagen integral dónde no es necesario aplicar un filtro tipo gaussiano a la salida de una imagen que ha sido filtrada, sino que utiliza filtros tipo caja de cualquier tamaño sobre la imagen original. El algoritmo SURF a diferencia de SIFT, no disminuye el tamaño de la imagen, sino que eleva el tamaño del filtro de caja.

Por ello, el determinante se calcula mediante la Ecuación 7:

$$\det(H_{aprox.}) = D_{xx}D_{yy} - (0{,}9D_{xy})^2 \quad (7)$$

donde el valor 0,9 tiene relación con la aproximación del filtro Gaussiano.

La representación de la derivada parcial de segundo orden de un filtro gaussiano discretizado y la aproximación de la derivada implementada por el descriptor SURF, se presenta en la Figura 4.

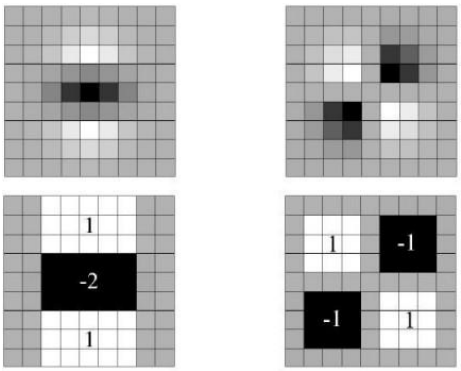


Figura 4. Derivada parcial de segundo orden del filtro gaussiano discretizado y aproximación de la derivada por el descriptor SURF
Fuente: Tomado de (Bay et al., 2006).
Elaboración: Bay, H.

Estos filtros pueden ser evaluados de manera rápida usando la imagen integral, independientemente del tamaño. De esta manera el espacio-escala se analiza escalando el tamaño del filtro en vez de reducir de manera iterativa el tamaño de la imagen, para así reducir considerablemente el tiempo de cálculo.

Posteriormente, para encontrar los puntos de interés en todas las escalas, se aplica una supresión no máxima en un vecindario 3×3×3 píxeles. Luego, se procede a interpolar datos cercanos para hallar la localización en el espacio-escala a la precisión de sub-píxeles. Para esto, el determinante de la función Hessiana se interpola tanto en la escala como en la posición de la imagen, concluyendo esta etapa de detección.

2.2.2.2.2. Asignación de la orientación.

En este punto se fija una orientación consistente a cada uno de los puntos de interés calculándose las respuestas de Haar-wavelet donde se toma un área circular centrada en el punto de interés, cuyo radio es 6σ, donde σ es la escala del punto de interés.

Las respuestas se representan como puntos en el espacio vectorial, dónde la respuesta en x es la proyección del vector a lo largo de la abscisa y las respuestas en y, la proyección a lo largo de la ordenada. La orientación dominante es seleccionada girando un segmento circular que cubre un ángulo de $\pi/3$ alrededor del origen (Figura 5). En cada posición, las respuestas en x e y dentro del segmento se suman y forman un nuevo vector. El vector de mayor longitud es el que da la orientación al punto de interés.

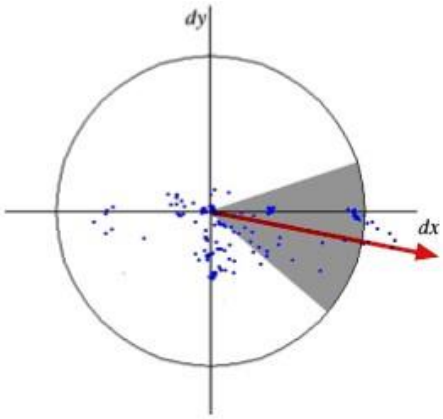


Figura 5. Vector de orientación y sus respuestas obtenidas a partir de las funciones de Haar.
Fuente: Tomado de (Bay et al., 2006).
Elaboración: Herbert, B.

2.2.2.2.3. Extracción de descriptores.

Consiste en la construcción de una región cuadrada de tamaño 20σ alrededor del punto de interés. Esta región se divide en subregiones cuadradas más pequeñas de tamaño 4×4.

Dentro de cada una de estas subregiones, se calcula las respuestas Haar-wavelet de tamaño 2σ para 25 puntos de muestras distribuidos uniformemente (Figura 6).

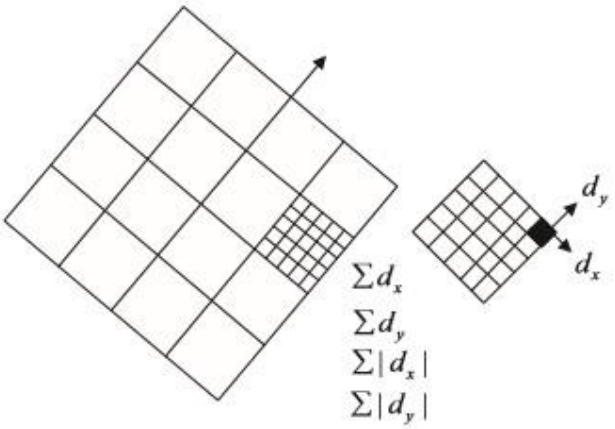


Figura 6. Composición de la característica del descriptor
Fuente: Tomado de (Lei & Wang, 2014).
Elaboración: Lei, F. y Wang, W.

Posteriormente, en cada subregión se suman las respuestas de dx y dy, y también se suman los valores absolutos de las respuestas $|dx|$ y $|dy|$, con el fin de almacenar información de la polaridad sobre los cambios de intensidad. Cada subregión posee como descriptor un vector v de longitud 4, el cuál describe su estructura de intensidad. El descriptor total viene dado por la Ecuación 8:

$$V_{sub-region} = [\sum dx, \sum dy, \sum|dx|, \sum|dy|] \quad (8)$$

Cada subregión aporta cuatro valores al vector descriptor que conlleva a un vector global de longitud 64. El descriptor resultante SURF es invariante a la rotación, la escala, brillo y contraste.

2.2.2.3. ORB: "Oriented FAST and Rotated BRIEF".

Este algoritmo fue desarrollado en OpenCV, presentado y publicado por Rubble, Rabaud, Konolige y Bradski (Rublee, Rabaud, Konolige, & Bradski, 2011), siendo una alternativa a los descriptores SIFT y SURF. Este es un descriptor de características binarias, muy rápido, invariante a la rotación, orientación y resistente al ruido, demostrando buenas capacidades en aplicaciones como lo son la detección de objetos y seguimiento en dispositivos móviles.

Está construido a partir de los detectores FAST (Rosten & Drummond, 2006) y basado en el descriptor BRIEF (Calonder, Lepetit, Strecha, & Fua, 2010), obteniendo un buen rendimiento a bajo costo computacional. Este algoritmo añade una componente de orientación precisa al detector FAST y un cómputo más eficiente del detector BRIEF, siendo robusto al ruido gaussiano que presente la imagen. Sin embargo, carece de un operador de orientación e invariancia a la rotación. Para poder solventar estos problemas, el algoritmo ORB propone:

2.2.2.3.1. oFAST: "FAST Keypoint Orientation".

Las características del algoritmo FAST son utilizadas en varios campos debido a que mejora notablemente la velocidad de cómputo, sin embargo, carece de una componente de orientación.

En la detección de los puntos FAST en la imagen se toma el umbral de intensidad entre el umbral de intensidad del píxel dentro de un área circular alrededor del centro. Para realizar una evaluación rápida de esquinas, generalmente se agrupa 16 píxeles, pero en el caso de ORB se toman 9, que indican la cantidad de ángulos que se están evaluando para la evaluación del píxel candidato y considerarlo como punto de interés. FAST no provee un espacio-escala, pero para ello crea una pirámide de imágenes y obtiene los puntos de interés en cada nivel de la pirámide, obteniendo así una invariancia a la escala.

Para la orientación de la esquina se utiliza una medida de orientación, que es el centroide de intensidad, el cual asume que la intensidad de una esquina está desplazada con respecto a su centro, y este vector puede usarse para computar una orientación mediante el uso del método de momentos geométricos cartesianos de Hu (Ming-Kuei Hu, 1962) presentado en la Ecuación 9:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{9}$$

Siendo m_{pq} el momento de la región del objeto, p y q el orden del momento, x e y son las coordenadas de los píxeles que están dentro de la región circular, mientras que $I(x,y)$ es la intensidad del píxel ubicado en las posiciones (x,y).

Una vez obtenidos los momentos se puede encontrar el centroide (Ecuación 10):

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}}\right) \tag{10}$$

Donde, $m_{10}, m_{00}, m_{01}, m_{00}$ son los cuatro cuadrantes del centroide que representan el centro de gravedad de una región, y la orientación θ del punto de interés, está dada por el ángulo del vector desde el centro de la esquina O al centroide C y se puede calcular con la Ecuación 11:

$$\theta = arctan2(m_{01}, m_{10}) \tag{11}$$

Siendo $arctan2$ la función arcotangente definida para los cuatro cuadrantes.

2.2.2.3.2. rBRIEB: "Rotation-Aware Brief".

Para la extracción de descriptores se hace uso de BRIEF y para corregir los problemas en cuanto a la rotación se genera una matriz que cubre los píxeles tomados para el punto clave y la rota para conocer la versión rotada de la característica. BRIEF maneja una varianza para

la definición de características y al momento de usar la correlación de orientación esto se pierde, volviendo a los descriptores más selectivos.

Este descriptor está constituido por una cadena de bits, que es el resultado de un conjunto de pruebas binarias calculadas con la imagen integral. Sea I una imagen y sea P una parte en I de tamaño $S \times S$ alrededor del punto de interés, se define una prueba binaria τ en P en la Ecuación 12:

$$\tau(P; x, y) \coloneqq \begin{cases} 1 & , \quad p(x) < p(y) \\ 0 & , \quad p(x) \geq p(y) \end{cases} \tag{12}$$

donde, x e y son las ubicaciones de los píxeles en P, $p(x)$ y $p(y)$ son los valores de los píxeles en las coordenadas x e y en una versión suavizada de P.

Un conjunto de n pares de ubicaciones (x, y) definen un conjunto de pruebas binarias. El descriptor BRIEF se genera como una secuencia binaria n-dimensional y se calcula con la Ecuación 13:

$$f_n(P) = \sum_{1 \leq i \leq n} 2^{i-1} \tau(p: x_i, y_i) \tag{13}$$

donde, los valores posibles de n son 128, 256 y 512 que son el tamaño de la cadena de bits en la localización x_i e y_i.

2.2.2.4. ASIFT.

Este algoritmo fue propuesto por Morel y Yu (Morel & Yu, 2009) y a diferencia del algoritmo SIFT, simula tres parámetros: acercamiento, ángulo de la cámara en latitud y longitud, y normaliza los otros parámetros: traslación en x e y, rotación y escala.

ASIFT complementa el algoritmo SIFT mediante la simulación de los parámetros que representan cambios en la orientación de la cámara cuando la imagen es captada. De esta manera, se simulan todas las posibles orientaciones de la cámara sin aumentar considerablemente la carga computacional y se obtiene un método invariante ante transformaciones afines (Morel & Yu, 2009).

2.2.2.4.1. Afinación de la aproximación local.

Cualquier deformación planar suave puede ser aproximada alrededor de cada punto por un mapa afín $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$, debido a la tangencialidad local de las deformaciones de perspectiva a los mapas afines y mediante una aproximación de Taylor (Morel & Yu, 2009).

Cada mapa afín con determinante estrictamente positiva que no sea una función semejante tiene una descomposición afín tal como se muestra en la Ecuación 14:

$$A = H_\lambda R_1(\psi) T_t R_2(\Phi) = \lambda \begin{bmatrix} cos(\psi) & -sin(\psi) \\ sin(\psi) & cos(\psi) \end{bmatrix} \begin{bmatrix} cos(\Phi) & -sin(\Phi) \\ sin(\Phi) & cos(\Phi) \end{bmatrix} \quad (14)$$

donde, $\lambda > 0$, λ es el determinante de A, R_i son rotaciones, $\Phi, \psi \in [0, \pi)$ y T_t es una inclinación nombrada matriz diagonal con un primer valor propio $t > 1$ y un segundo valor propio, llamado "*tilt"* que está representado por t que mide la inclinación entre la vista frontal de la imagen y una vista inclinada que es igual a "1".

2.2.2.4.2. Descripción.

Todas las vistas posibles o distorsiones de la imagen son simuladas en los parámetros que modelan la orientación del eje óptico de la cámara. Los ángulos φ y θ representan los ejes ópticos de la cámara como son la longitud y latitud, respectivamente, y existe un tercer ángulo ψ que es el valor que parametriza el giro de la cámara (Figura 8). A estas imágenes simuladas se aplica SIFT para hallar los puntos característicos, y de esta manera se cubren los seis parámetros que presenta el modelo ASIFT. El esquema gráfico del funcionamiento de ASIFT se puede observar en la Figura 7.

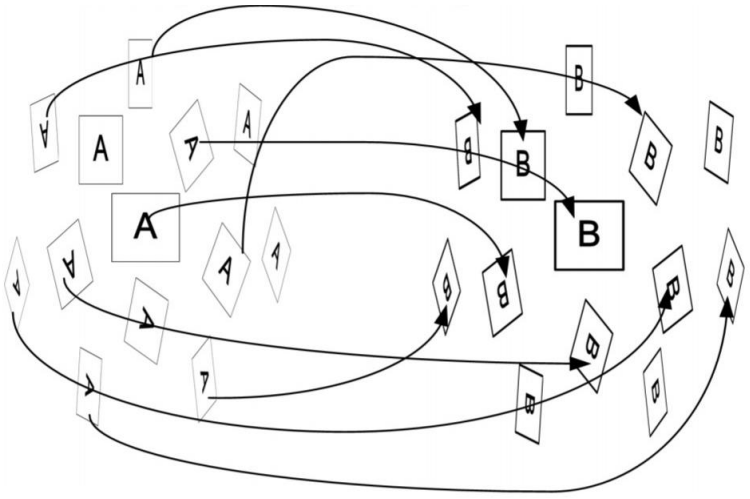


Figura 7. Esquema de funcionamiento ASIFT.
Fuente: Tomado de (Morel & Yu, 2009).
Elaboración: Morel, J. y Yu, G.

Por tanto, si se elige de manera adecuada el muestreo espacial de parámetros necesario para la simulación, el cálculo computacional no resulta más complejo que el cálculo de los descriptores SIFT.

En la Figura 8 se puede observar el muestreo de los parámetros θ y φ denotados con puntos negros. En la parte izquierda de la imagen se observa la ilustración en perspectiva del hemisferio de observación y en la parte derecha una vista cenital del hemisferio de observación.

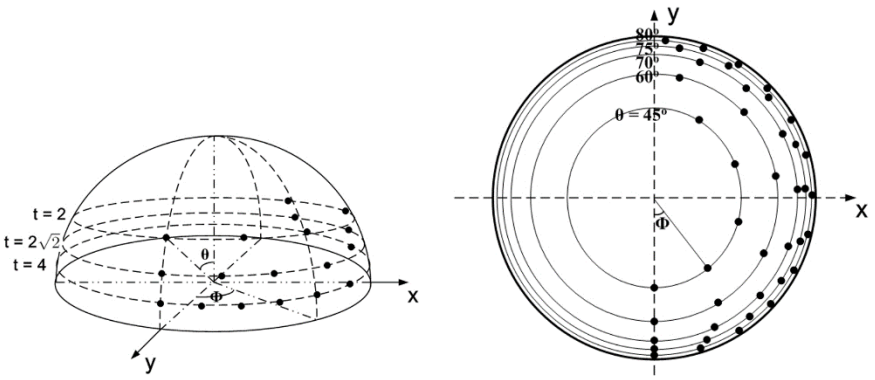


Figura 8. Muestreo espacial óptimo para ASIFT.
Fuente: Tomado de (Morel & Yu, 2009).
Elaboración: Morel, J. y Yu, G.

2.2.2.4.3. Correspondencia.

Al momento de encontrar las correspondencias es importante definir una métrica que permita establecer la similitud entre dos puntos característicos. Para ello es de gran importancia definir un criterio que permita afirmar que dos características deberían ser asociadas.

La métrica utilizada por ASIFT realiza una comparación eficiente entre características diferentes, siendo robusta en la detección de pequeñas variaciones en las características. Para ello este algoritmo trabaja en base a la distancia de transporte (Earth Mover Distance), presentando un mejor comportamiento que otras distancias clásicas bin a bin (distancia Euclidiana, Manhattan o χ^2, por ejemplo) en cuanto a la comparación de características locales.

2.2.2.5. Detector de MSER.

Este algoritmo de detección de regiones MSER (regiones que sin importar el nivel de brillo que posea el entorno de la imagen, se vuelven estables a través de una función), propuesto por Matas (Matas, Chum, Urban, & Pajdla, 2004), trabaja con imágenes en formato a escala de grises, siguiendo un proceso semejante a un algoritmo de componentes conexas.

El detector de MSER detecta regiones con una serie de propiedades de robustez e invariancia que permiten la aplicación de técnicas de reconocimiento y seguimiento de objetos o de búsqueda sobre grandes bases de datos de manera eficiente.

La aplicación de este algoritmo junto a una serie de umbralizaciones de estabilidad desde los mínimos de la imagen hasta los máximos y viceversa, permite obtener las regiones que presentan propiedades de invariancia afín, detección multi-escala y estabilidad.

2.2.3. *Homografía.*

Es una transformación que determina la correspondencia que existe entre dos imágenes bidimensionales modificando la geometría de la imagen, es decir, la transformación de las coordenadas espaciales, con la particularidad de que las rectas de una imagen permanecen rectas luego de su transformación.

La homografía está formada por una matriz de rotación y traslación, siendo una matriz cuadrada de 3×3 que trabaja con puntos homogéneos, manteniendo la imagen invariante en: área, distancia entre puntos y, ángulos entre dos rectas.

2.2.4. *Modelos de estimación de parámetros.*

2.2.4.1. RANSAC.

Es un modelo de estimación de parámetros de un modelo matemático de un conjunto de datos cercanos ("inliers") y que elimina las dificultades provocadas por aquellos datos que se encuentran distantes del resto ("outliers") para poder encontrar la matriz de transformación (Fischler & Bolles, 1981).

Este proceso se lo puede resumir en ocho pasos: en el primero se escoge una muestra de datos de manera aleatoria, y en el segundo se estima los parámetros de la muestra tomada. Si el modelo es una línea, $ax + bx + c = 0$, o también en su forma matricial, $M = [a, b, c]^T$, M es el parámetro a ser estimado con la hipótesis de ser verdadero. Si todos estos datos son "inliers", la hipótesis se encontrará más cercana a ser verdadera, necesitando de cierto número de iteraciones para su recolección. Si esto falla, marcará un error de estimación el cual dependerá de un umbral establecido por el usuario, culminando con el paso tres. En el cuarto paso se realiza la selección de los candidatos "inliers" en base al error. El algoritmo escoge la hipótesis con mayor probabilidad en base a los candidatos a "inliers" considerados como verdaderos, siendo éste el quinto paso. En el sexto, se mantiene el parámetro hasta el final de la estimación. Como penúltimo paso, se optimiza el resultado y se entrega la estimación final como último paso (Choi, Kim, & Yu, 1997).

2.2.4.2. MSAC.

Uno de los inconvenientes que presenta RANSAC es encontrar un umbral adecuado t. Para mejorar esta situación y beneficiar ajustes con puntos cercanos al modelo, se propone un simple estimador llamado MSAC (Torr & Zisserman, 2000) el cual penaliza los puntos "inliers" según el error involucrado, es decir, los "inliers" que posean un error grande tienen mayor penalización que aquellos con un error pequeño mediante el uso de la Ecuación 15:

$$C = \sum_{i=1}^{n} \rho_2(e_i^2) \tag{15}$$

donde, n es el número de muestras para ajustar el modelo MSAC y el término $\rho_2(e_i^2)$ es la probabilidad de error a minimizar. Los errores son minimizados con un valor fijo, mientras que las muestras válidas son puntuadas en función de cómo se ajustan los datos y viene dado por la Ecuación 16:

$$\rho_2(e_i^2) = \begin{cases} e^2, & e^2 < t^2 \\ constante, & e^2 \geq t^2 \end{cases} \tag{16}$$

Donde, i es el número de iteraciones para reducir el tiempo de estimación, e_i^2 son los residuos de las n muestras utilizadas y t^2 es el umbral de corte.

2.2.4.3. MLESAC.

Es una adaptación de RANSAC desarrollado por Torr y Zisserman (Torr & Zisserman, 2000), la cual sustituye la hipótesis de los errores de una sola distribución por otra en dónde los errores resultan de una mezcla. Estos datos pueden ser "inliers" u "outliers", donde los "inliers" siguen una distribución normal y los "outliers" siguen una distribución uniforme (dentro de una ventana de tamaño v). La distribución resultante se la puede expresar como en la Ecuación 17:

$$f = \gamma \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{e^2}{2\sigma^2}\right) + (1+\gamma)\frac{1}{v} \tag{17}$$

siendo γ la proporción de "inliers" y σ que representa la desviación estándar del ruido gaussiano. Esta técnica aumenta la probabilidad de que los puntos sean "inliers". MLESAC ha demostrado ser superior a RANSAC en ciertas tareas de estimación (Torr & Zisserman, 2000).

2.2.4.4. PROSAC.

Este método a diferencia de RANSAC, propone guiar el procedimiento de muestreo si se tiene un conocimiento a priori de la información, es decir, ya no toma de manera aleatoria la muestra inicial de cada iteración, sino que prioriza el orden de selección de estos datos en base a un subconjunto de datos con mayor calidad. PROSAC ha demostrado ser un método robusto y significativamente más rápido que RANSAC (Chum & Matas, 2005).

2.2.4.5. MINPRAN.

Este método utiliza un enfoque contrario a MLESAC. En lugar de aumentar la probabilidad de que un conjunto de puntos sean "inliers", MINPRAN pretende reducir la probabilidad de que los puntos sean "outliers" y utiliza en el nivel básico el mismo mecanismo de RANSAC para agrupar los puntos "inliers" y "outliers". El conjunto de "inliers" son aquellos que están bajo un umbral t del modelo hipotético. Mientras que para los "outliers" se utiliza una distribución

binomial en lugar de una distribución uniforme como sucede en MLESAC. El modelo MINPRAN posee como ventaja la obtención de buenos resultados a pesar de que se desconozca la distribución de “inliers” (Stewart, 1995).

2.2.5. *Semáforos inteligentes.*

Un semáforo inteligente es aquel dispositivo que a diferencia de los semáforos tradicionales es capaz de tomar decisiones en función del tráfico dado. El comportamiento de este tipo de semáforos se da en forma dinámica y se ajusta según varios parámetros como el flujo vehicular, velocidad media de los vehículos, tipos de vehículos o incluso el límite de velocidad de las vías (Martínez, 2014).

Entre las ventajas que dispone el uso de este tipo de semaforización se encuentran:

- Controlar la gran congestión vehicular.
- Evitar tiempos excesivos de viaje.
- Reducir el problema de esperas innecesarias en una intersección.
- Aminorar el gasto de combustible por parte de los vehículos.
- Disminuir la contaminación ambiental.

2.2.5.1. Tecnologías.

Existen varias formas de implementar un sistema de semaforización inteligente comprendiendo desde una sencilla decisión del estado (luz de color verde o luz de color roja) según el tráfico, hasta decisiones más complejas, como tomar una decisión a causa de un accidente, cierre de vías e incluso alterar el tráfico vehicular.

La decisión que puede tomar el semáforo depende en gran manera de acuerdo al tipo de tecnología que utiliza. Entre las tecnologías más utilizadas están las cuatro presentadas en la sección 2.2.5.1.1.

2.2.5.1.1. Semáforo inteligente con RFID.

Para un control inteligente del tráfico se puede variar los tiempos de espera en cada intersección según la afluencia vehicular. Este sistema consta de las siguientes partes principales que son: la tarjeta RFID, punto de acceso, servidor de ubicación, WAN y una base de datos centralizada, la cual almacena los datos y a partir de ellos se toma una decisión. El esquema de esta tecnología se presenta en la Figura 9.

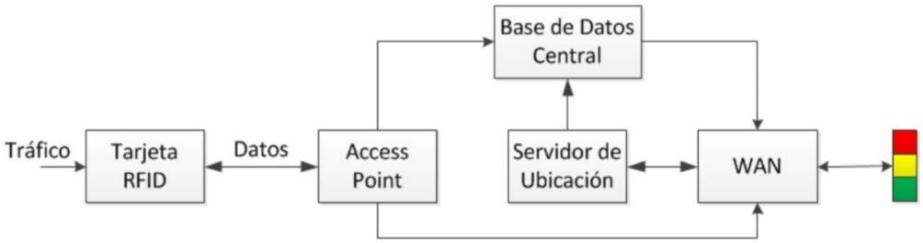


Figura 9. Esquema de un semáforo inteligente con tecnología RFID.
Fuente: Tomado de (Martínez, 2014)**.**
Elaboración: Martínez, M.

Para tomar una decisión en cuanto a la regulación de la congestión vehicular se depende de la información en tiempo real facilitado por el sistema RFID. Mediante la utilización de sensores se consigue la ubicación y tiempos de recorrido de cada vehículo, almacenándose como una etiqueta que puede ser detectada mediante un identificador (Shi, Xu, & Dai, 2013).

La decisión que el algoritmo toma es enviada por medio de Internet hacia el semáforo. Los sensores que están ubicados en distintos sectores obtienen los tiempos y los tipos de vehículos.

Si se produce un accidente, es posible seguir controlando el tráfico, ya que la información es almacenada en la base de datos y utilizada posteriormente. Es aquí donde el sistema puede tomar decisiones y producir un escenario general de flujo de tráfico a través de la identificación de un gran número de situaciones como:

- Identificación de la ruta más activa.
- Presentación de la ruta de salida más eficaz.
- Determinación del patrón de circulación de vehículos en un determinado día.

La desventaja que se presenta en este tipo de tecnología, es que cada vehículo debe poseer su propia etiqueta RFID.

2.2.5.1.2. Semáforo inteligente con redes de sensores inalámbricos.

Este tipo de sistema maneja básicamente la misma idea de los semáforos que utilizan la tecnología RFID. Este sistema cuenta con dos partes principales: la red de sensores inalámbricos (WSN) y la estación base (BS) que se encarga de ejecutar los algoritmos de control.

Consiste en un grupo de sensores diseñados para facilitar la comunicación y el flujo de tránsito. Cada sensor tiene la tarea de generar los datos del tráfico, el número de vehículos, procesos de salida, velocidad de cada vehículo como también su longitud. Estos datos recolectados se envían a la estación base en tiempo real.

Para la comunicación se hace uso de Acceso Múltiple por División de Tiempo (TDMA), debido a que gestiona eficientemente la energía, permitiendo a los nodos conectados a la red entrar en un estado inactivo hasta que sus estados de tiempo sean asignados (Yousef, Al-Karaki, & Shatnawi, 2010).

En el esquema presentado en la Figura 10, se observan los nodos sensores de tráfico (TSN) que son los responsables de la detección y contabilización de los vehículos, como también de la transmisión de esta información a la estación base. El sistema de detección de vehículos se lo realiza con la ayuda de cuatro componentes: un sensor para la detección de señales que son generadas por los vehículos, un procesador para los datos detectados, una unidad de comunicación para el envío de estos datos a la estación base y la fuente de energía (Yousef et al., 2010).

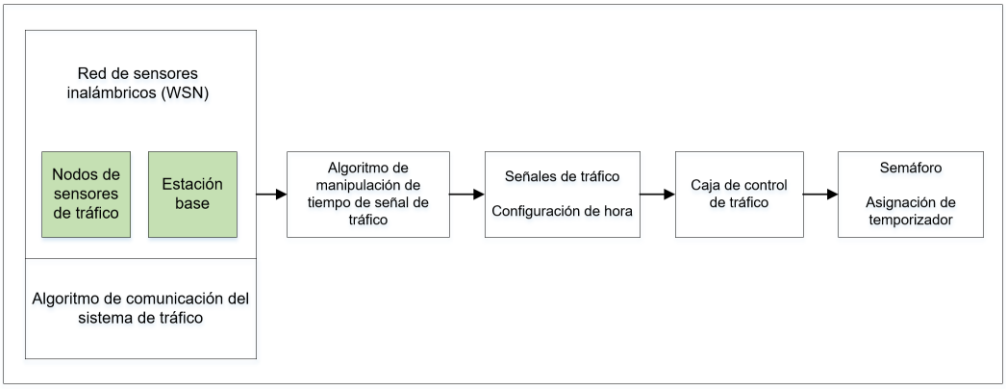


Figura 10. Esquema de un semáforo inteligente con tecnología WSN.
Fuente: Tomado de (Yousef, Al-Karaki, & Shatnawi, 2010).
Elaboración: Autor.

El algoritmo de comunicación del sistema de tráfico (TSA) está diseñado para controlar las rutas de comunicación entre todos los TSN y la estación base, como también con la caja de control de tráfico. Estos datos recolectados en la estación base, se pasan al algoritmo de manipulación de tiempo de señal de tráfico (TSTMA), el cual se encarga de establecer duraciones de tiempo de señales de una manera dinámica en función de los parámetros recibidos.

2.2.5.1.3. Semáforo inteligente mediante procesamiento de imágenes.

Con la ayuda del procesamiento de imágenes es posible medir la densidad del tráfico y mediante los datos obtenidos tomar una decisión. Este método posee las siguientes etapas:

- **Adquisición de imágenes:** las imágenes son captadas mediante una cámara digital. Se toma una imagen con la calle llena de autos y otra sin autos que sirve de referencia fuera de horas pico.
- **Conversión de imágenes:** las imágenes captadas en formato RGB se las convierte en un formato en escala de grises.

- **Mejora de la imagen:** la imagen en muchas situaciones es afectada por varios factores, como el ruido e iluminación. Para ello se mejora la calidad de la imagen mediante un pre-procesamiento.
- **Procesamiento de la imagen obtenida:** se resta la imagen sin vehículos de la imagen original, obteniendo como resultado solamente los vehículos. A la imagen resultante de esta operación se la procesa mediante algoritmos de detección de bordes y transformaciones morfológicas con el fin de determinar el número de vehículos.

Una de las ventajas que puede obtenerse mediante este sistema es que puede usarse en monitoreo de calles y también en sistemas de seguridad y control (Chávez, 2015).

2.2.5.1.4. Semáforo inteligente basado en inteligencia artificial.

Este tipo de sistema se divide en sistemas de refuerzo de aprendizaje y lógica difusa. En el caso de refuerzo de aprendizaje se realizan estimaciones que minimicen el tiempo de espera de los vehículos. El controlador del semáforo debe ser capaz de estimar cuanto tardaría un automóvil en llegar a su destino (pasando por varios semáforos) cuando la luz estaba en verde como cuando la luz se puso en rojo. Este tipo de sistema calcula los promedios a largo plazo en los tiempos de espera mediante la utilización de algoritmos de programación dinámica.

En el segundo caso se utiliza la técnica de lógica difusa. Para ello se determina la presencia o ausencia de vehículos permitiendo el paso a la calle donde existe mayor cantidad de automóviles y si existen carriles vacíos, buscar un carril con vehículos y colocar dicho carril en color verde en el mapa resultante final para permitir el flujo.

2.2.6. *Algoritmos de búsqueda de la ruta más corta.*

Al momento de trabajar con grafos dirigidos etiquetados o ponderados con pesos no negativos, es muy utilizado encontrar el camino más corto entre dos vértices del grafo, es decir, permitir llegar desde un vértice hacia otro recorriendo el grafo a través de las aristas y buscando la menor distancia entre los mismos.

2.2.6.1. Algoritmo de Dijkstra.

Este algoritmo realizado por Edsger Dijkstra (Dijkstra, 1959), se aplica sobre un grafo ponderado, calcula la distancia más corta desde un vértice inicial hacia el resto de vértices del grafo y explora los caminos más cortos entre los vértices. Este método no funciona en grafos con aristas de coste negativo ya que al elegir el nodo con menor distancia, pueden excluirse de la búsqueda nodos que en próximas iteraciones bajarían el costo general del camino al pasar por una arista con costo negativo.

El algoritmo utiliza la matriz de adyacencia etiquetada o matriz de costos, que es la misma matriz de adyacencia, pero con la diferencia de que en lugar de colocar un número "1" cuando dos vértices son adyacentes, se coloca el peso o la ponderación asignado a la arista que los une.

Este algoritmo se basa en el siguiente principio: si el camino más corto entre los vértices u y v pasa por el vértice w, entonces la parte del camino que va desde w a v debe ser el camino más corto entre todos los caminos que van desde w a v. De esta manera se construyen sucesivamente los caminos de costo mínimo que parten desde el nodo de origen hasta cada vértice del grafo.

2.2.6.2. Algoritmo BFS: "Breadth-First Search".

Es un algoritmo de búsqueda sin información que expande y examina todos los nodos de un árbol de manera ordenada para buscar una solución sin utilizar ninguna estrategia heurística.

La idea básica de este algoritmo es recorrer y buscar elementos en un grafo formando un árbol a medida que recorre el grafo. El algoritmo elige como inicio un nodo raíz y explora todos los vecinos del nodo. Para cada uno de los vecinos se exploran sus respectivos vecinos adyacentes y así hasta recorrer todo el árbol.

El algoritmo BFS encuentra la ruta más corta entre dos vértices cuando el peso de todas las aristas entre todos los nodos es "1", también es utilizado cuando se requiere llegar con un movimiento de caballo de un punto a otro con el menor número de pasos o para salir de un laberinto con el menor número de pasos (Cormen, Leirserson, & Rivest, 2001).

2.2.6.3. Algoritmo Bellman-Ford.

Este método determina la ruta más corta desde el nodo origen que es la raíz del árbol hacia los demás nodos del grafo. Para ello se requiere como entrada un grafo cuyas aristas posean peso, pero con la diferencia de que este algoritmo acepta valores negativos ya que permite la detección de la existencia de un ciclo negativo, siendo un modo más general que el algoritmo de Dijkstra.

El algoritmo devuelve un valor booleano si encuentra una arista con peso negativo, caso contrario calcula y devuelve el camino con el menor costo. Para cada vértice v perteneciente a V, se mantiene un atributo $d(v)$ como costo del camino mínimo desde el nodo origen al vértice v (Bellman, 1954; Ford & Fulkerson, 1962).

2.2.6.4. Algoritmo DFS: "Depth-First Search".

Es un método de exploración de grafos ya sean orientados o no. Parte desde un vértice denominado *fuente* y explora recursivamente sus vértices sucesores. Busca en profundidad tanto como sea posible y cada arista es explorada desde el último vértice descubierto v. Cuando todas las aristas desde v han sido exploradas, la búsqueda retrocede, denominándose "backtracks" y de esta manera se exploran todas las aristas.

El proceso continúa hasta que todos los vértices alcanzables desde el vértice fuente hayan sido descubiertos. Si aún existen vértices sin descubrir, uno de ellos es seleccionado como un nuevo nodo vértice fuente y se repite la búsqueda (Cormen et al., 2001).

2.3. Trabajos relacionados

El presente trabajo tiene como finalidad encontrar una solución para disminuir el tráfico en el casco urbano de la ciudad de Loja mediante el procesamiento de imágenes aéreas. Para ello en esta sección se ha recurrido a la búsqueda de trabajos en revistas científicas, documentos técnicos y conferencias para observar como tratan de solucionar estos problemas.

2.3.1. *Técnicas para la creación de vistas panorámicas tipo mosaico.*

La construcción automática de mosaicos de imágenes en alta resolución se utiliza en diversas aplicaciones como la creación de imágenes panorámicas y satelitales. Para captar imágenes aéreas, se trabaja con vehículos aéreos no tripulados y cámaras digitales incorporadas en los mismos, ya que poseen grandes ventajas como la flexibilidad y seguridad, pudiendo ser utilizados en la obtención de un panorama de un área en un corto tiempo.

Sin embargo, al momento de combinar dos o más imágenes, se puede tener problemas tales como una imagen de mala calidad, un gran ángulo de rotación y pequeñas distorsiones que se puedan presentar en la imagen resultante, dificultando la creación de mosaicos con métodos convencionales. Para ello, existen varias técnicas que tratan de disminuir estos problemas lo más posible y se presentan en la sección 2.3.1.1.

2.3.1.1. Creación de vistas panorámicas utilizando SIFT.

El uso de técnicas basadas en SIFT es ampliamente utilizado para la construcción automática de mosaicos de imágenes en alta resolución, debido a sus propiedades de invariancia ante la rotación y la escala, pudiendo ser usados en distintas aplicaciones, entre ellas la construcción de vistas satelitales a partir de un conjunto de imágenes.

Una vez obtenidos los puntos clave se procede a extraer los puntos característicos por medio del cálculo de la distancia euclidiana de los descriptores, pero pueden existir puntos erróneos detectados por SIFT. Para ello Yang y Guo (Yang & Guo, 2008) proponen el uso del algoritmo

RANSAC, el cual es un modelo de estimación encargado de eliminar estos falsos puntos característicos con el fin de obtener una mejor correspondencia y con ello finalmente obtener el mosaico (Figura 11). Los resultados obtenidos al aplicar este modelo presentan un algoritmo robusto ante la traslación, rotación, ruido y escalamiento (Tabla 1).

a) b)

Figura 11. a) Características para la correspondencia luego de aplicar RANSAC y b) Mosaico de imágenes.
Fuente: Tomado de (Yang & Guo, 2008).
Elaboración: Yang, Z. y Guo, B.

Tabla 1. Resultados de la creación del mosaico de imágenes aplicando RANSAC.

Par de imágenes	Correcto		Calculado	
	Escala	Ángulo de rotación (θ)	Escala	Ángulo de rotación (θ)
a	1,0	45,0°	1,0	44,9°
b	0,1	0,0°	0,1	0,0°
c	0,6	15,0°	0,6	14,9°
d	4,0	125,0°	4,0	124,9°

Fuente: Tomado de (Yang & Guo, 2008).
Elaboración: Autor.

Cuando se tienen muchos desajustes se dificulta el encontrar la matriz de transformación. Para ello el artículo de Shi (Shi et al., 2013), propone la mejora del algoritmo RANSAC, removiendo los puntos característicos erróneos detectados por el algoritmo SIFT (Tabla 2) mediante dos métodos: el primero eliminando puntos característicos que no pertenecen al área deseada y segundo, la eliminación de los puntos de cruce.

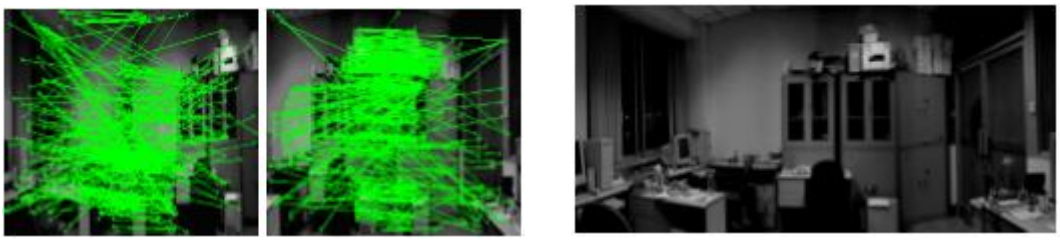
Tabla 2. Número de puntos característicos.

Umbral	0,7	0,8
Correspondencias	24	39
Correspondencias con el primer método	16	24
Correspondencias con el segundo método	13	20
Correspondencias luego de encontrar el modelo	8	12

Fuente: Tomado de (Shi et al., 2013).
Elaboración: Autor.

Otro artículo que propone SIFT en la creación de mosaicos es el de Gao, Li y Chen (Gao, Li, & Chen, 2011) que adopta dicho descriptor para extraer los puntos característicos de cada imagen (Figura 12) pero con la diferencia que el valor óptimo de la matriz de transformación es calculado por PSO (Kennedy, 2011), el cual se encarga de la optimización del cómputo para la creación del mosaico de imágenes (Figura 13), demostrando validez y viabilidad a través del método propuesto tal como se presenta en la Tabla 3.

Figura 12. Secuencia de imágenes para formar el mosaico.
Fuente: Tomado de (Gao et al., 2011).
Elaboración: Gao, H., Li, D y Chen, F.



a) b)

Figura 13. a) Correspondencias con SIFT. b) Mosaico resultante.
Fuente: Tomado de (Gao et al., 2011).
Elaboración: Gao, H., Li, D y Chen, F.

Tabla 3. Resultados de la creación del mosaico basado en SIFT.

Imágenes de prueba	Resolución	Porcentaje de puntos característicos	Tiempo (s)
Interior	800x600	92,8 %	10,2
Exterior	226x344	95,5 %	8,3

Fuente: Tomado de (Gao et al., 2011)
Elaboración: Autor.

Una de las desventajas que se tiene al utilizar vehículos aéreos no tripulados es la inestabilidad de la nave donde en muchas situaciones suelen captarse imágenes de mala calidad con un gran ángulo de rotación, dificultando la creación de mosaicos de imágenes a través de métodos convencionales. Es por esto que el trabajo presentado por Sun y Zeng (S. Sun & Zeng, 2013), plantea reducir estos problemas al momento de la creación del mosaico (Figura 14) a través de una mejora del método SIFT, haciéndolo más robusto mediante la creación de una pirámide de imágenes para reducir el número de puntos característicos SIFT sin necesidad de aumentar el cálculo y obteniendo mejores resultados (Tabla 4).

Figura 14. Mosaico con el método SIFT mejorado.
Fuente: Tomado de (S. Sun & Zeng, 2013).
Elaboración: Sun, S y Zeng, Z.

Tabla 4. Resultados del algoritmo SIFT mejorado y el algoritmo SIFT tradicional.

Par de imágenes	Algoritmo SIFT mejorado				Algoritmo SIFT tradicional			
	Umbral de relación de distancia	Puntos característicos	Puntos correctos	Precisión	Umbral de relación de distancia	Puntos característicos	Puntos correctos	Precisión
1	0,6	111	108	97,3 %	0,8	189	119	63,0 %
2	0,5	132	124	93,9 %	0,8	254	163	64,2 %
3	0,4	192	187	97,4 %	0,8	349	236	67,6 %
4	0,5	213	201	94,4 %	0,8	341	238	69,8 %
5	0,6	172	172	91,5 %	0,8	298	193	64,8 %

Fuente: Tomado de (Hong, Lin, Zhang, & Li, 2009).
Elaboración: Autor.

En el artículo de Su y Ai (Su & Ai, 2011) se propone una técnica para la creación de mosaicos mediante sensores remotos montados en un vehículo aéreo no tripulado a partir de imágenes multi-espectrales y características SIFT (Figura 15). Este trabajo combina la extracción de puntos característicos SIFT con la correspondencia aproximada BBF (Beis & Lowe, 1997) y también se basa en una evaluación del desempeño RANSAC (Choi et al., 1997) para la eliminación de errores. Esta técnica obtiene un gran número de características correspondientes y una matriz de transformación estable para un mosaico de imágenes adicional. Los resultados del número de correspondencias halladas en cada grupo de imágenes y el valor medio cuadrático, es decir el error de píxeles se presenta en la Tabla 5.

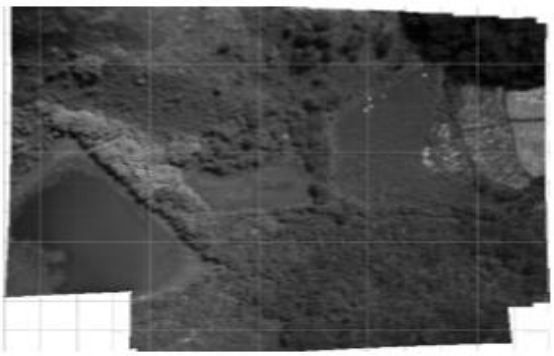


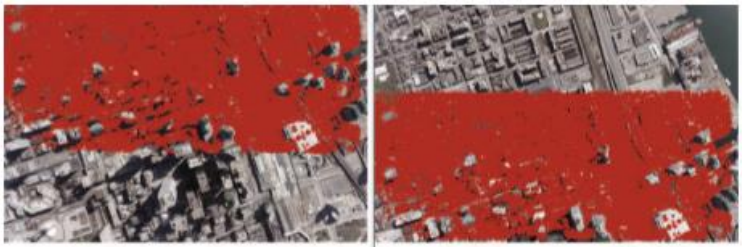
Figura 15. Mosaico con el método SIFT adaptativo.
Fuente: Tomado de (Su & Ai, 2011).
Elaboración: Su, J. y Ai, M.

Tabla 5. Resultados del número de correspondencias calculadas.

Grupo de imágenes	Número de imágenes	Número de puntos	Correspondencias	Valor cuadrático medio (RMS)
1	12	3109	58	2,2
2	12	3265	60	2,3
3	12	3245	59	2,1
4	12	3393	59	1,9
5	12	1292	27	2,1
6	12	3135	56	1,7
7	12	3077	58	1,9
8	12	3391	60	1,5
9	12	2946	54	1,6
10	12	3348	58	1,7
Promedio RMS (Unidad: Píxel)				1,9

Fuente: Tomado de (Su & Ai, 2011).
Elaboración: Autor.

Una de las limitaciones que presenta SIFT es el uso de una gran cantidad de memoria, para lo cual, Sun, Zhao, Huan y Dissanayake (Y. Sun, Zhao, Huang, Yan, & Dissanayake, 2014) diseñan un método denominado Block-SIFT realizando una comparación entre el código abierto SiftGPU (Wu, 2007) y el algoritmo propuesto L^2-SIFT. Cada imagen de gran tamaño se la divide en bloques y los posibles bloques correspondientes de la otra imagen, son determinados pre-estimando la transformación relativa entre ambas imágenes y haciendo uso de una estructura de datos tipo árbol rojo-negro para la reducción de la complejidad de búsqueda al momento de hacer coincidir las correspondencias (Figura 16). Mientras que, para la evaluación de la precisión de las características extraídas y adaptadas del algoritmo planteado, se aplica un ajuste de haz con parametrización de ángulo de paralaje (ParallaxBA) para obtener el error medio cuadrático de las re-proyecciones de características. Los resultados se presentan en la Tabla 6.



(a)

(b)

Figura 16. a) Puntos característicos extraídos mediante L^2-SIFT
b) Correspondencias entre ambas imágenes.
Fuente: Tomado de (Y. Sun et al., 2014).
Elaboración: Sun, Y.

Tabla 6. Análisis de la precisión del algoritmo L^2-SIFT.

Imágenes de las ciudades	Puntos característicos	Correspondencias	Outliers	MSE
Toronto	239279	113685	0	0,0
Vaihingen	1201982	554169	0	0,1
DunHuan	597289	250782	2	0,2
Jinan	2864740	1228959	52	0,1
Village	4098528	1544021	58	0,1
College	3107524	1236502	75	0,7
Taian	6016649	2743553	148	0,0

Fuente: Tomado de (Y. Sun et al., 2014)
Elaboración: Autor.

En el trabajo de Koch, Zhuo, Reinartz y Fraundorfer (Koch, Zhuo, Reinartz, & Fraundorfer, 2016) se propone un método para la mejora de correspondencias aplicando un esquema de coincidencia de uno a varios y añadiendo restricciones geométricas para lograr coincidencias geométricas correctas en regiones repetitivas de la imagen. Este método está diseñado para la correspondencia de imágenes donde la profundidad de escena es escasa, algo que es típico al usar vehículos aéreos no tripulados.

En el documento de Liu y Wen (Liu & Wen, 2017) se trabaja conjuntamente con SIFT y un mejoramiento del algoritmo PROSAC para calcular la matriz de transformación H, mejorando la eficiencia en comparación a RANSAC, ya que detecta una mayor cantidad de “inliers” y para completar el mosaico se utiliza un método de suavizado. Este algoritmo evita las desventajas del método tradicional para la realización de mosaicos bajo distintas condiciones de escala e iluminación.

2.3.1.2. Creación de vistas panorámicas utilizando SURF.

Otro algoritmo detector y descriptor que es muy utilizado en aplicaciones como la creación de mosaicos de imágenes es SURF, ya que posee gran robustez para la extracción de características, siendo una alternativa a SIFT.

En el artículo de Hong (Hong et al., 2009) se propone un algoritmo para la creación de un mosaico de imágenes basado en características SURF (Figura 17), aplicando también el algoritmo RANSAC para eliminar errores de correspondencia y obtener una óptima matriz de transformación entre las imágenes. Como resultado se puede observar que este algoritmo supera los métodos tradicionales para la creación de mosaicos de imágenes que son susceptibles a diferentes objetos en escala, movimiento y posee una gran mejora en cuanto a cálculo, demostrando ser un método rápido y efectivo (Tabla 7).

(a)

(b)

Figura 17. a) Par de imágenes y b) Mosaico creado con SURF.
Fuente: Tomado de (Hong et al., 2009).
Elaboración: Hong, J.

Tabla 7. Comparación del rendimiento computacional de SURF.

Imágenes	Algoritmo	
	Algoritmo 1 (ms)	Algoritmo 2 (ms)
a, b	1063	507
c, d	1102	536

Fuente: Tomado de (Hong et al., 2009).
Elaboración: Autor.

Existen algunos métodos para la detección de puntos característicos y la creación de imágenes panorámicas. Uno de ellos es OpenSURF, el cual fue introducido por Evans (Evans, 2009), encontrándose en la biblioteca de código abierto para el procesamiento de imágenes y visión por computador y es utilizado en el artículo presentado por Wang y Watada (Wang & Watada, 2015) para la creación de imágenes panorámicas.

Una de las desventajas que posee SURF es la baja precisión y poca aplicabilidad en imágenes con gran rotación. Con el fin de resolver estos problemas, Long, Chen y Bao (Long, Chen, & Bao, 2013) proponen una mejora de este algoritmo a través del agrupamiento en lugar de RANSAC para eliminar puntos erróneos que puedan causar una mala correspondencia entre las imágenes. El algoritmo propuesto mejora la precisión y la eficacia de las correspondencias, pero la estabilidad del algoritmo necesita ser mejorada como también el ajuste del valor de los parámetros (Tabla 8).

Tabla 8. Comparación del método tradicional SURF con el algoritmo mejorado.

Imagen	Algoritmo	Número de correspondencias	Tiempo del algoritmo mejorado (ms)	Tiempo total del algoritmo (ms)
1	Tracional	249	795	21276
	Mejorado	233	229	17497
2	Tradicional	90	5331	56994
	Mejorado	191	2210	49848

Fuente: Tomado de (Long et al., 2013).
Elaboración: Autor.

En el artículo de Lei y Wang (Lei & Wang, 2014) se presenta un método rápido para la creación de mosaicos basados en SURF. Este método permite la extracción de los puntos característicos usando SURF y el algoritmo BBF (Beis & Lowe, 1997) que es el encargado de encontrar de manera eficiente una solución aproximada en la búsqueda del vecino más cercano en imágenes con una gran resolución. Seguidamente, se estima la relación de conversión entre imágenes utilizando RANSAC y el método de mínimos cuadrados (MSE); y por último, se utiliza el algoritmo de gradación de entrada y salida (Szeliski, 1996) para reducir las variaciones de iluminación al momento de fusionar la imagen.

2.3.2. *Técnicas de detección de vehículos.*

El excesivo tráfico vehicular es un tema que preocupa a varias ciudades y que se desea resolver mediante la búsqueda de mecanismos precisos y seguros para establecer las variables para un mejor control de la congestión vehicular. Es así como la detección de vehículos por medio de visión por computador u otras tecnologías resultan una solución óptima debido a su eficiencia para reducir el tráfico en cada una de las calles de la ciudad.

Para prevenir este problema, Zhao y Nevatia (Zhao & Nevatia, 2003) presentan un sistema para la detección de automóviles a lo largo de la calle en imágenes aéreas donde los autos son objetos muy pequeños y formulan el problema como reconocimiento de objetos en 3D, teniendo en cuenta características como: límite del auto, límite del parabrisas delantero y la sombra producida por el vehículo. Estas características son afectadas por la intensidad en la iluminación del auto y por la sombra que produce el mismo, por lo cual, esta información se representa como un modelo bayesiano para integrar todas las características.

En el artículo presentado por Agrawal y Hickman (Agrawal & Hickman, 2004) se plantea una metodología para obtener estimaciones de longitud de una fila de automóviles a partir de una imagen aérea tomada a 1800 pies (548,64 m) sobre el nivel del suelo, mediante el análisis de componentes conectadas en el umbral de la región de interés. Las estimaciones de la fila de vehículos se obtienen mediante dos técnicas: por el conteo individual de los vehículos en la carretera y por el área del polígono que contiene los vehículos.

Xiyan Cheng y Quinggan Meng (Chen & Meng, 2013) formulan un método para la detección vehicular con la ayuda vehículos aéreos no tripulados donde integra el descriptor SIFT y un Modelo de Tipo Implícito (ISM). El resultado presenta a este método con una tasa de exactitud del 94,13 %, siendo afectado por el tamaño de las imágenes de entrenamiento, ya que mientras más muestras de entrenamiento se tiene, se obtendrá una mayor precisión en la detección (Figura 18). En cuanto a la velocidad de procesamiento es bastante lento comparada con el procesamiento en tiempo real, ya que SIFT necesita procesar cada punto clave para la detección de los vehículos.

Figura 18. Resultado final de la detección de vehículos.
Fuente: Tomado de (Chen & Meng, 2013).
Elaboración: Chen, X. y Meng, Q.

El artículo de Gleason (Gleason, Nefian, Bouyssounousse, Fong, & Bebis, 2011) se centra en la detección de vehículos terrestres en entornos rurales para la detección automática de amenazas de oleoductos y gasoductos, mediante la ayuda de vehículos aéreos no tripulados. Todo esto se lo realiza con la detección en cascada, siendo una solución para la rápida detección de vehículos, rechazando la mayor parte del fondo en la imagen, y utilizando técnicas de clasificación de imágenes, siendo usados los siguientes: KNN (Cover & Hart, 1967), SVM (Hearst, Dumais, Osuna, Platt, & Scholkopf, 1998), árboles de decisión y árboles aleatorios (Breiman, 2001). Mientras que en la parte de extracción de características utilizan técnicas como: histograma de gradientes (HoG) (Dalal & Triggs, 2005) y coeficientes de Gabor (Zehang Sun, Bebis, & Miller, 2006).

El artículo de Xu (Xu, Yu, Wu, Wang, & Ma, 2016) propone un método avanzado para la mejora en la detección de vehículos mediante el algoritmo de detección de objetos Viola-Jones (Viola & Jones, 2001) desde un vehículo aéreo no tripulado a baja altura. Sin embargo,

se observa que el algoritmo posee dificultades como la sensibilidad a la rotación y la detección de vehículos con orientaciones desconocidas en las imágenes aéreas (Figura 19). Para ello se plantea solucionar este problema mediante el ajuste de la orientación de la carretera, rotando cada imagen aérea captada, de manera que la carretera y los vehículos queden alineados de manera horizontal para ser detectados con dicho algoritmo (Figura 20). Este método presenta dificultades en cuanto a iluminación y la detección de buses, camiones, motocicletas, entre muchos otros medios de transporte.

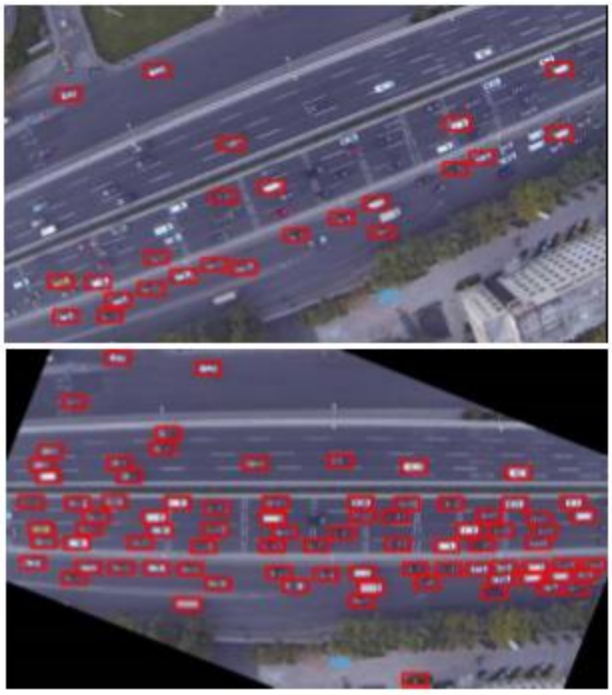


Figura 19. Detección de vehículos usando: a) Método V-J original b) Método mejorado V-J
Fuente: Tomado de (Xu et al., 2016).
Elaboración: Xu, Y.

Figura 20. Detección de vehículos usando bajas condiciones de iluminación.
Fuente: Tomado de (Xu et al., 2016).
Elaboración: Xu, Y.

2.3.3. *Técnicas para la implementación de semáforos inteligentes.*

En varias ciudades alrededor del mundo se presenta el problema de congestión vehicular. En vista de ello, se ha implementado sistemas de semaforización distintos a los métodos tradicionales con el fin de evitar una gran aglomeración en horas pico.

Una de estas ciudades que se ve afectada por el tráfico vehicular es Chiclayo, ubicado en el vecino país de Perú. Para ello en el trabajo de Santamaría y Moscol (Santamaría & Moscol, 2015) se realiza un prototipo de un sistema de semaforización basado en la inteligencia artificial, específicamente mediante la utilización de técnicas de lógica difusa, tomando como base el número de vehículos que ocupan un segmento de longitud específico de una vía. Los resultados obtenidos han sido favorables en la intersección dónde se han realizado las pruebas.

Otra alternativa de semaforización inteligente es el uso de redes de sensores inalámbricos (WSN). El artículo de Zhou, Cao, Zeng y Wu (Zhou, Cao, Zeng, & Wu, 2010) presentan un algoritmo para la mejora al sistema de semaforización a través de datos recolectados por una red de sensores inalámbricos en tiempo real. Esta solución ha sido implementada en una intersección con cuatro direcciones, dónde cada calle posee dos carriles; para ello el algoritmo detecta el número de vehículos y determina el tiempo que se encenderá la luz de color verde en cada carril. Esta alternativa demuestra un mayor rendimiento y menor tiempo de espera promedio para los vehículos en una intersección.

En el trabajo de Wen (Wen, 2008) se trata de solucionar este tipo de inconvenientes mediante el uso de semaforización inteligente utilizando tecnología RFID. Para conocer cómo se establecen los tiempos de encendido de las luces verdes y rojas del semáforo, se ha diseñado seis sub-modelos para la simulación de problemas generados por el tráfico en horas pico. Cada sub-modelo representa una avenida con tres intersecciones en una zona urbana. La eficiencia del algoritmo propuesto se ha visto reflejada al ser implementada y se ha observado que el tiempo de espera bajó de manera brusca cuando la duración de la luz de color rojo ha sido de 65 segundos y la duración de la luz verde de 125 segundos.

En el trabajo presentado por Chávez (Chávez, 2015) se desarrolla un sistema en la ciudad de Ambato, constituido por dos cámaras que adquieren las imágenes de las áreas de interés para el control de los tiempos de duración de las fases de los semáforos que está ligado con el número de vehículos existentes en la vía a través del procesamiento de imágenes. Para la extracción y envío de la información hacia cada semáforo se hace uso de la placa Arduino.

CAPÍTULO III.
MATERIALES Y MÉTODOS

3.1. Análisis de requerimientos

Una vez que se ha analizado los conceptos sobre las posibles soluciones para el conteo vehicular a través de imágenes aéreas, se definió que el sistema consta de cuatro fases fundamentales: adquisición, procesamiento, aplicación e implementación (Figura 21).

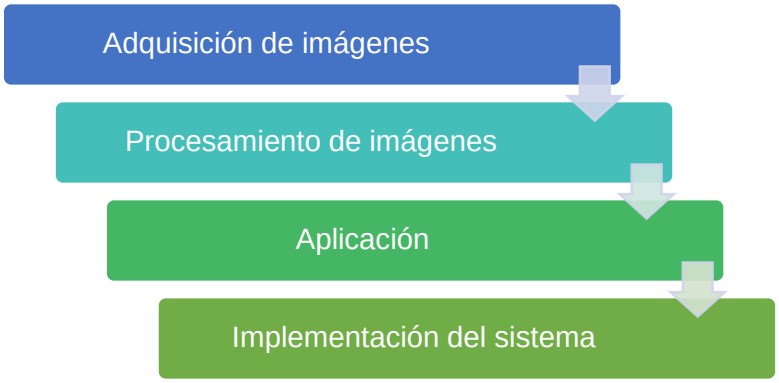


Figura 21. Fases del proyecto.
Fuente: Propia del autor
Elaboración: Autor.

3.1.1. Programas.

Para el desarrollo del presente trabajo de fin de titulación se requiere de un programa que contenga librerías para el procesamiento de las imágenes, capaz de crear una interfaz para el diseño de la aplicación, pero sobre todo capaz de comunicarse con tarjetas para la adquisición de datos para la realización de las pruebas correspondientes.

También se debe tener en cuenta los algoritmos de visión por computador, tales como detectores y descriptores que ayuden a encontrar las características locales en una imagen y que son de gran ayuda para la creación de mosaicos a partir de las imágenes aéreas captadas.

De la misma manera se pretende trabajar con grafos para la simulación del sentido de las vías y que permita al usuario encontrar una ruta con menor congestión dentro del casco urbano.

3.1.2. Equipos físicos.

Para el funcionamiento de las presentes soluciones son necesarios los siguientes dispositivos:

- **Vehículo aéreo no tripulado:** requiere que sea capaz de mantener de manera autónoma un nivel de vuelo controlado remotamente y sostenido.
- **Computador:** entre los requerimientos, debe contar con un procesador que sea de alto nivel para poder trabajar en tiempo real con las imágenes obtenidas.
- **Cámara:** debe poseer una alta resolución, que sean de tipo digital y fácil de montar en el vehículo aéreo no tripulado.

- **Semáforos:** el prototipo se orienta en semáforos de tipo vehicular, para lo cual se necesita de dispositivos que simulen su funcionamiento.
- **Tarjeta de adquisición de datos:** esta tarjeta tiene que ser capaz de obtener los datos resultantes desde el programa utilizado para el procesamiento de imágenes, y de esta manera ser enviados hacia cada semáforo.

3.2. Elección del programa

Se ha optado por el programa MATLAB, el cual cumple con todas las características mencionadas anteriormente; siendo una herramienta altamente robusta, amigable con el usuario y no necesita de una gran cantidad de memoria, con una Caja de Herramientas también conocido como “Toolbox” (conjunto de instrucciones con un fin en particular) que es específicamente dedicado al procesamiento de imágenes con algoritmos robustos y veloces, y también cuenta con otro “Toolbox” que permite la comunicación entre MATLAB y la tarjeta de adquisición de datos. En este caso se utilizó Arduino, ya que dispone de un entorno de desarrollo, de abundante información en su página web (Arduino, 2016) y un paquete para soporte para Arduino (MathWorks Simulink Team, 2014) para la comunicación con el programa MATLAB.

Para la elección de los algoritmos detectores y descriptores se realizó una comparación en cuanto a la velocidad de procesamiento, detección de puntos característicos y memoria requerida, siendo escogidos SIFT y RANSAC, por su alto nivel de detección de puntos clave y capacidad de eliminar los denominados puntos erróneos detectados, respectivamente. Estos dos métodos poseen una abundante cantidad de trabajos relacionados, ya que son altamente robustos y confiables, facilitando la creación de vistas panorámicas tipo mosaico.

Para la búsqueda de la ruta más descongestionada se escogió cuatro tipos de métodos para localizar un camino alternativo con menor congestión vehicular, siendo estos los siguientes: Dijkstra, Breadth-First Search, Bellman-Ford y el método de aciclicidad. Estos métodos también vienen incorporados en el programa MATLAB.

3.3. Elección de los elementos físicos

Para la elección del vehículo aéreo no tripulado que ayude al sobrevuelo por el centro histórico de la ciudad, se ha optado trabajar con un drone de marca DJI Phanthom 3 Standard. Esto debido a las facilidades que presentó como la disponibilidad y las prestaciones como la integración de una cámara con una resolución de 12 megapíxeles para la captación de las imágenes y una aplicación tanto para Android como para iOS capaz de presentar en pantalla la imagen en tiempo real de la zona que se quiera captar, así como también la información de la altura a la que se encuentra y sus coordenadas.

En cuanto al ordenador, se ha trabajado con dos equipos que se dispuso al momento de realizar el presente trabajo con las siguientes características: el primero con procesador Intel Core i5-4210U a 2.4 GHz y 12 GB de memoria RAM y el segundo ordenador con procesador Intel Core i7-6700HQ a 2,6 GHz, con 8 GB de memoria RAM y una tarjeta de vídeo Nvidia GeForce GTX 960M con 4 GB de memoria. Cabe recalcar que solo el segundo ordenador posee la tarjeta de vídeo.

En cuanto a la elección de la tarjeta de adquisición de datos, se escogió trabajar con una placa Arduino Mega 2560 debido a que es una plataforma de software libre, el bajo costo que posee, disponibilidad en el medio, reducción de la complejidad de los circuitos electrónicos y por sobre todo la compatibilidad con el programa MATLAB.

3.4. Ubicación del vehículo aéreo no tripulado y semáforos

El sitio en el que se procedió a volar el vehículo aéreo no tripulado para la obtención de las imágenes aéreas de prueba fue la plaza de la Independencia San Sebastián en la ciudad de Loja, debido al gran congestionamiento vehicular que existe en este sector en horas pico.

Para efectos de simulación, se trabajó con los datos obtenidos en el trabajo de Guamán (Guamán, 2012), en donde se presenta las calles con mayor afluencia vehicular y su variación en diferentes horas pico dentro de la ciudad, observando que el mayor conflicto vehicular se concentra en las calles: Bolívar, Bernardo Valdivieso y Olmedo, con niveles de servicios de operación muy críticos en las horas pico analizadas.

La imagen resultante está delimitada por las siguientes intersecciones: Juan José Peña y Alonso de Mercadillo, 18 de Noviembre y Alonso de Mercadillo, Juan José Peña y 10 de Agosto, y, 18 de Noviembre y 10 de Agosto. En cuanto a los semáforos se colocó uno con doble cara en cada intersección dando como resultado 30 semáforos ubicados en cada cruce de las vías.

El motivo porque se delimitó así esta área está basado en los resultados obtenidos en el trabajo mencionado anteriormente (Guamán, 2012), y para efectos de simulación se puede tener problemas en cuanto al tiempo de procesamiento debido al gran tamaño de la imagen que se puede tener al momento de generar el mosaico.

CAPÍTULO IV.
DESARROLLO E IMPLEMENTACIÓN

4.1. Descripción general del sistema

El sistema propuesto en el presente trabajo de titulación tiene como objetivo el conteo vehicular en cada intersección dentro del casco urbano y la toma de decisiones de cada semáforo en función de los vehículos contabilizados por cada calle. Este sistema está formado por cuatro etapas fundamentales que son: adquisición de las imágenes mediante una cámara digital, el procesamiento de las imágenes, la aplicación mediante una GUI (Interfaz gráfica de usuario) en MATLAB, la implementación del algoritmo para la presentación en pantalla de la ruta con menor congestión según el destino al que desee movilizarse el usuario.

Seguidamente, el sistema se pone en marcha al momento de realizar el mosaico de imágenes aéreas. Se procede a elegir un mapa en la GUI, dónde cada mapa posee un escenario con distinto nivel de tráfico. Cabe recalcar que, por falta de disponibilidad del vehículo aéreo no tripulado, se replicó de manera manual el mosaico obtenido de la plaza de San Sebastián hasta lograr el mosaico del casco urbano de la ciudad de Loja ya que al contar con un solo vehículo aéreo no se logró captar todas las imágenes del centro histórico. De la misma manera, se procedió a colocar o quitar los vehículos de manera manual en cada calle para la simulación de los distintos escenarios cuyo tráfico vehicular depende del escenario.

Seleccionado el escenario en la GUI, se realizó el procesamiento de la imagen con el fin de contabilizar el número de vehículos por cada calle del mapa para que cada semáforo tome una decisión para permitir o no el paso de vehículos y a su vez la recomendación de la calle con menor congestión. Todo esto se lo realizó mediante la programación de una GUI en el programa MATLAB (ver Anexo 2).

Otra opción que presenta la aplicación es el poder encontrar la ruta con menor congestión. Para ello el usuario tiene la opción de elegir semáforo de origen y el semáforo de destino. De esta manera el algoritmo procede a encontrar la ruta o las rutas alternativas si existe el caso con menor congestionamiento vehicular.

4.2. Construcción de una vista panorámica tipo mosaico

Para la construcción de la vista panorámica tipo mosaico, se recurrió al código "*Construcción de vistas panorámicas tipo mosaico"* (Aguirre, 2012) desarrollado en el programa MATLAB (ver Anexo 1) y se utilizaron 4 imágenes aéreas tomadas con la ayuda de un vehículo aéreo no tripulado, esto en cada esquina de la plaza de la Independencia San Sebastián de la ciudad de Loja en horas pico (07h00 – 08h00, 12h00 – 14h00 y 17h00 – 18h00) debido a que en estas horas existe un mayor tráfico vehicular. El tamaño de cada imagen capturada es de 4000 píxeles en cada fila y 3000 píxeles en cada columna. Las imágenes se redujeron un 10 %, es decir, se trabajó con imágenes con un tamaño de 3600×2700 píxeles, debido a

problemas que existieron por falta de memoria RAM de los ordenadores en los que se trabajó, ya que dispusieron de una memoria de 8 GB y 12 GB respectivamente. Se realizó pruebas con imágenes en escala de grises y también se descompuso la imagen en sus canales de colores: rojo, verde y azul; esto con el fin de observar en que canal se obtenía mejores resultados para la detección de puntos característicos y correspondencias.

Las herramientas de programación utilizadas han sido el Toolbox VLFeat (Vedaldi & Fulkerson, 2010), el Toolbox BALU (Mery, 2015) y el Toolbox de Procesamiento de Imágenes de MATLAB, los cuales presentan algoritmos y funciones para un mejor tratamiento de las imágenes.

El diseño para obtener el mosaico resultante está basado en encontrar los puntos característicos de dos imágenes y encontrar la mejor correspondencia entre estos puntos SIFT, y con ello calcular la mejor homografía aplicando RANSAC para finalmente obtener la imagen fusionada. Las imágenes deben seguir un orden para obtener una correspondencia de puntos entre ellas. En este caso se tuvo un total de tres correspondencias. La primera correspondencia se la realizó entre la primera imagen captada ubicada en el extremo superior izquierdo y la segunda del extremo inferior izquierdo. El mismo proceso se realizó con la imagen ubicada en el extremo inferior derecho pero esta vez, la correspondencia se detectó con el primer resultado obtenido; y, de la misma forma se efectuó el proceso mediante la segunda imagen resultante y la última imagen ubicada en el extremo superior derecho, obteniendo así el mosaico final.

Para encontrar los puntos característicos SIFT y sus descriptores, se utilizó la función *vl_sift,* pero antes de aplicar esta función, las imágenes se convirtieron a escala de grises y mediante cada canal de color que ha sido descompuesta la imagen, ya que de esta manera se obtiene un valor normalizado en el intervalo [0 , 255]. Se trabajó en formato “single” con valores en un rango de [-10^{38} , 10^{38}] (4 bytes por elemento) debido a que los valores por defecto de algunos umbrales están ajustados para este caso. La salida de esta función se obtiene en dos vectores, dónde el primer vector posee 4 filas con información acerca de: la ubicación tanto en x como en y, la escala y la orientación dada en radianes. El número de columnas depende del número de puntos encontrados y el segundo vector contiene información de los descriptores encontrados para cada punto (columnas) y 128 elementos con valores de medidas de orientación o gradiente (filas).

Para el establecimiento de correspondencias entre los puntos SIFT de cada par de imágenes, se hizo uso de la función *vl_ubcmatch*, la cual entrega a su salida dos vectores, dónde el primero indica la correspondencia existente entre los puntos encontrados anteriormente de la segunda imagen y la primera imagen; y, el segundo vector proporciona su distancia euclidiana

al cuadrado. Otra de las opciones que presenta este comando es el umbral, el cual significa que dos descriptores tanto de la primera como la segunda imagen son correspondidos entre sí, sólo si la distancia entre ambos descriptores multiplicada por el umbral no es mayor que la distancia del descriptor de la primera imagen con todos los demás descriptores.

En el siguiente paso se calculó la homografía que transforma la segunda imagen en una correspondencia con la primera imagen. Para ello se procedió a convertir los datos obtenidos a partir de los puntos SIFT en un vector de tipo homogéneo de tres filas por el número de columnas que corresponda. Este procedimiento se aplicó en ambas imágenes, el mismo que sirve como dato de entrada para la utilización de la función *Bmv_homographyRANSAC*. Esta función tiene como finalidad la estimación de la matriz de homografía o de transformación (H) que es una transformación proyectiva que determina la correspondencia entre ambas imágenes, utilizando el modelo RANSAC, la cual se procedió a normalizar mediante la división de esta matriz para el valor obtenido (factor de escala) en el elemento h_{33} de la matriz de transformación H. Los valores contenidos en h_{13} y h_{23} entregan la información de translación, t_x y t_y, correspondiente de la segunda imagen con relación a la primera. El error de píxeles se calculó entre la primera y la segunda imagen y significa la máxima diferencia encontrada entre los puntos SIFT de una imagen al ser correspondida con la otra imagen.

El establecimiento del tamaño de la imagen resultante de la fusión entre el par de imágenes, se calculó por medio de los cuatro puntos transformados de la imagen. En este caso la imagen trasformada es la segunda imagen al espacio de la primera. De esta manera se conoce la ubicación en la que se encontrarán los extremos de la imagen transformada en el espacio de la primera imagen, mediante la multiplicación de estos puntos por la inversa de la matriz de transformación H.

Luego de esto se estableció una rejilla de valores, considerando como inicio el mínimo valor de columnas entre la primera imagen y la transformada de la segunda imagen, y como final, el valor máximo de columnas de igual manera entre el extremo de la primera imagen y la transformada de la segunda imagen. De la misma forma, para encontrar los valores para las filas, se estableció la rejilla mediante la función *meshgrid* con la información de ambas imágenes a fusionarse. Para ello se calculó la rejilla para el par de imágenes con la misma función antes mencionada *meshgrid*, con la diferencia de que el espacio de la segunda imagen debe ser transformado multiplicando las matrices del nuevo espacio, por los valores de la transformada H.

Finalmente, se asignó la información contenida en cada píxel para ambas imágenes y se procedió a realizar la interpolación mediante la función *interp2*. Esta función posee como característica, el poder escoger el método para realizar la interpolación, siendo estos: vecinos

más cercanos, bilineal, bicúbica y spline. Este último método se lo descartó ya que no obtuvo los resultados deseados.

A continuación, en la Figura 22, se presenta un esquema de lo mencionado anteriormente para la creación de una vista tipo mosaico.

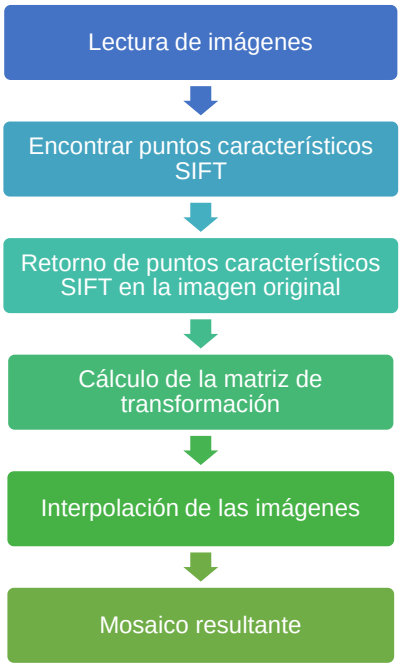


Figura 22. Método para la elaboración del mosaico.
Fuente: Imagen propia del autor.
Elaboración: Autor.

4.3. Diseño de una interfaz gráfica de usuario en MATLAB

Una vez creado el mosaico a partir de imágenes aéreas, se programó una GUI (Interfaz Gráfica de Usuario) en el programa MATLAB con el fin de que el usuario tenga la posibilidad de interacturar con la aplicación de una manera más amigable y eficaz.

La GUI programada cuenta con las siguientes opciones: selección del escenario, selección de cualquier intersección dentro del casco urbano para realizar un acercamiento y la posibilidad de escoger un semáforo de origen y un semáforo de destino dentro del casco urbano para la presentación en pantalla de la ruta con menor nivel de congestión.

4.3.1. Selección del escenario.

El primer paso es cargar la GUI y la primera opción que se le presenta al usuario es poder escoger entre los diez tipos de escenarios distintos como se presentan en la Figura 23 y Figura 24, donde la primera opción (Mapa 1), contiene un escenario sin ningún vehículo en las calles señalado con el color verde, de esta manera, la disponibilidad de cada calle será del 100 % y el número de vehículos va en incremento de manera proporcional en los siguientes escenarios

hasta el décimo y último mapa (Mapa 10) señalado con un color rojo, el cual contiene un escenario con una gran congestión vehicular.

Una vez seleccionado el escenario, se presenta una barra de proceso ubicado en la parte inferior de la aplicación, la cual indica el estado del proceso y el tiempo estimado restante para la conclusión del proceso.

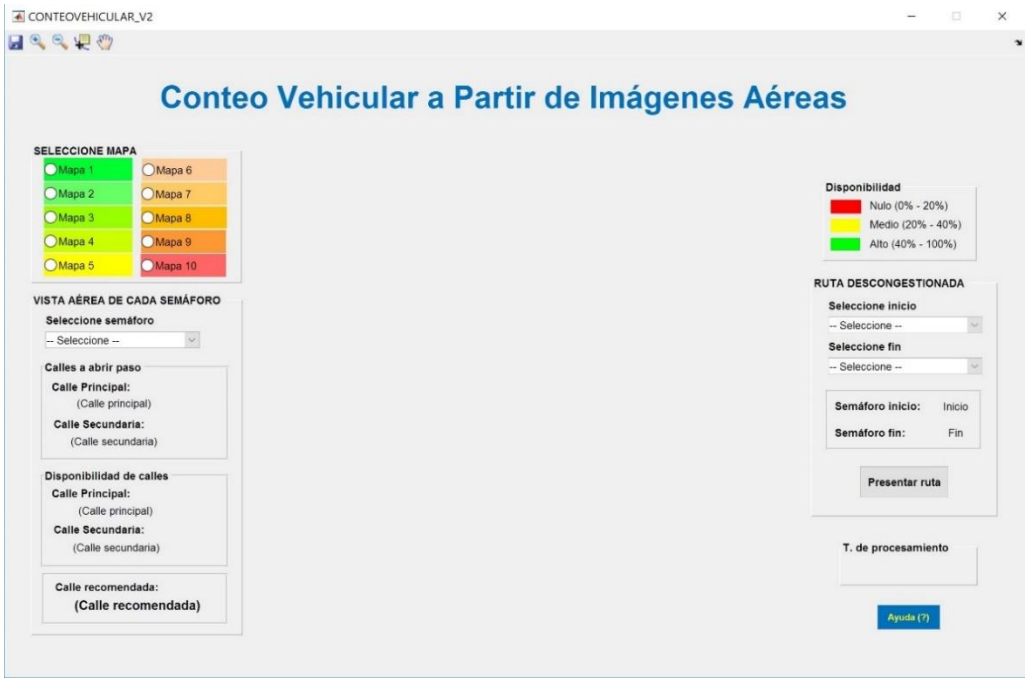


Figura 23. Aplicativo GUI en MATLAB
Fuente: Imagen propia del autor
Elaboración: Autor.

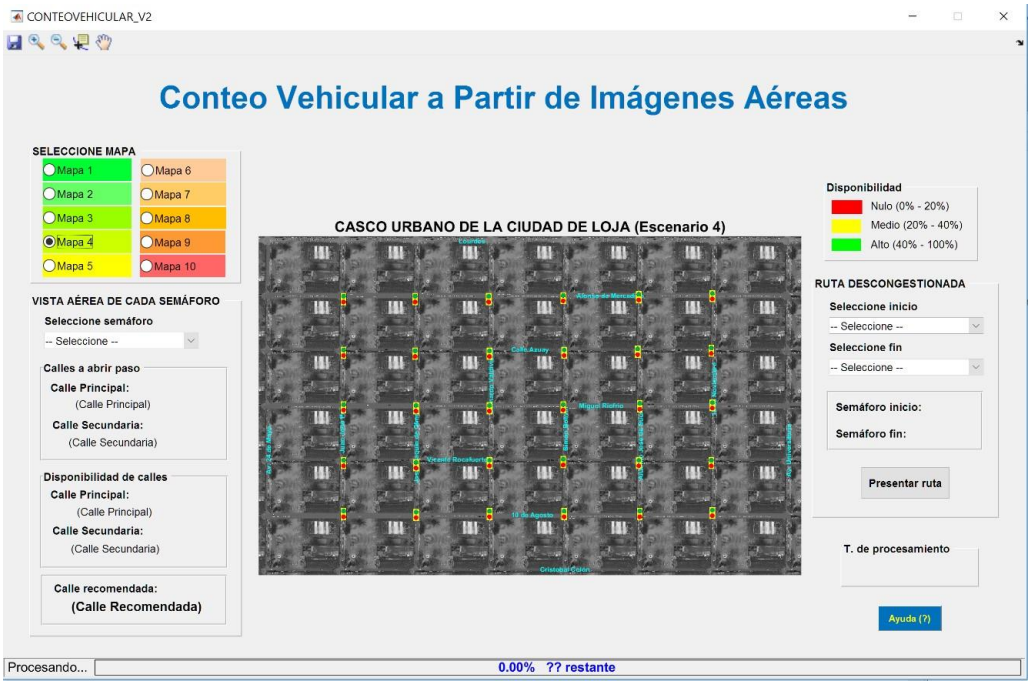


Figura 24. Mapa del casco urbano (escenario 4) y semáforos en la GUI de MATLAB.
Fuente: Imagen propia del autor.
Elaboración: Autor.

4.3.2. Segmentación de la imagen

Una vez seleccionado el escenario, automáticamente el programa procede a pre-procesar la imagen con el fin de mejorar los niveles de contraste e iluminación, ya que, en ciertos casos, la imagen puede verse afectada por las sombras producidas por el entorno, así como también ayudar a la detección de vehículos que son muy oscuros o poseen niveles de gris muy cercanos a los niveles de la calle. Para ello se procedió a visualizar la distribución de los píxeles mediante la ayuda del histograma de la imagen en escala de grises, donde una vez conocidos los valores del nivel de intensidad de los píxeles de los autos, se recorre la imagen en búsqueda de los píxeles con niveles de intensidad bajos para que obtengan un nivel mayor y así los vehículos puedan ser detectados mediante el algoritmo propuesto.

Para ilustrar esto, en la Figura 25 y Figura 26 se expone todo lo mencionado anteriormente.

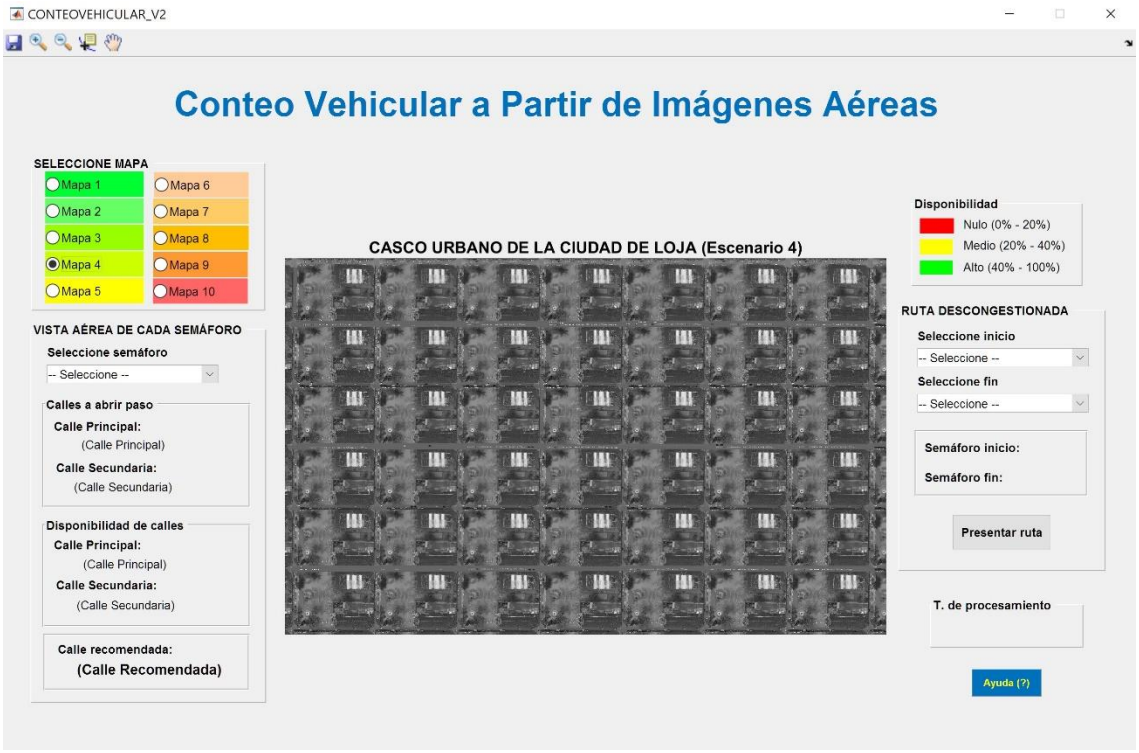


Figura 25. Mosaico de imágenes seleccionado y cargado en la GUI.
Fuente: Imagen propia del autor.
Elaboración: Autor.

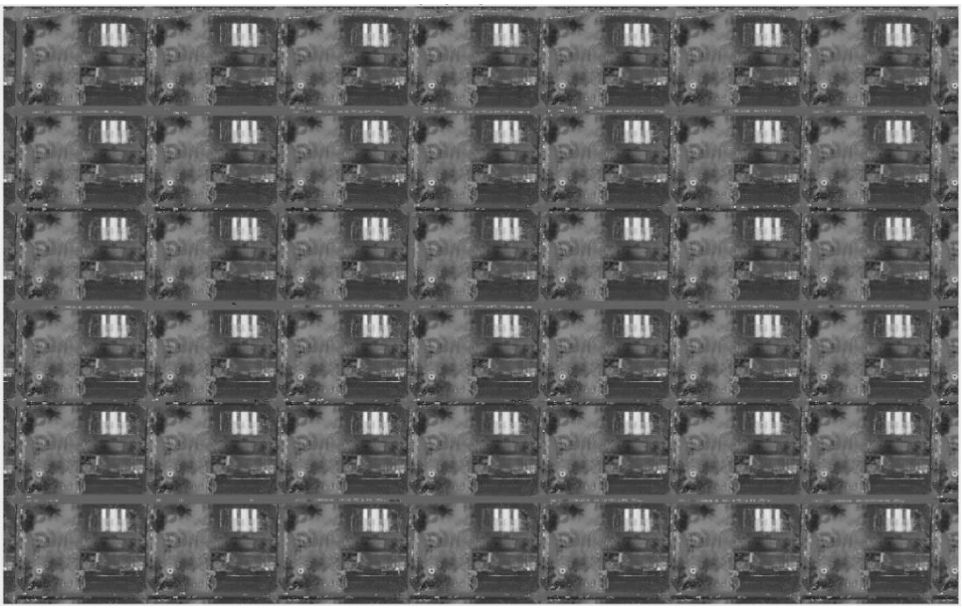


Figura 26. Mosaico de imágenes pre-procesado.
Fuente: Imagen propia del autor
Elaboración: Autor.

El siguiente paso es estimar el fondo de la imagen (Figura 27), mediante el uso de la apertura morfológica, con la ayuda la función *imopen* y un elemento estructurante “disk”, la cual viene incluida dentro del Toolbox de Procesamiento de Imágenes del programa MATLAB.

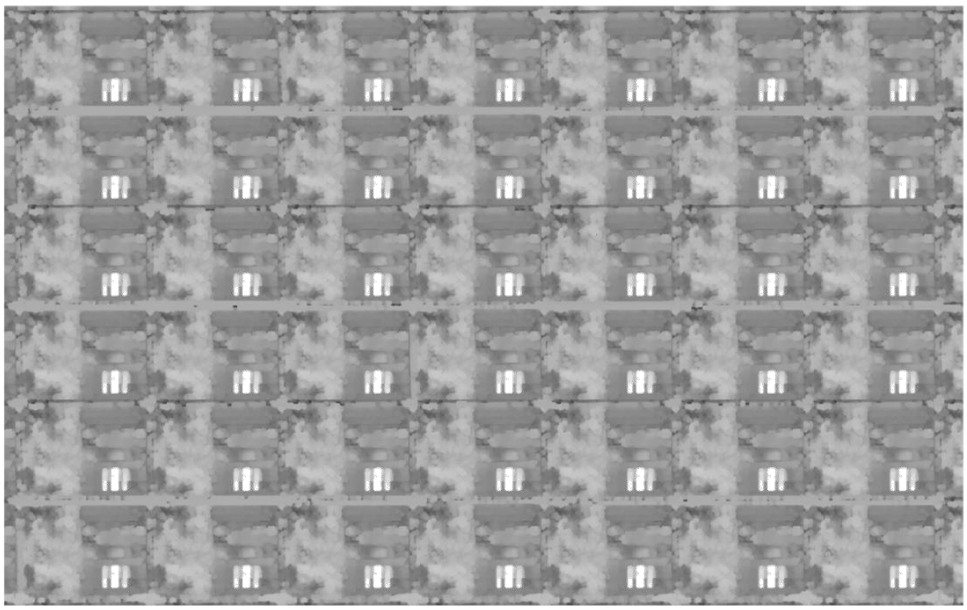


Figura 27. Uso de la apertura morfológica aplicado al mosaico de imágenes.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Seguidamente, a la imagen que contiene el escenario seleccionado se le resta la imagen que contiene el fondo estimado mediante la apertura morfológica, dando como resultado una imagen en la que constan solamente los vehículos (Figura 28). A esta imagen se le aumenta el contraste para facilitar el proceso de binarización.

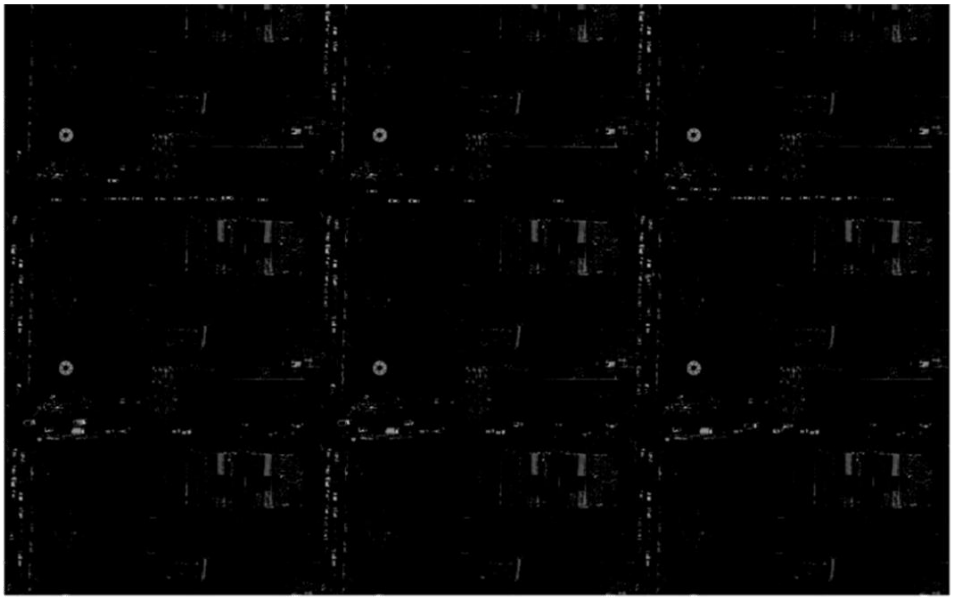


Figura 28. Aumento del contraste de la imagen
Fuente: Imagen propia del autor
Elaboración: Autor.

Para continuar, el proceso de binarización se realizó con la ayuda de dos funciones: *graythresh* e *im2bw*. La primera función toma la imagen en escala de grises y calcula un umbral global mediante la utilización del método Otsu para minimizar la varianza entre los píxeles blanco y negro. La segunda función convierte la imagen en binaria (Figura 29). Todo esto lo realiza según el umbral detectado anteriormente, dónde los píxeles que están bajo el umbral toman el valor de 0 (negro), mientras que los que están por encima del umbral toman el valor de 1 (blanco).

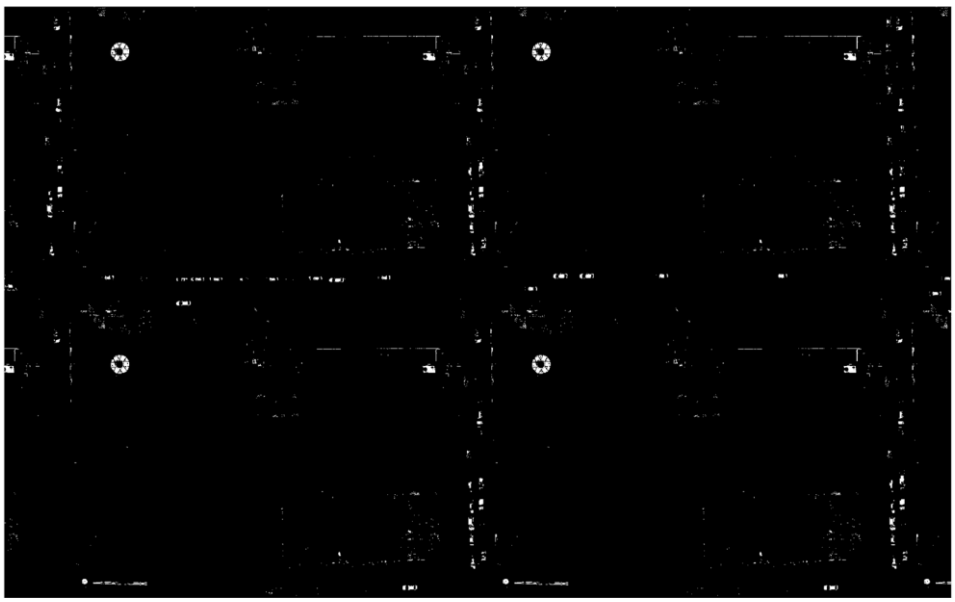


Figura 29. Imagen binaria sin procesar.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Posteriormente, el programa procede a realizar algunas transformaciones morfológicas como la dilatación y la erosión a cada objeto de la imagen. Esto con el fin de detectar de mejor manera los vehículos. Para la dilatación de la imagen se utilizó elementos estructurales, los cuáles definen la forma y el tamaño de la vecindad del píxel a analizarse.

Para el presente trabajo se utilizó dos elementos estructurales, el uno con un ángulo de 90 grados y el otro de 0 grados, dilatándose primero con el elemento estructural vertical, seguido por el elemento estructural horizontal (Figura 30).

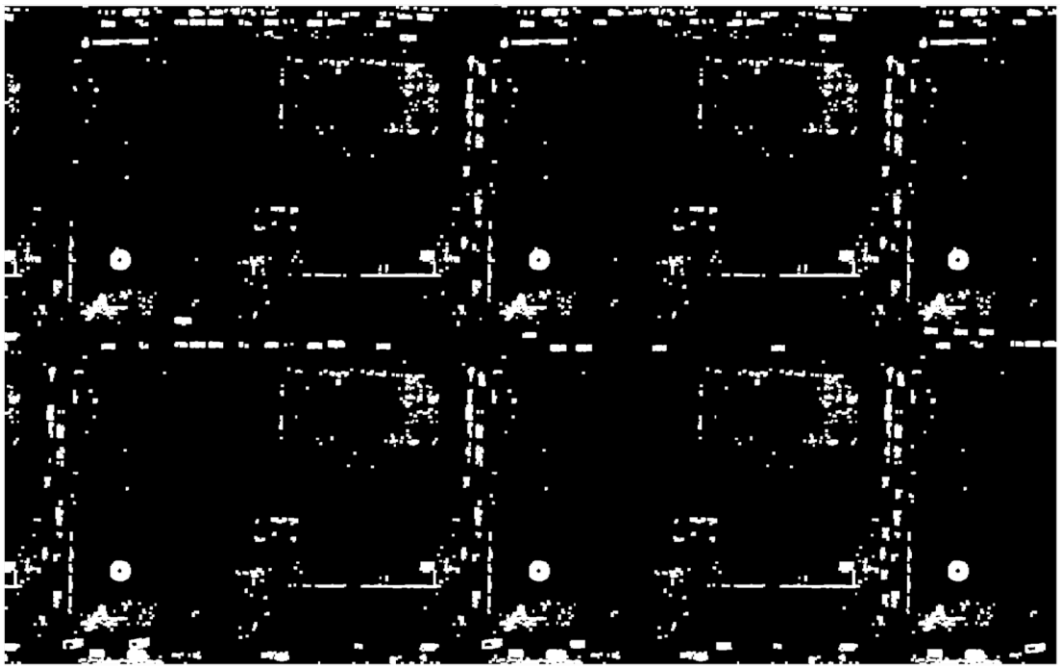


Figura 30. Imagen binaria sin procesar.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Para culminar con la segmentación de la imagen y hacer que los objetos, en este caso los automóviles, tengan un aspecto natural, se suavizó el contorno del objeto mediante la erosión de la imagen con la ayuda de un elemento estructural tipo diamante, el mismo que presentó mejores resultados ante otro tipo de elementos. Todo esto se lo realizó con la función *imerode* (Figura 31) y de esta manera se obtuvo la imagen resultante que se presenta en la Figura 32.

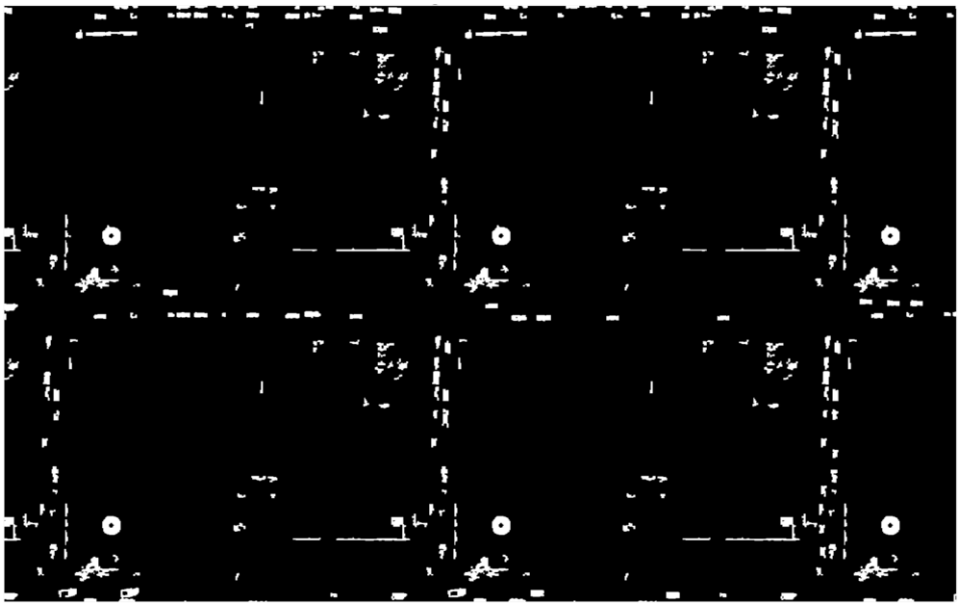


Figura 31. Imagen erosionada.
Fuente: Imagen propia del autor.
Elaboración: Autor.

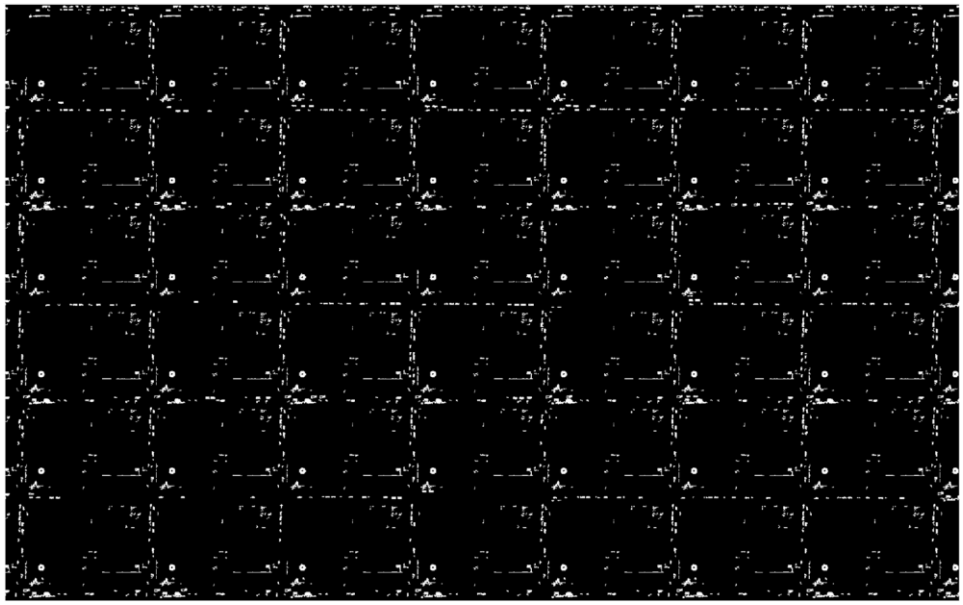


Figura 32. Imagen procesada.
Fuente: Imagen propia del autor.
Elaboración: Autor.

4.3.3. Conteo vehicular.

Una vez finalizada la segmentación de la imagen, el programa realiza el conteo de vehículos en cada calle mediante la función *contabilizarGPU*. Esto se realizó colocando una región de interés (ROI) en cada una de las calles. Con la ayuda de la función *impoly* (Figura 33), se colocó las coordenadas de manera manual en cada una de las calles que fueron de interés para proceder con la contabilización en cada una de las vías, con el fin de evitar detectar mayor cantidad de falsos positivos.

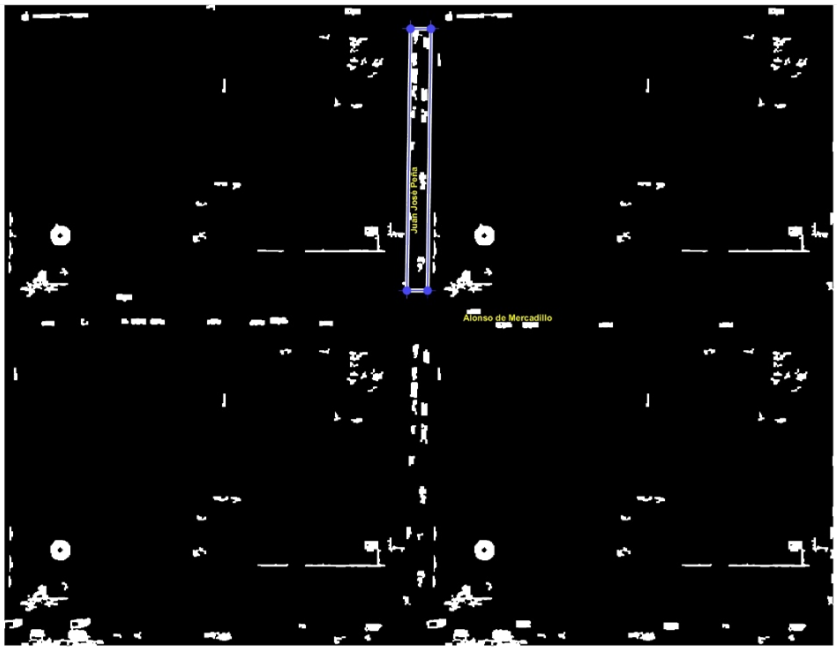


Figura 33. Conteo de vehículos de la ROI seleccionada
Fuente: Imagen propia del autor.
Elaboración: Autor.

Seleccionada la ROI de cada calle, se crea una máscara de tipo binario de la región seleccionada mediante la función *createMask*. De esta manera se multiplica la imagen binaria con la máscara creada, obteniendo así solo los automóviles que son de interés para su posterior contabilización. Esta operación se realizó en cada calle (Figura 34) y mediante la ayuda de la función *gpuArray* se copia la matriz a la GPU para tratar de reducir el tiempo de cálculo.

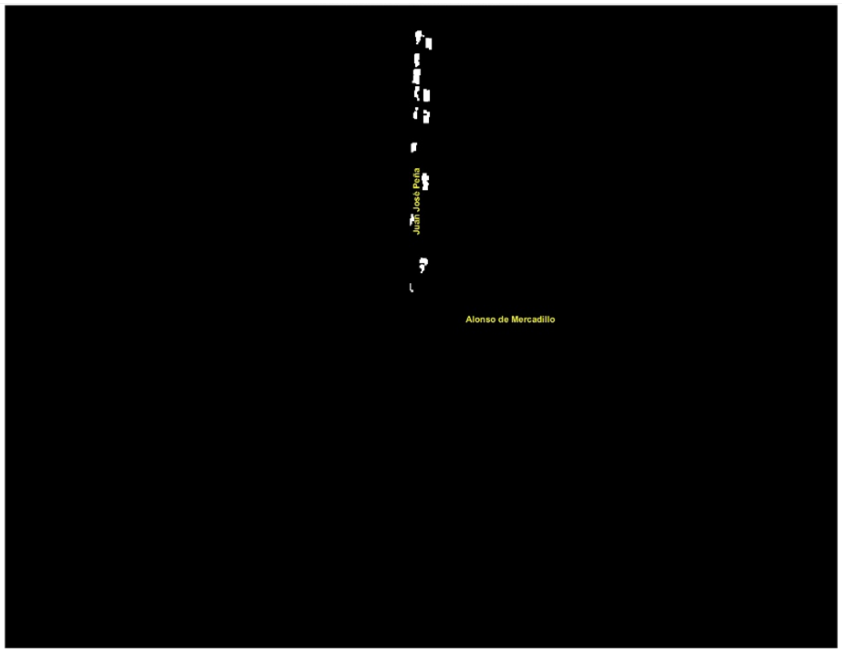


Figura 34. Vehículos a contabilizar por cada calle.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Finalmente, para obtener el número de vehículos por cada calle, se etiquetó cada objeto encontrado con la función *bwlabel*, la cual retorna la matriz etiquetada con los objetos contenidos en la imagen binarizada. De esta manera se obtuvo el número de objetos contabilizados, en este caso el total de vehículos en cada calle a través de la función *max*.

4.3.4. Proceso de semaforización.

Una vez detectados y contabilizados los vehículos por cada calle, el algoritmo es capaz de decidir a que calle debe abrir y cerrar el paso en función de los vehículos contabilizados. Este proceso se realizó en cada intersección.

La Figura 35 presenta un diagrama de flujo para realizar el proceso de semaforización que inicia desde el mosaico de imágenes que ha sido creado y finaliza en la decisión que toma el semáforo en función de los vehículos detectados.

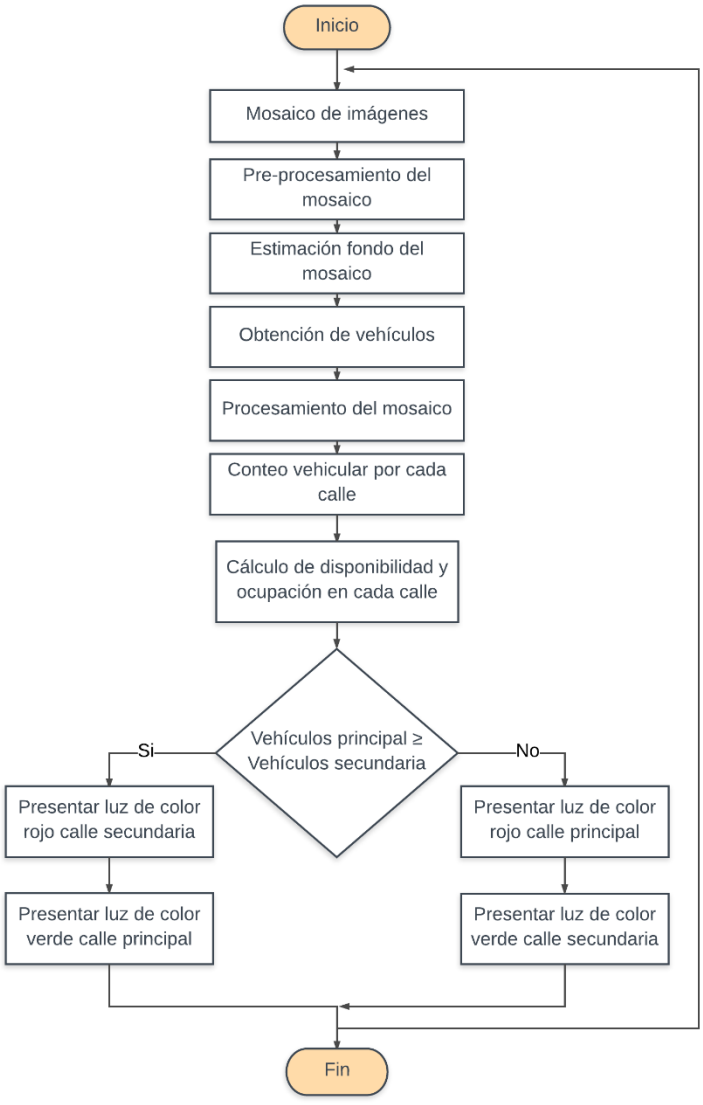


Figura 35. Proceso de semaforización en cada intersección de calles del casco urbano.
Fuente: Imagen propia del autor.
Elaboración: Autor

4.3.5. Cálculo y presentación del porcentaje de congestión en cada calle.

Para conocer el porcentaje de congestión, primero se calculó el área de cada calle y de cada vehículo. Para ello, el problema se lo tomó como si se tratase de hallar el área de un polígono, ya que se conoce las coordenadas de cada uno de sus vértices. Un método que ayuda a calcular el área es el determinante de las coordenadas del polígono, en este caso, los vértices de la ROI, utilizando la Ecuación 19. Para esto se creó la función *area_calc* que permitió calcular la determinante de manera más rápida en MATLAB. De esta manera, conociendo el área de los vehículos y el área de la calle, se calculó la probabilidad de ocupación de todos los vehículos que circulan a través de cada calle con la ayuda de la Ecuación 18.

$$Area\ calle = \frac{1}{2}\begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \\ x_3 & y_3 \\ x_4 & y_4 \\ x_1 & y_1 \end{vmatrix} \tag{18}$$

$$Porc.\ ocupación = \frac{Área\ total\ vehículos}{Área\ de\ la\ calle} \times 100 \tag{19}$$

Una vez realizado el cálculo, la aplicación carga el mapa seleccionado y se presenta las flechas o el símbolo de PARE en cada semáforo según la disponibilidad que se tenga en cada calle, presentando un color distinto que va a depender del siguiente porcentaje: para una disponibilidad del 0 % al 20 %, se presenta un símbolo de color rojo; del 20 % al 40 % una flecha se indica con un color amarillo, mientras que de 40 % hasta 100 % se presenta una flecha con color verde. En la Figura 36 se presenta este proceso mediante un diagrama de flujo.

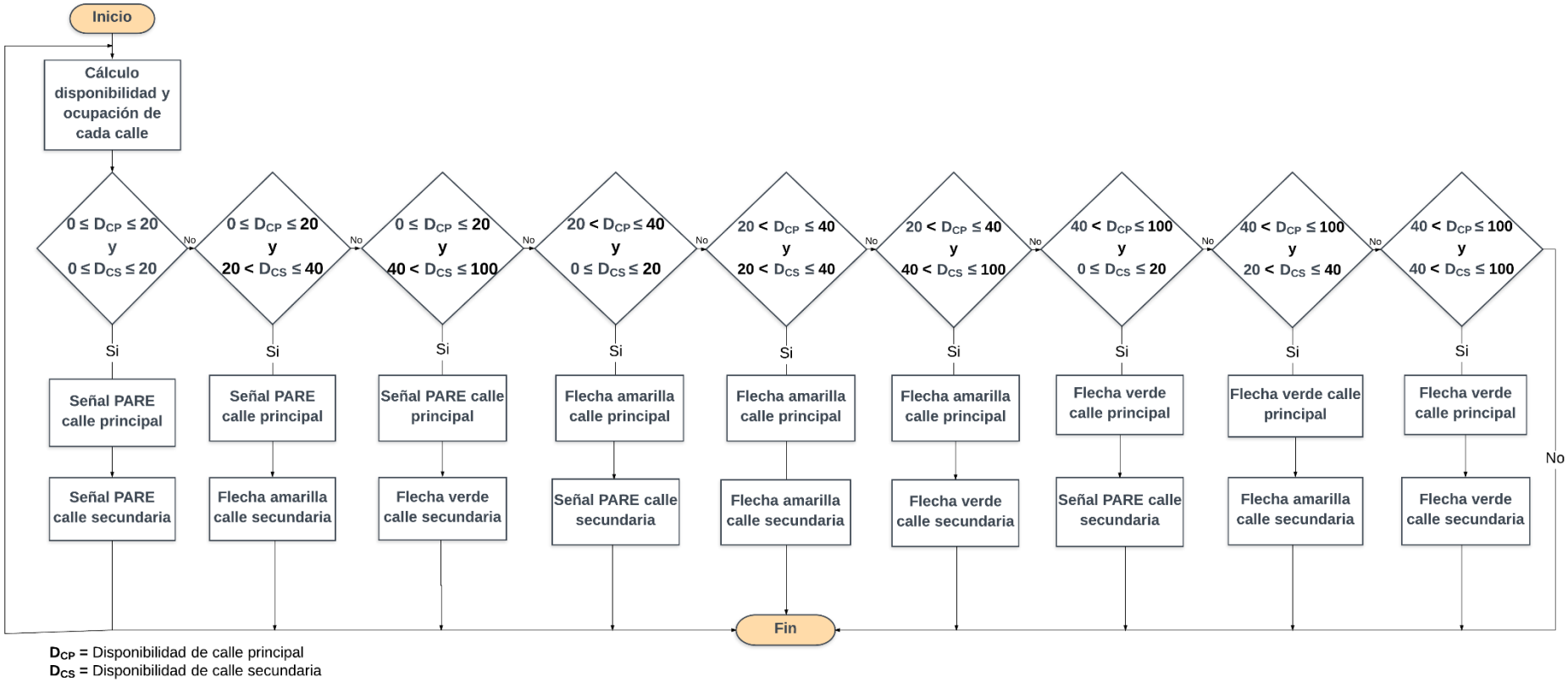


Figura 36. Proceso de presentación del porcentaje de congestión en cada calle.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Cabe recalcar que las flechas se indican según la dirección de la vía y cada una muestra el porcentaje de descongestión como se presenta en la Figura 37.

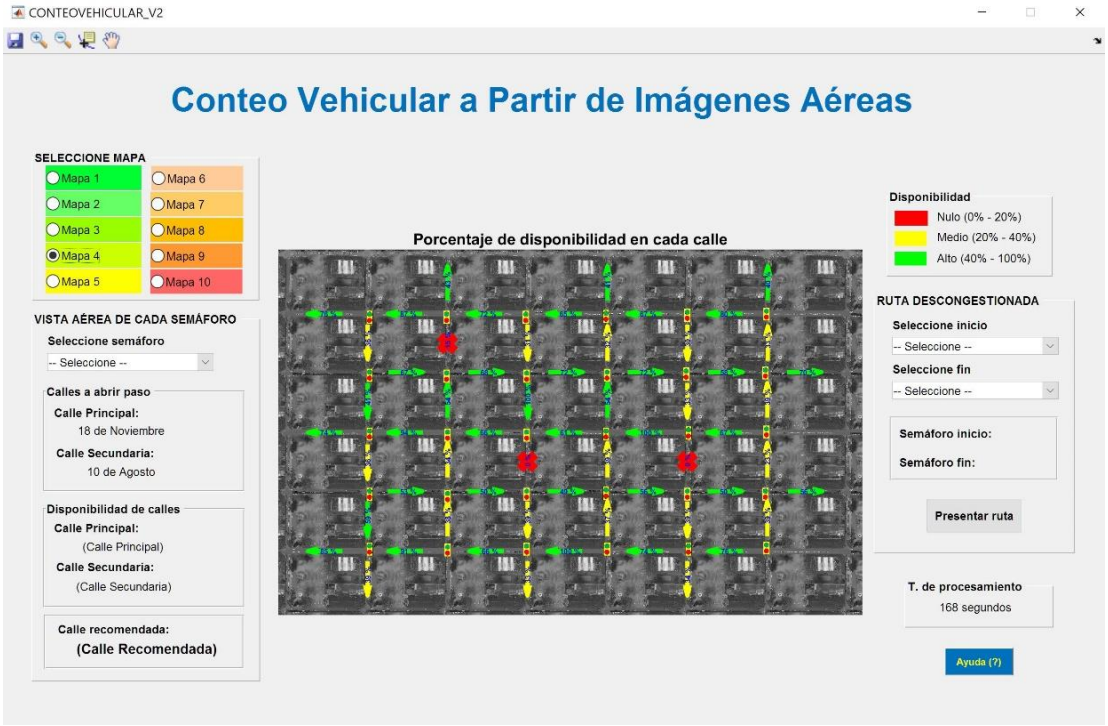


Figura 37. Presentación del porcentaje de disponibilidad en cada calle.
Fuente: Imagen propia del autor.
Elaboración: Autor.

4.3.6. Acercamiento en cada intersersección.

Otra opción que presenta la aplicación es que el usuario tiene la posibilidad de escoger un semáforo para realizar un acercamiento sobre la intersección de las calles que cubre el mismo. Realizado esto, el programa automáticamente presenta los vehículos detectados y para ello se dibuja las cajas de frontera sobre los vehículos detectados en cada calle, recurriéndose al código realizado por Barragán (Barragán, 2015).

Si se detecta una mayor cantidad de vehículos en la calle principal, la caja de frontera presenta un color verde y los vehículos de la calle secundaria se encierran en un recuadro de colo rojo, caso contrario, los vehículos de la calle secundaria poseerán un recuadro verde, mientras que los vehículos de la calle principal uno de color rojo. En caso de que el número de vehículos en cada calle sea el mismo, se da preferencia a los vehículos de la calle principal.

Seguidamente, el algoritmo automáticamente toma una decisión en función del porcentaje de disponibilidad de las calles siguientes al semáforo y recomienda al conductor la ruta con menor congestión mediante un aviso con el nombre de la calle a seguir como también la presentación del porcentaje de disponibilidad en cada flecha o símbolo de PARE (Figura 38).

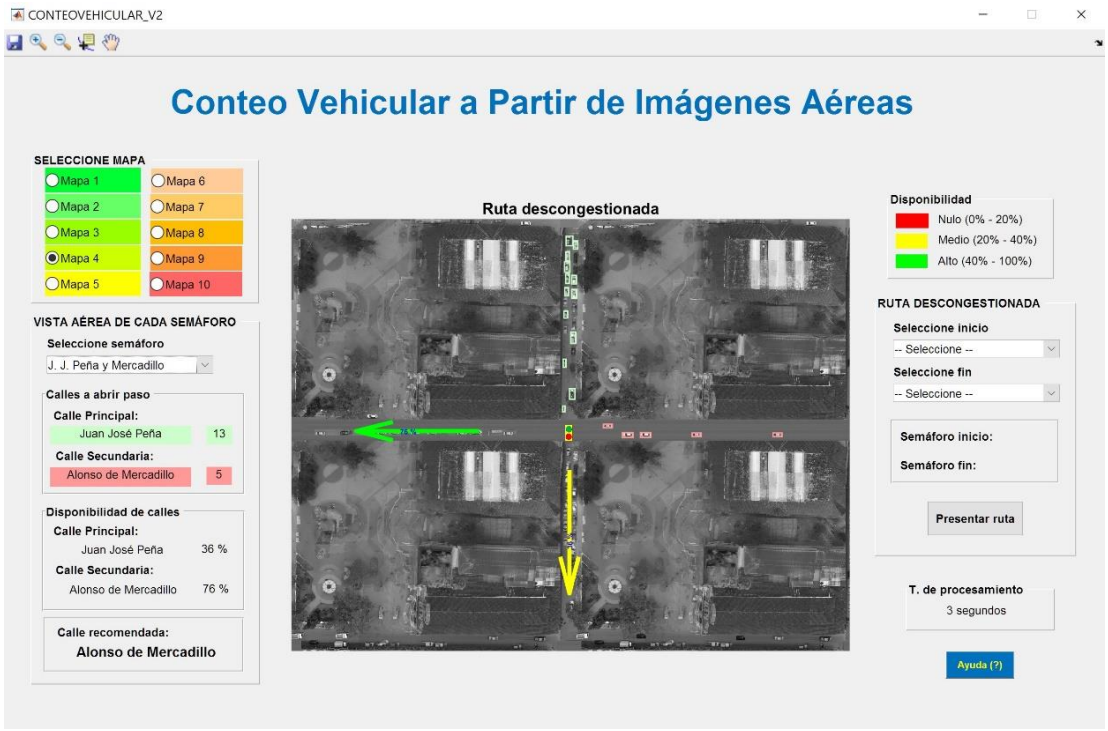


Figura 38. Vista aérea de cada intersección de calles.
Fuente: Imagen propia del autor.
Elaboración: Autor.

4.3.7. Búsqueda de la ruta con menor congestión.

La tercera opción que brinda la aplicación es presentar la ruta con menor congestión en pantalla. Esto mediante la utilización de grafos dirigidos que simulan cada semáforo como vértice del grafo y las aristas representan el sentido de las calles que conectan cada nodo, donde el porcentaje de ocupación encontrado anteriormente en el conteo vehicular se colocó como el peso que contiene cada arista para que los algoritmos recorran el grafo en búsqueda del camino con menor congestión dentro del área que cubre el centro histórico.

Para observar la ruta con menor congestión en pantalla, el usuario tiene la posibilidad de escoger un semáforo inicial y un semáforo de destino al que desee movilizarse. Para ello la aplicación utiliza cuatro métodos a la vez para la presentación de la ruta: Dijkstra, Bellman-Ford, Breadth-First Search y el método acíclico, los cuales son algoritmos que sirven para la búsqueda del camino más corto en un grafo, pero pueden ser adaptados al presente trabajo. En el caso de existir más de una ruta con baja congestión, se presenta en pantalla un mensaje indicando el número de rutas alternativas que existe (Figura 39) y cada ruta poseerá un color distinto.

Cada semáforo posee un número que permita identificarlo, en este caso el semáforo ubicado en la intersección de las calles Juan José Peña y Alonso de Mercadillo es el primer nodo identificado con el número “1” mientras que el último semáforo identificado con el número “30” es el que está ubicado en la intersección de la calle 18 de Noviembre y 10 de Agosto.

Las aristas poseen un peso, donde cada peso es el porcentaje de ocupación (de color negro) que ha sido calculado en cada calle tal como se presenta en la Figura 40. Estos porcentajes ayudan a que los algoritmos encuentren la ruta más descongestionada dentro del casco urbano de la ciudad.

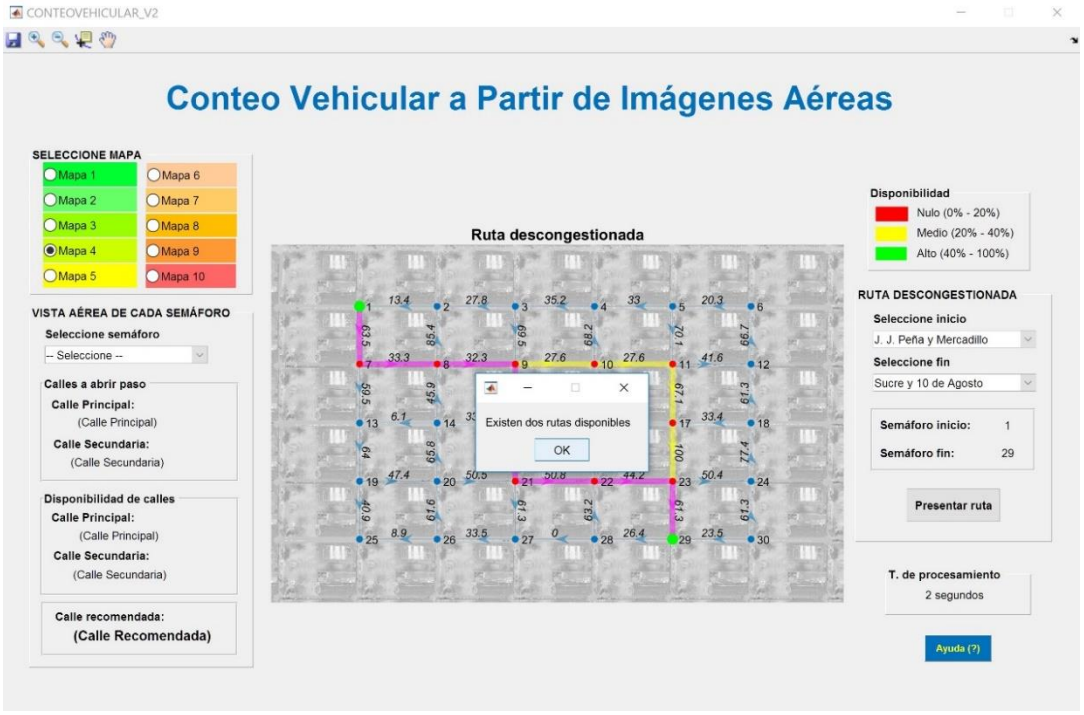


Figura 39. Presentación del mensaje de rutas disponibles en la GUI de MATLAB.
Fuente: Imagen propia del autor.
Elaboración: Autor.

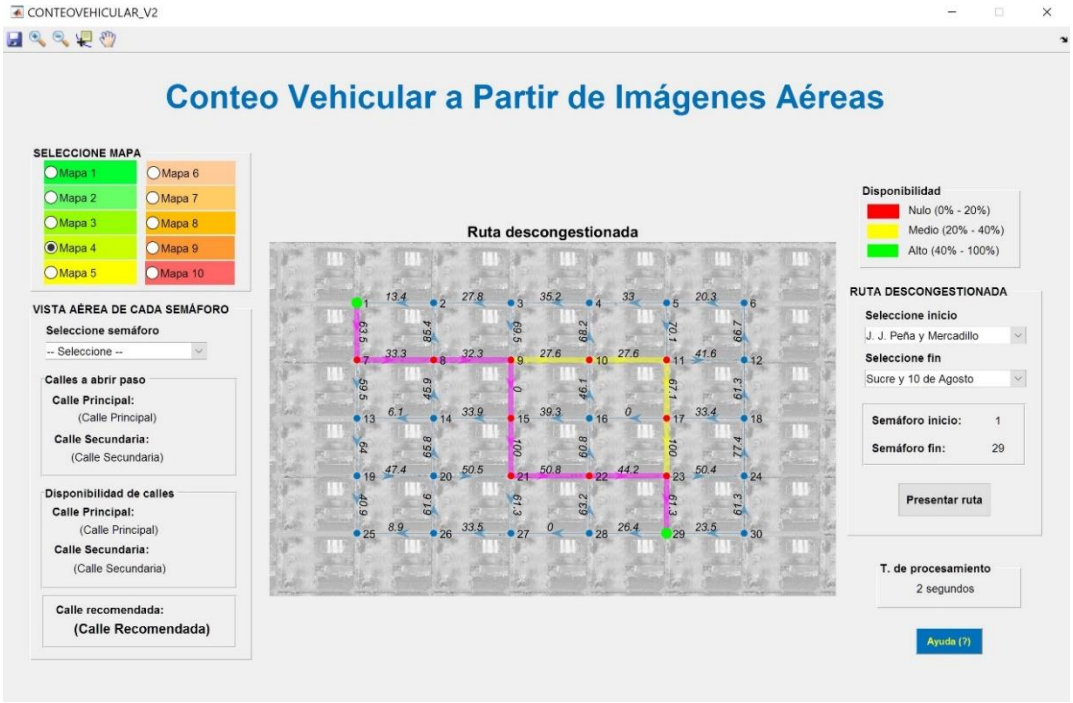


Figura 40. Presentación de rutas con menor congestión cargado en la GUI de MATLAB.
Fuente: Imagen propia del autor.
Elaboración: Autor.

CAPITULO V.
EXPERIMENTOS Y RESULTADOS

5.1. Descripción de los experimentos realizados

En este capítulo se presentan los resultados obtenidos en la creación de las vistas panorámicas tipo mosaico, los resultados de la contabilización de vehículos en cada una de las calles del casco urbano de la ciudad de Loja, la presentación del porcentaje de disponibilidad en cada calle, la búsqueda de la ruta con menor congestión y el tiempo que toma cada proceso mencionado anteriormente.

5.1.1. Resultados de la creación de la vista panorámica tipo mosaico.

El primer experimento realizado fue la creación de vistas panorámicas tipo mosaico, para lo cual se realizó pruebas en los tres canales de color de la imagen aérea y a escala de grises para poder determinar en cual de estos cuatro casos se encontraba el mayor número de correspondencias. De la misma forma se realizaron pruebas con los distintos tipos de interpolaciones (bicúbica, bilineal y vecinos más cercanos) y así observar en cual se obtenía mejores resultados.

Para obtener el mosaico se partió de cuatro imágenes aéreas captadas a través de la cámara colocada en el vehículo aéreo no tripulado. Como primer paso para la elaboración del mosaico se procedió a detectar los puntos característicos de la primera y segunda imagen original. En el segundo paso se calculó sus correspondencias (Figura 41) y de esta manera se logró fusionar estas dos imágenes en una sola (Figura 42).

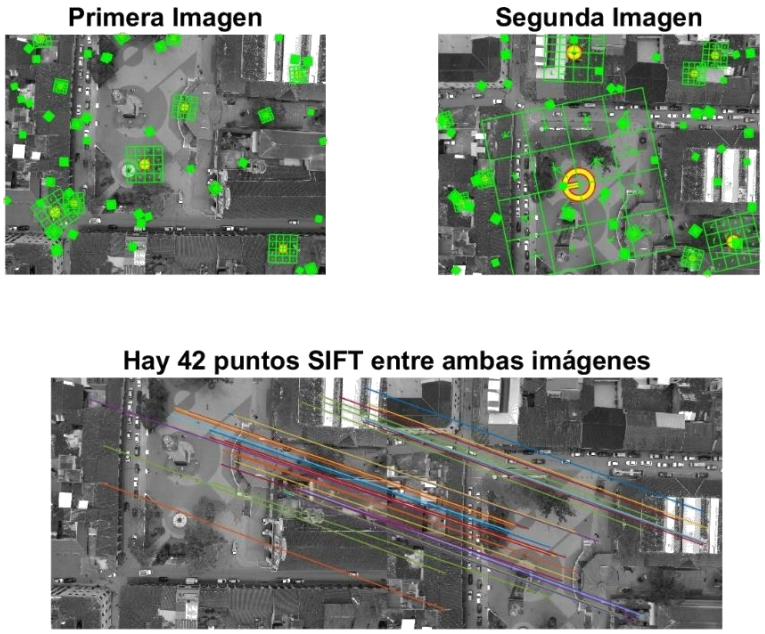


Figura 41. a) Puntos característicos detectados en la primera y segunda imagen. b) Correspondencias entre ambas imágenes.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Figura 42. Primera fusión de imágenes para formar el mosaico.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Al igual que el procedimiento para obtener la primera fusión de imágenes, el tercer paso consiste en la detección de puntos característicos pero con la diferencia que en este paso se trabajó con la primera imagen fusionada y la tercera imagen (Figura 42). En el cuarto paso se cálculo las correspondencias entre estas dos imágenes (Figura 43). La segunda fusión obtenida se presenta en la Figura 44.

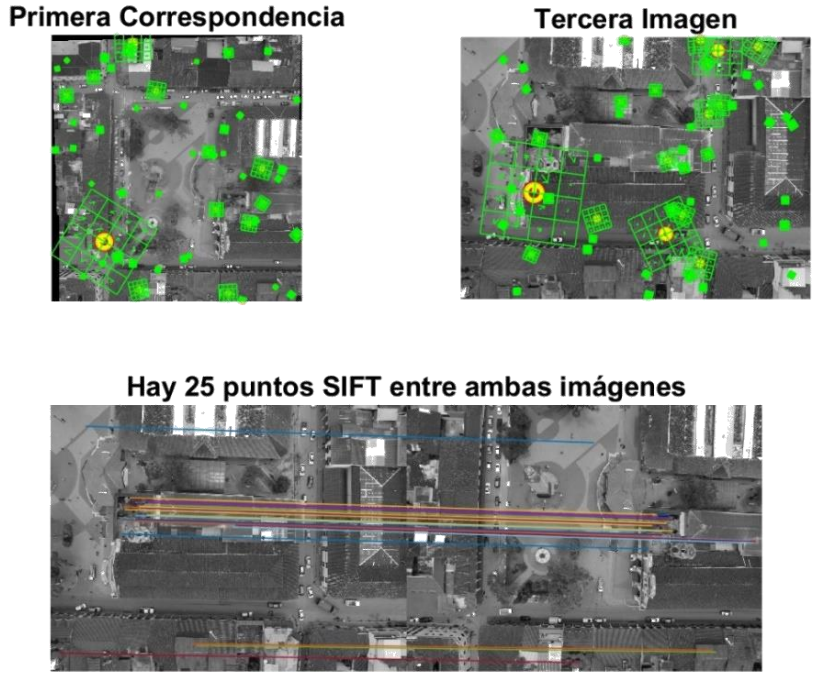


Figura 43. Puntos característicos detectados en la primera fusión y tercera imagen.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Figura 44. Segunda fusión de imágenes para formar el mosaico.
Fuente: Imagen propia del autor.
Elaboración: Autor.

Para finalizar, el mosaico resultante que se presenta en la Figura 46 se obtuvo con el mismo procedimiento anterior (Figura 45), pero en este caso se trabajó con la segunda imagen fusionada (Figura 44) y la cuarta imagen aérea captada.

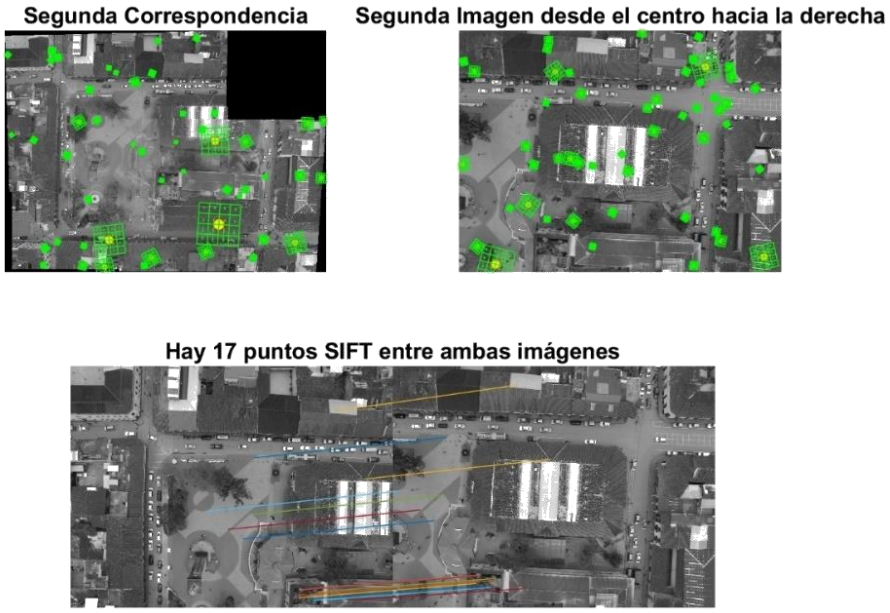


Figura 45. Puntos característicos detectados en la segunda fusión y tercera imagen.
Fuente: Imagen propia del autor
Elaboración: Autor.

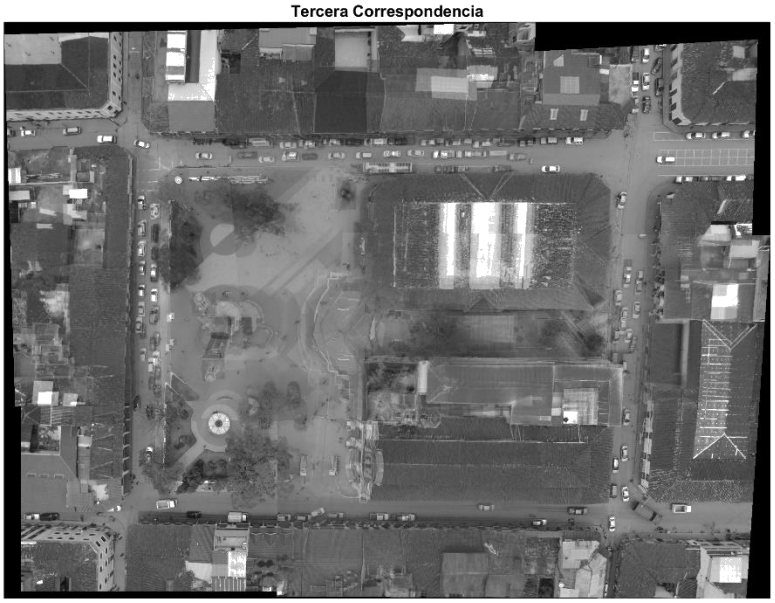


Figura 46. Mosaico final resultante.
Fuente: Imagen propia del autor.
Elaboración: Autor.

5.1.2. Resultados de la contabilización de vehículos por cada calle.

Otro de los experimentos fue la implementación y realización de las pruebas del algoritmo en diez escenarios con distinta cantidad de vehículos por cada mapa (Figura 47 hasta Figura 56), donde el primer mapa contiene un escenario sin vehículos en todas las calles, mientras que el décimo mapa presenta un escenario con un gran número de vehículos en cada calle. El número de vehículos se incrementó de manera proporcional hasta llegar al último escenario con un número elevado de vehículos, resultando en una gran congestión vehicular.

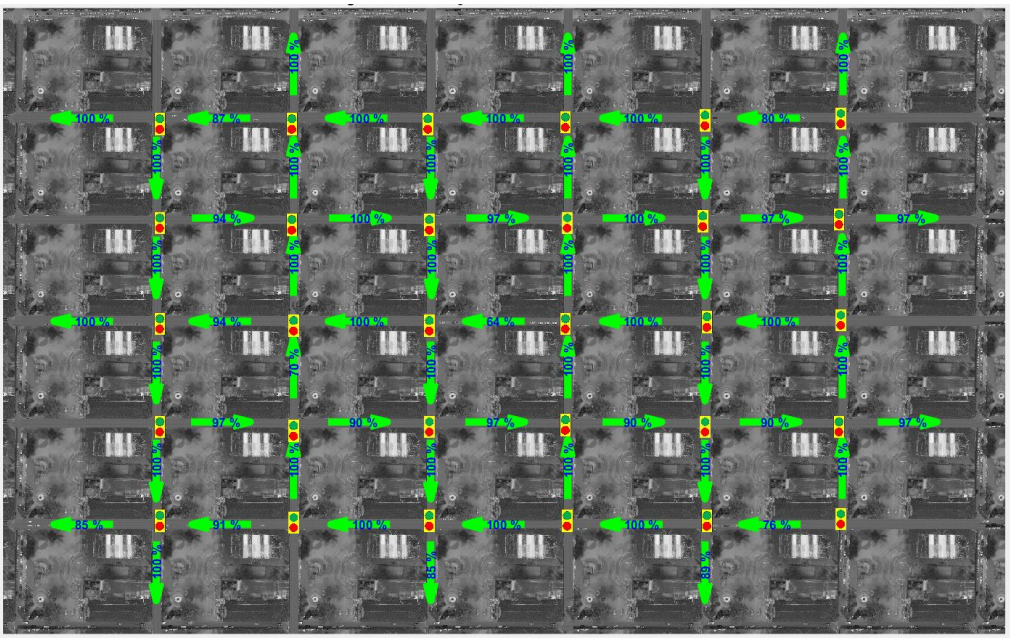


Figura 47. Disponibilidad de calles en el primer escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

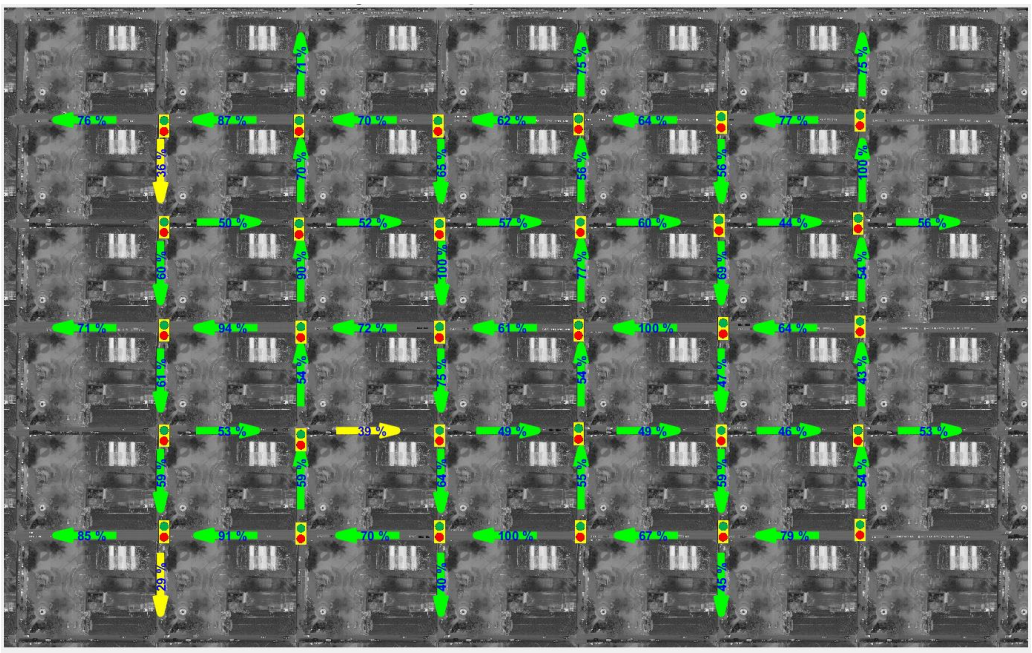


Figura 48. Disponibilidad de calles en el segundo escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

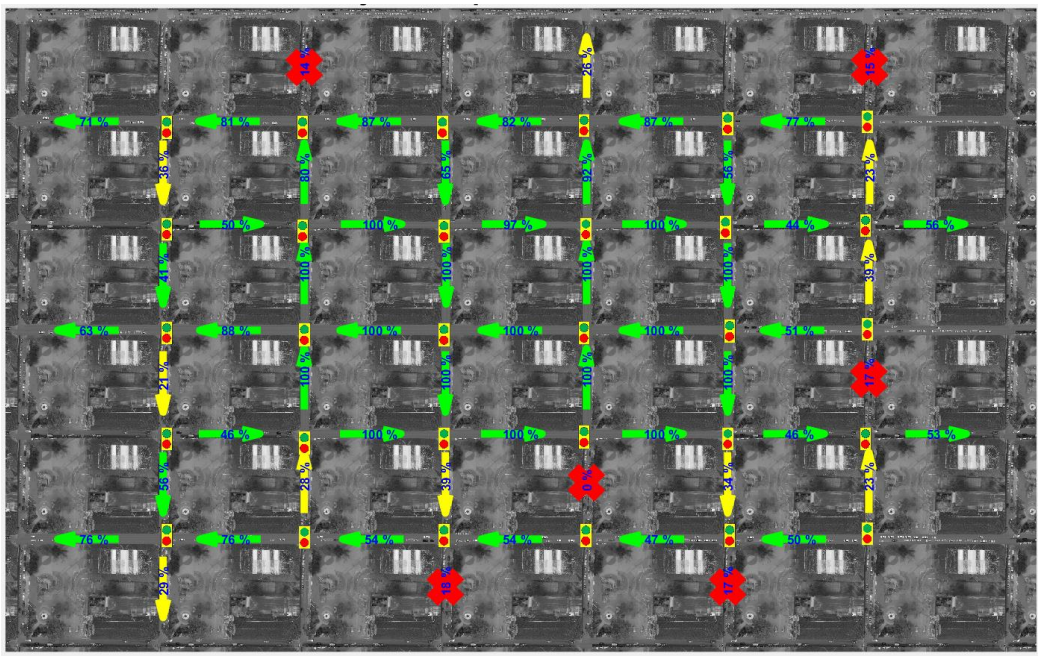


Figura 49. Disponibilidad de calles en el tercer escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

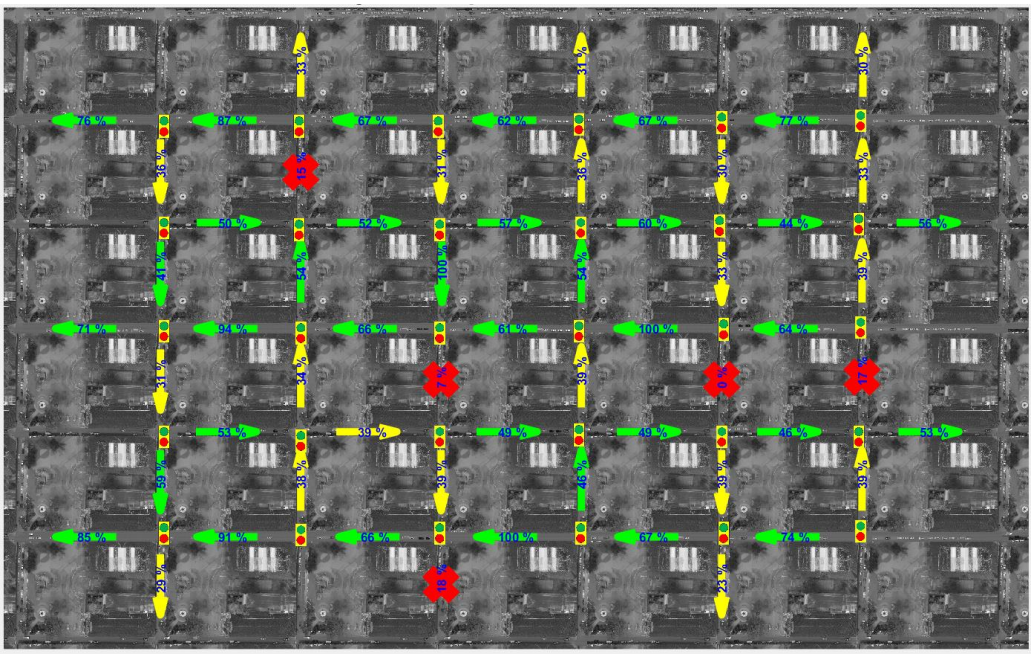


Figura 50. Disponibilidad de calles en el cuarto escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

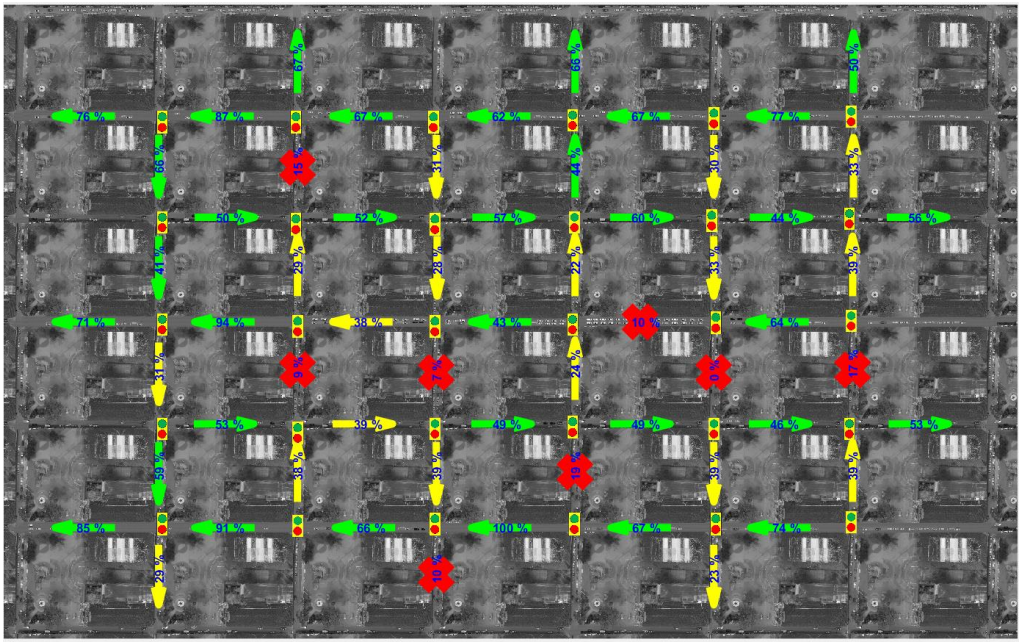


Figura 51. Disponibilidad de calles en el quinto escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

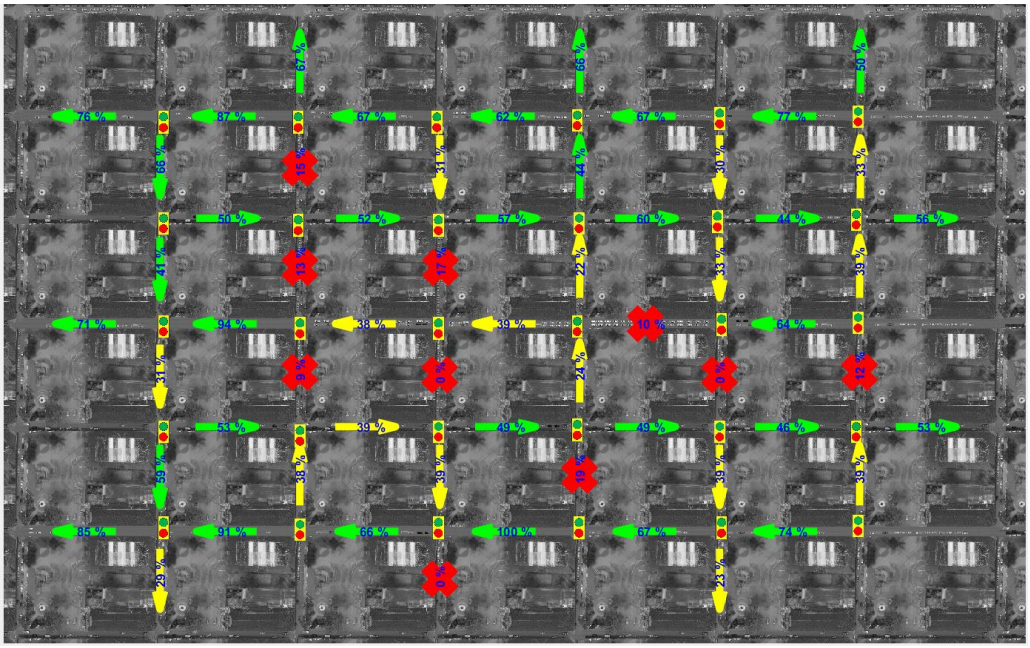


Figura 52. Disponibilidad de calles en el sexto escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

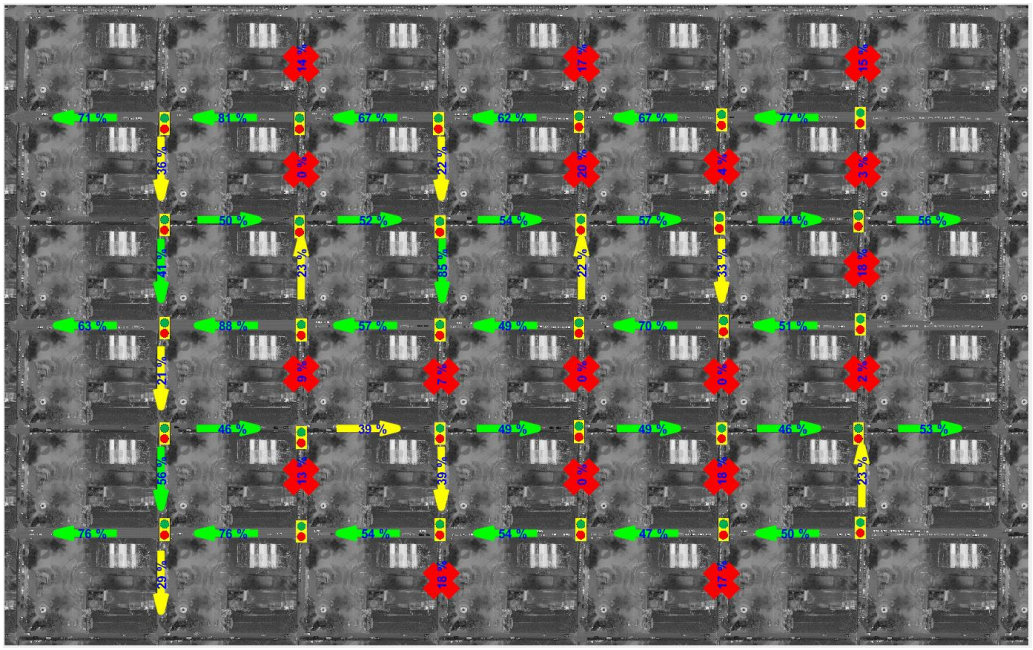


Figura 53. Disponibilidad de calles en el séptimo escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

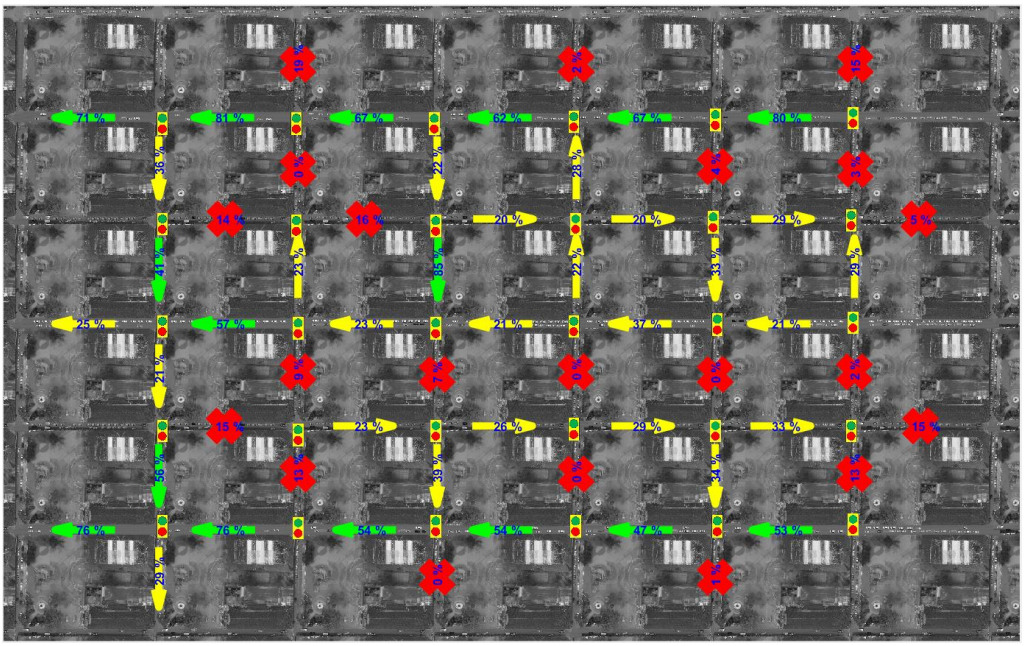


Figura 54. Disponibilidad de calles en el octavo escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

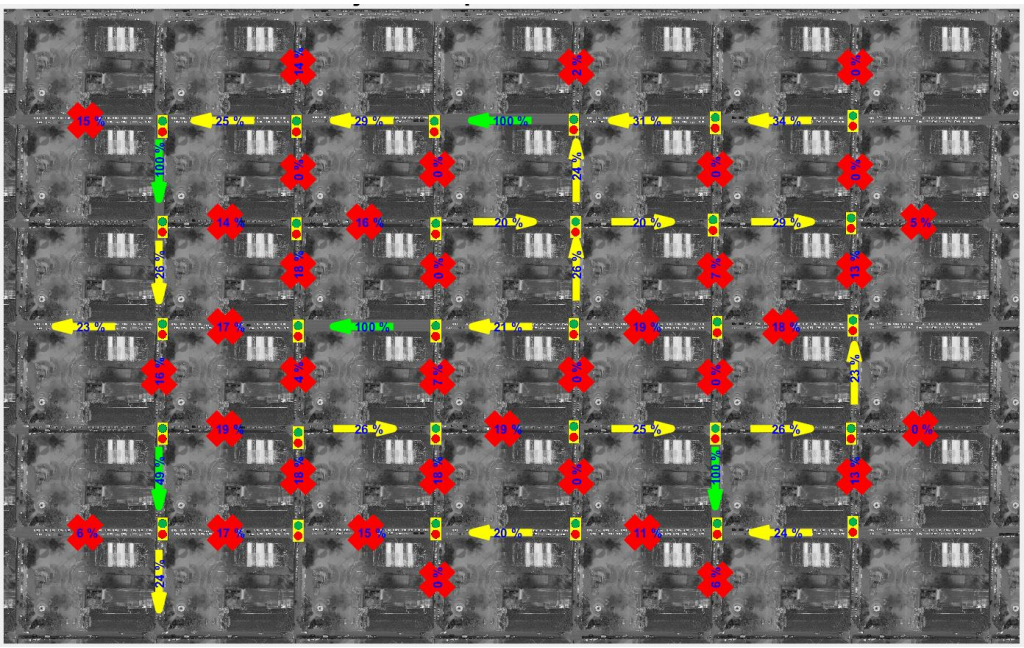


Figura 55. Disponibilidad de calles en el noveno escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

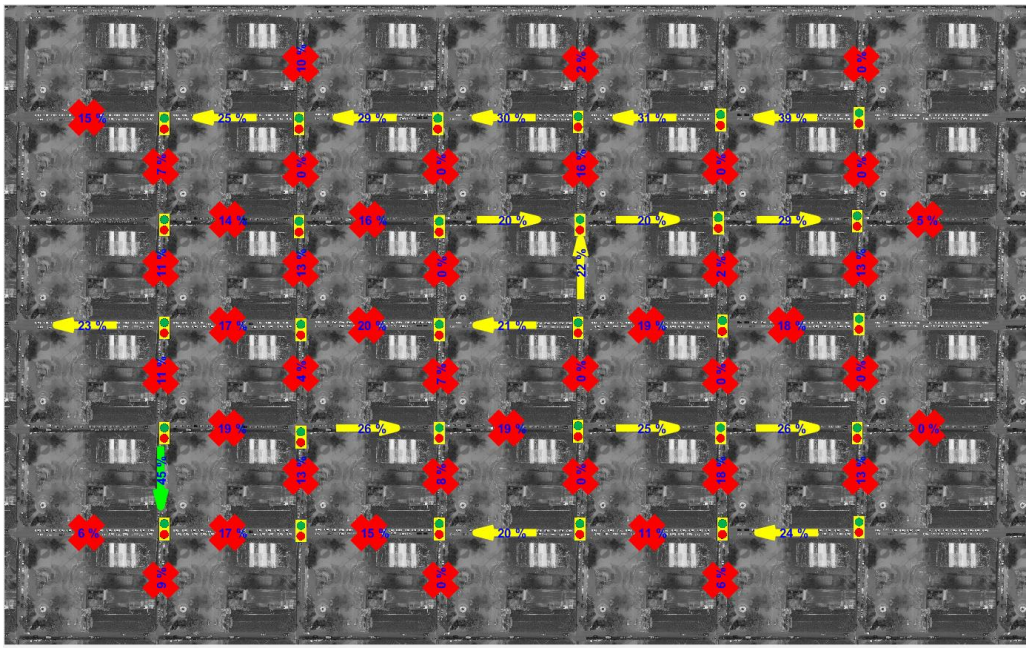


Figura 56. Disponibilidad de calles en el décimo escenario.
Fuente: Imagen propia del autor.
Elaboración: Autor.

5.2. Tablas y gráficas de resultados

Para observar si el tiempo de procesamiento depende del procesador del ordenador con el que se trabajó, se realizó pruebas tanto con la CPU (Unidad Central de Proceso) como con la GPU (Unidad de Procesamiento de Gráficos) para obtener resultados en cuanto a velocidad de procesamiento.

Los resultados obtenidos en este trabajo han sido calculados con la ayuda del programa MATLAB, realizando pruebas de comparación entre dos ordenadores con las siguientes características: el primero con procesador Intel Core i5-4210U a 2.4 GHz y 12 GB de memoria RAM y el segundo con un procesador Intel Core i7-6700HQ a 2.6 GHz con 8 GB de memoria RAM, pero con la diferencia que el segundo ordenador posee una tarjeta de video Nvidia incorporada de 4 GB dedicadas para el cálculo con la GPU.

5.2.1. Puntos característicos SIFT detectados.

Mediante el algoritmo detector y descriptor SIFT fueron detectados los puntos característicos en cada una de las imágenes, así como también los puntos característicos de cada fusión de imágenes hasta la obtención del mosaico completo. Estos resultados fueron calculados en distintas resoluciones de las imágenes y en cada canal de color, los mismos que son presentados en la Tabla 9.

Tabla 9. Puntos SIFT detectados para cada par de imágenes

Canal	Resolución de la imagen (píxeles)	Primer Mosaico		Segundo mosaico		Tercer mosaico	
		Primera imagen	Segunda imagen	Primera fusión	Tercera imagen	Segunda fusión	Cuarta imagen
Rojo	1200x900	4440	4493	5961	4153	6076	4445
Verde		4213	4141	5949	3937	7880	4035
Azul		4066	3954	6225	3867	8166	3984
Gris		4101	4105	6293	3874	8120	3978
Rojo	2000x1500	12254	12472	18197	11629	24828	11848
Verde		11553	11570	15967	10861	20354	11000
Azul		10891	10812	16242	10549	18081	10663
Gris		11374	11356	16488	10664	21359	10759
Rojo	2800x2100	23728	23508	33807	21471	45591	21617
Verde		22456	22259	31661	20629	38593	20416
Azul		21460	20960	30761	19958	34561	19958
Gris		22216	22079	31122	20321	38117	20252
Rojo	3600x2700	42765	41550	59591	38455	87608	38048
Verde		41685	39945	59479	37369	71593	36978
Azul		40515	38462	57583	36928	63576	36547
Gris		41577	39764	65275	37213	77092	36681

Fuente: Propia del autor.
Elaboración: Autor.

5.2.2. Número de correspondencias calculadas.

Los resultados presentados en la Tabla 10, pertenecen a las correspondencias calculadas luego de obtener los puntos característicos SIFT en las distintas pruebas realizadas sobre cada canal de color (Tabla 9).

Tabla 10. Número de correspondencias calculadas.

Canal	Resolución de la imagen (píxeles)	Correspondencias		
		Primera fusión	Segunda fusión	Tercera fusión
Rojo	1200x900	39	17	14
Verde		42	25	17
Azul		26	15	9
Gris		64	23	25
Rojo	2000x1500	104	60	46
Verde		66	37	29
Azul		63	41	24
Gris		131	68	59
Rojo	2800x2100	81	60	53
Verde		142	92	79
Azul		81	60	49
Gris		203	116	84
Rojo	3600x2700	83	56	65
Verde		150	105	90
Azul		87	79	58
Gris		209	126	96

Fuente: Propia del autor.
Elaboración: Autor.

Entre la Figura 57, Figura 58, Figura 59 y Figura 60 se aprecia que el mayor número de correspondencias encontradas estuvieron presentes en las imágenes a escala de grises en los distintos tamaños en los que se ha reducido la imagen para la realización de las pruebas. También se pudo observar que a mayor tamaño de la imagen, existe mayor número de correspondencias en las distintas pruebas realizadas sobre cada canal de color.

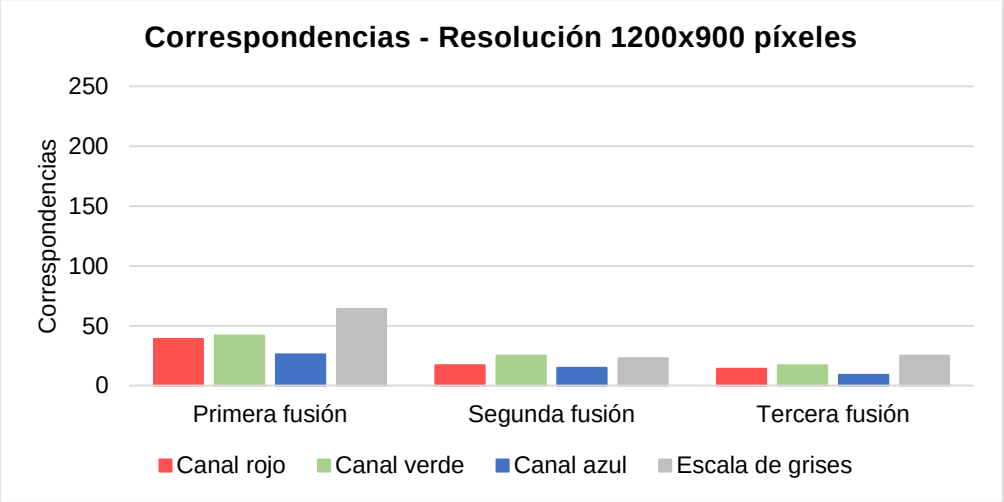


Figura 57. Número de correspondencias por canal de color. Imagen reducida al 30 %.
Fuente: Propia del autor.
Elaboración: Autor.

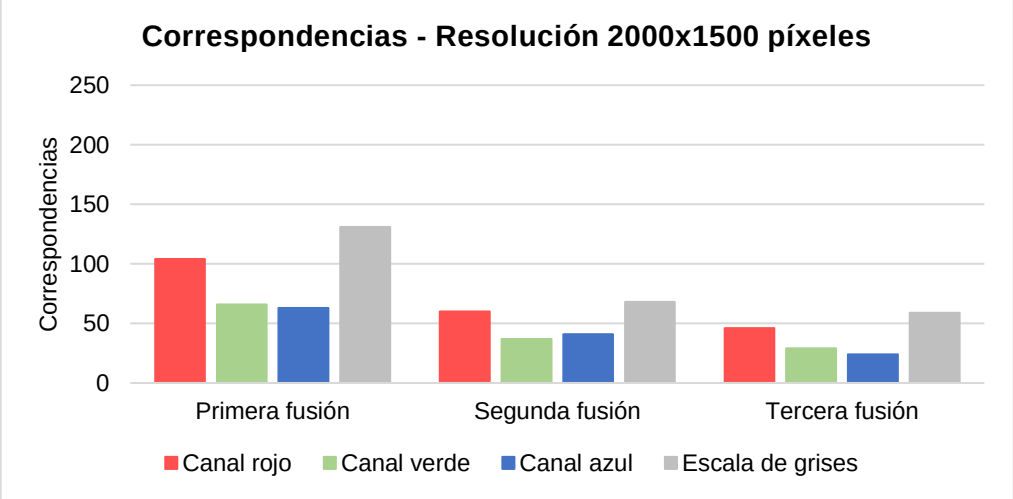


Figura 58. Número de correspondencias por canal de color. Imagen reducida al 50 %.
Fuente: Propia del autor.
Elaboración: Autor.

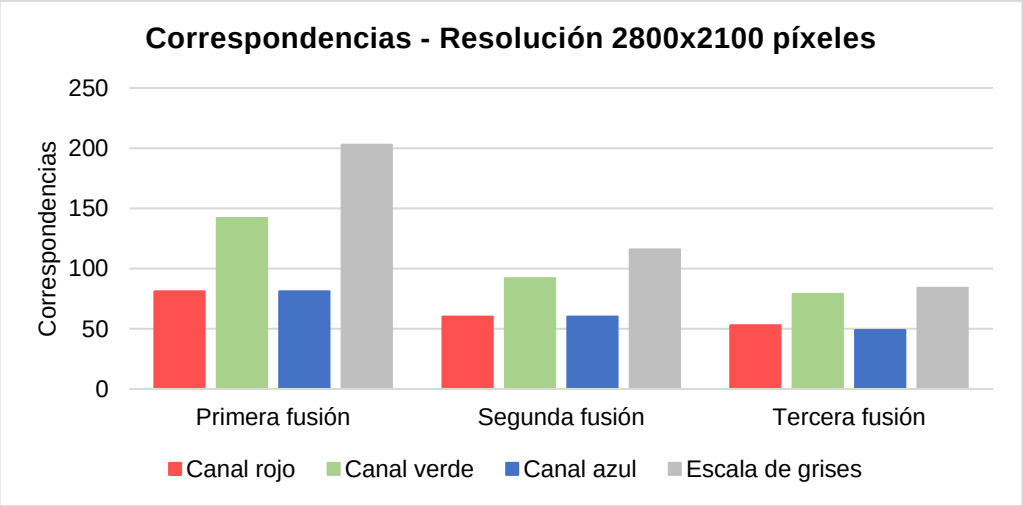


Figura 59. Número de correspondencias por canal de color. Imagen reducida al 70 %.
Fuente: Propia del autor.
Elaboración: Autor.

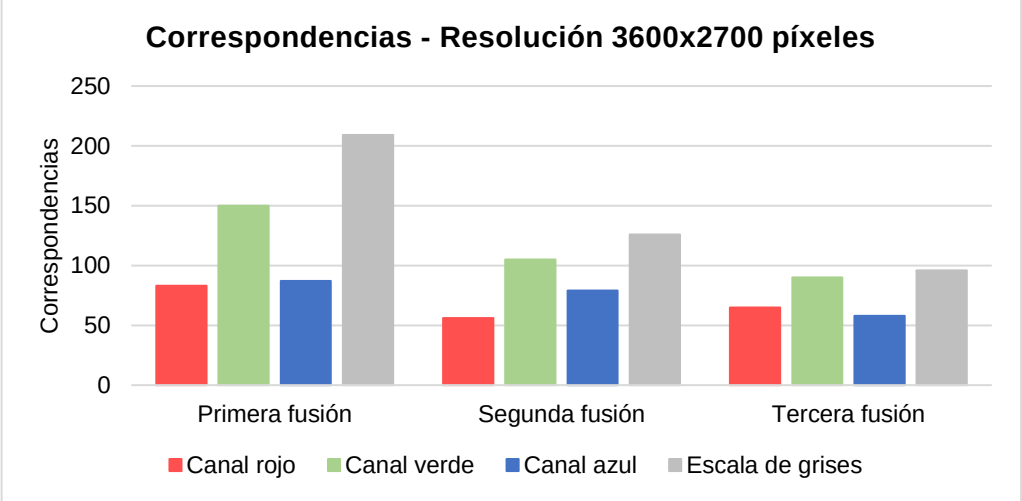


Figura 60. Número de correspondencias por canal de color. Imagen reducida al 90 %.
Fuente: Propia del autor
Elaboración: Autor.

5.2.3. Resultados con los tipos de interpolación.

En base a las iteraciones desarrolladas, se pudo calcular el error de diferencia mínimo y máximo entre los puntos SIFT detectados en las distintas resoluciones de la imagen luego de haber calculado su homografía utilizando el modelo RANSAC.

Tabla 11. Resultados de las pruebas realizadas con interpolación bicúbica.

Canal	Resolución de la imagen (píxeles)	Interpolación	Error en píxeles			Core i7	Core i5
			Primera fusión	Segunda fusión	Tercera fusión	Tiempo (s)	Tiempo (s)
Rojo	1200x900	Bicúbica	18,3	0,0	0,9	27,5	--
Verde			26,3	0,6	35,8	19,1	26,1
Azul			20,5	1,1	2,0	20,1	24,4
Gris			23,3	2,7	6,3	19,8	23,4
Rojo	2000x1500		49,0	50,2	60,9	78,6	93,5
Verde			50,8	13,9	386,8	76,1	86,3
Azul			52,9	1,1	4,1	61,5	89,5
Gris			60,3	10,2	74,7	68,0	84,8
Rojo	2800x2100		65,3	1,0	1,6	238,8	258,7
Verde			82,6	75,6	609,5	215,9	232,8
Azul			65,5	11,4	89,6	208,3	219,7
Gris			113,5	45,8	91,9	206,6	240,9
Rojo	3600x2700		88,4	1,0	11,9	529,5	695,7
Verde			102,9	35,4	469,8	539,8	656,9
Azul			130,4	23,2	22,2	498,9	622,1
Gris			48,4	15,3	69,3	491,8	657,3

Fuente: Propia del autor.
Elaboración: Autor.

Tabla 12. Resultados de las pruebas realizadas con interpolación bilineal.

Canal	Resolución de la imagen (píxeles)	Interpolación	Error en píxeles			Core i7	Core i5
			Primer Mosaico	Segundo mosaico	Tercer mosaico	Tiempo (s)	Tiempo (s)
Rojo	1200x900	Bilineal	19,2	48,0	No correspondencia	--	--
Verde			28,4	8,8	0,0	22,0	24,0
Azul			21,0	17,7	16,6	30,8	26,9
Gris			19,5	0,0	20,3	24,0	25,1
Rojo	2000x1500		45,5	7,1	46,3	103,0	94,5
Verde			45,6	11,9	0,8	75,8	88,8
Azul			44,1	0,3	3,4	97,8	108,1
Gris			60,9	47,0	3,0	79,7	89,6
Rojo	2800x2100		72,3	57,5	53,6	229,2	266,1
Verde			82,6	22,9	36,7	239,2	241,0
Azul			64,4	1,0	22,0	211,8	231.2
Gris			53,0	28,7	0,0	225,0	235,9
Rojo	3600x2700		96,4	2,2	0,0	630,0	--
Verde			102,3	38,2	134,1	531,6	698,3
Azul			87,8	16,7	7,7	497,2	818,8
Gris			146,0	27,8	75,8	531,7	653,5

Fuente: Propia del autor.
Elaboración: Autor.

Tabla 13. Resultados de las pruebas realizadas con interpolación vecino más cercano.

Canal	Resolución de la imagen (píxeles)	Interpolación	Error en píxeles			Core i7	Core i5
			Primer Mosaico	Segundo mosaico	Tercer mosaico	Tiempo (s)	Tiempo (s)
Rojo	1200x900	Vecino más cercano	26,2	0,9	No correspondencias	--	--
Verde			25,8	0,5	36,5	21,5	29,5
Azul			24,8	0,0	6,0	21,2	24,3
Gris			27,8	11,5	1,4	20,3	24,5
Rojo	2000x1500		52,3	12,3	45,0	85,2	96,1
Verde			44,7	9,7	66,5	75,2	87,5
Azul			42,1	10,5	1,9	63,8	83,7
Gris			59,6	204,9	5,4	73,7	86,4
Rojo	2800x2100		69,5	1,9	42,9	231,7	273,2
Verde			78,1	57,5	2,1	196,0	239,1
Azul			68,3	0,7	1,6	176,3	--
Gris			97,1	41,4	15,9	186,5	236,1
Rojo	3600x2700		96,1	9,4	108,0	568,9	694,4
Verde			107,5	89,7	130,1	537,2	672,8
Azul			80,9	12,8	16,6	483,6	630,9
Gris			120,8	67,0	112,7	537,2	671,3

Fuente: Propia del autor.
Elaboración: Autor.

En las gráficas presentadas en la Figura 61, Figura 62 y Figura 63, se puede apreciar una comparación del tiempo que tardó el algoritmo en realizar el mosaico de imágenes, teniendo como resultados que, a menor tamaño de la imagen, menor es el tiempo que le tomó al algoritmo crear el mosaico con las cuatro imágenes aéreas captadas.

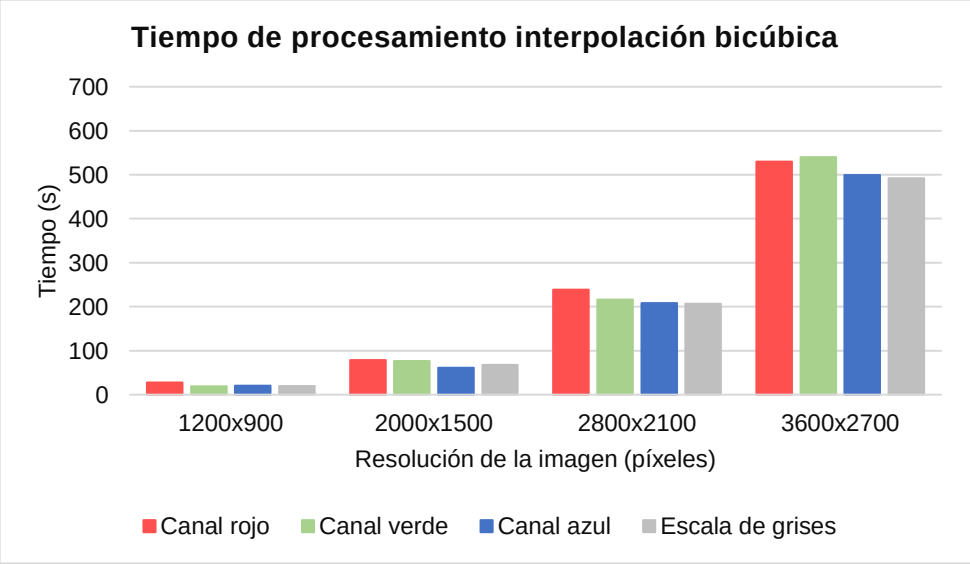


Figura 61. Tiempo de procesamiento para la creación del mosaico con interpolación bicúbica.
Fuente: Propia del autor.
Elaboración: Autor.

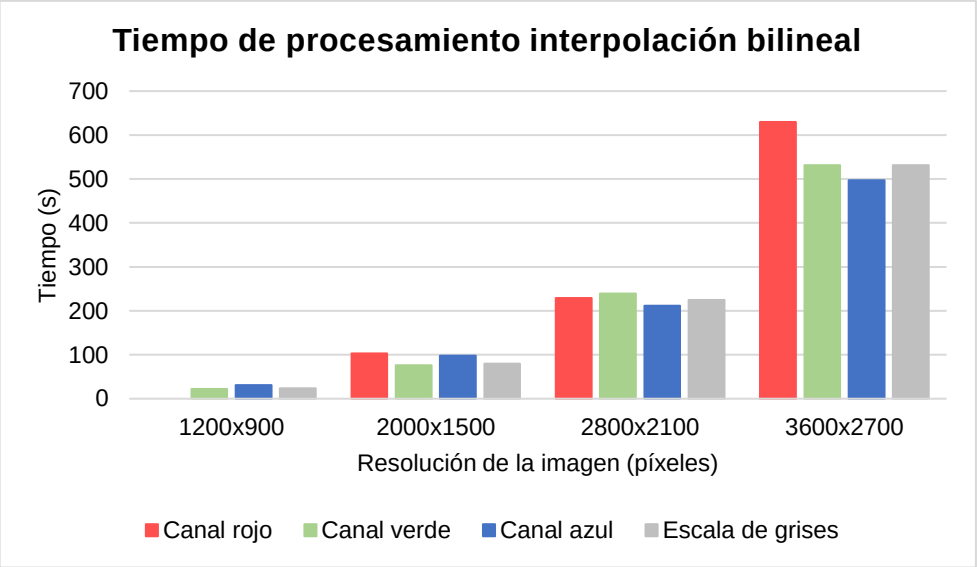


Figura 62. Tiempo de procesamiento para la creación del mosaico con interpolación bilineal.
Fuente: Propia del autor
Elaboración: Autor.

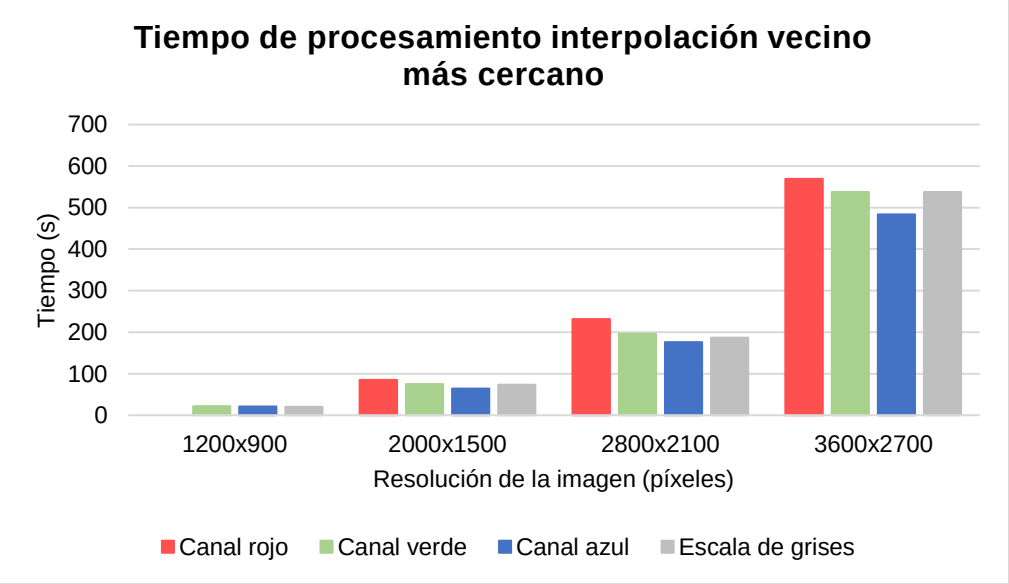


Figura 63. Tiempo de procesamiento para la creación del mosaico con interpolación vecino más cercano
Fuente: Propia del autor
Elaboración: Autor.

De la misma manera se presenta una comparación del promedio del tiempo total requerido para la creación del mosaico de imágenes entre los dos ordenadores con los distintos tipos de interpolaciones: bicúbica (Figura 64), bilineal (Figura 65) y vecinos más cercanos (Figura 66). Se obtuvo como resultado que en la computadora Dell con procesador Core i7 tarda menor tiempo en realizar el proceso en las tres pruebas realizadas.

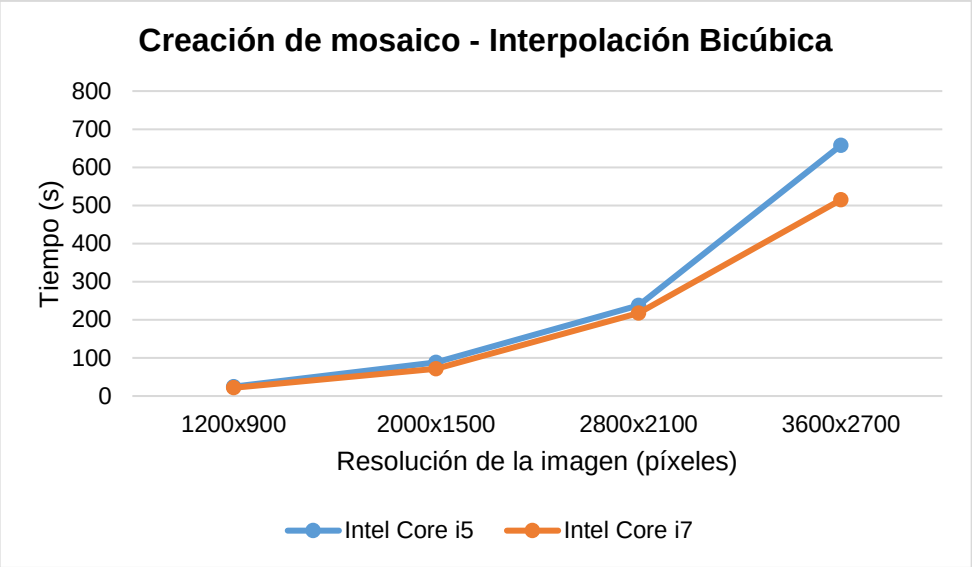


Figura 64. Comparación tiempo de procesamiento entre los dos ordenadores con interpolación bicúbica
Fuente: Propia del autor.
Elaboración: Autor.

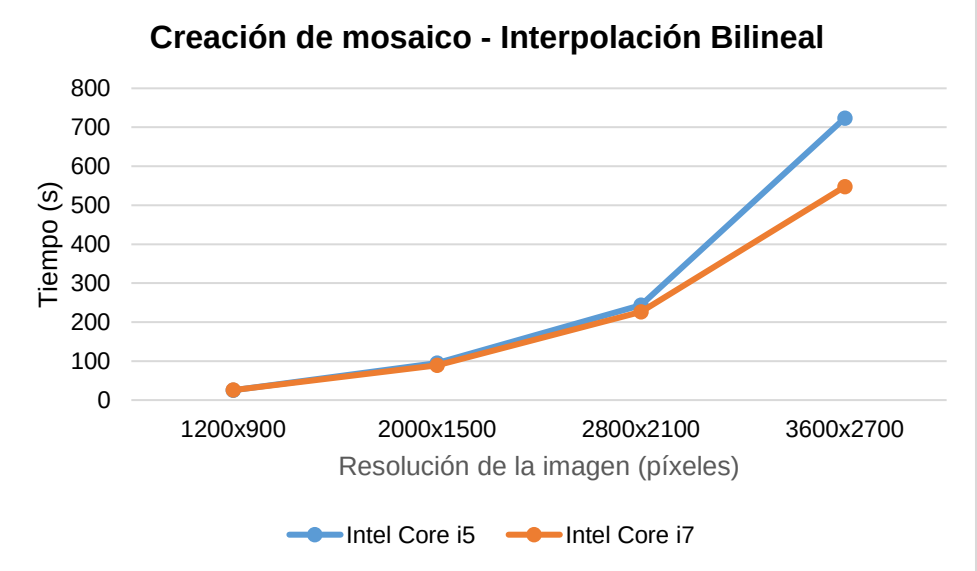


Figura 65. Comparación tiempo de procesamiento entre los dos ordenadores con interpolación bilineal.
Fuente: Propia del autor.
Elaboración: Autor.

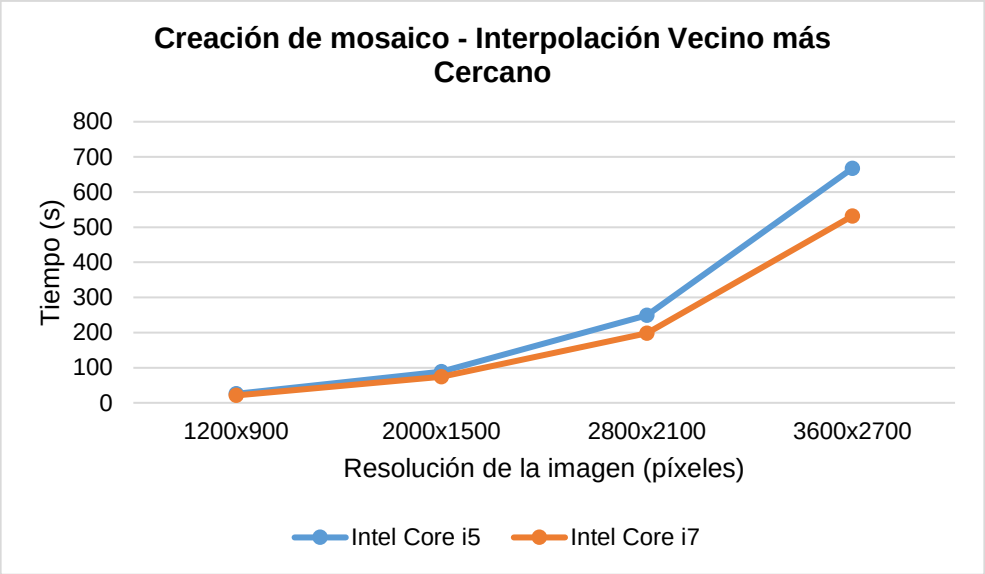


Figura 66. Comparación tiempo de procesamiento entre los dos ordenadores con interpolación del vecino más cercano.
Fuente: Propia del autor.
Elaboración: Autor.

5.2.4. Resultados del conteo vehicular.

Estos resultados se han dividido en dos: tiempo de procesamiento en el conteo vehicular y la precisión con la que el algoritmo detecta los vehículos en cada vía.

En la Tabla 14 se presenta el tiempo de procesamiento que tardó cada ordenador en realizar el conteo de los vehículos, el cálculo de disponibilidad y ocupación en cada calle hasta la presentación en pantalla de la disponibilidad que existió en cada una. Se obtuvo tres resultados por cada escenario debido a que en el ordenador con procesador Core i7 se pudo realizar el cálculo con la CPU y también con la GPU.

Tabla 14. Tiempo de procesamiento para el conteo vehicular

Escenario	Tiempo (s)		
	Core i5 - 4210U (CPU)	Core i7 - 6700HQ (CPU)	Core i7 - 6700HQ (GPU)
Escenario 1	229	151	132
Escenario 2	230	152	149
Escenario 3	207	155	149
Escenario 4	191	154	149
Escenario 5	194	156	156
Escenario 6	195	161	149
Escenario 7	195	152	151
Escenario 8	195	152	150
Escenario 9	195	152	149
Escenario 10	197	159	151

Fuente: Propia del autor.
Elaboración: Autor.

Como se puede apreciar en la Figura 67, trabajar con la GPU en el ordenador con procesador Core i7 demostró ser 1,04 veces más rápido que trabajar con la CPU en el mismo ordenador y fue 1,37 veces más rápido en comparación al computador con procesador Intel Core i5 en cuanto al tiempo que tomó el procesamiento para el conteo vehicular.

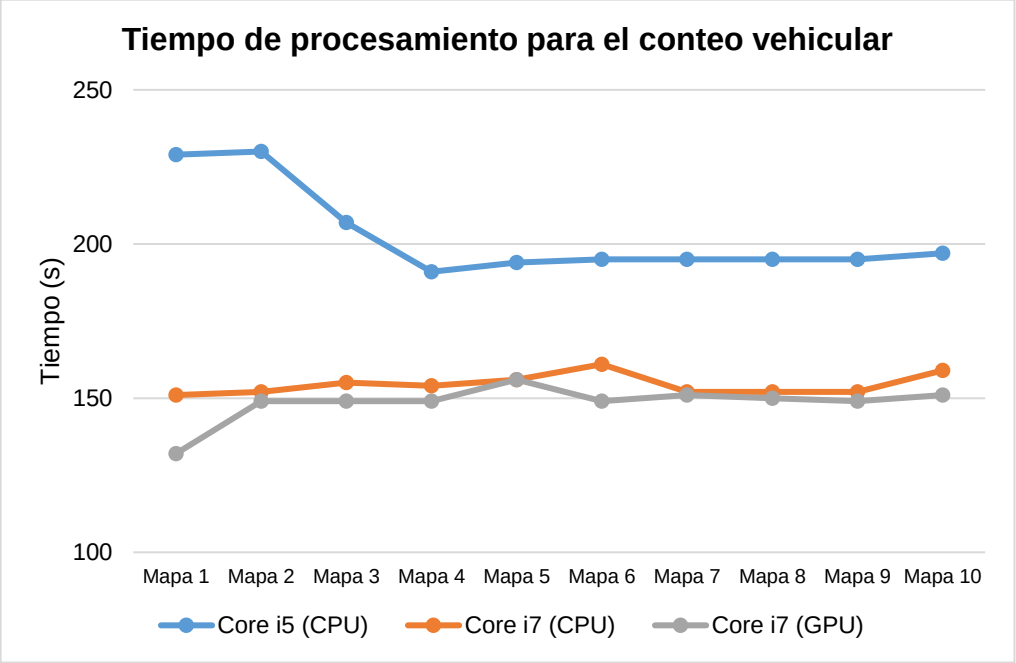


Figura 67. Comparación tiempo de procesamiento para el conteo vehicular.
Fuente: Propia del autor.
Elaboración: Autor.

En cuanto a la detección de vehículos se optó por realizar pruebas en cinco intersecciones aleatorias por cada escenario (escenario 1 hasta el escenario 10) y con ello se conoció el porcentaje de precisión del algoritmo para el conteo vehicular propuesto en el presente trabajo de titulación. Para ello se ha tomado en cuenta parámetros como: vehículos detectados, falsos positivos que son los objetos detectados pero distintos a los vehículos y falsos negativos que son los vehículos que no han sido detectados por el algoritmo tanto en la calle principal como en la calle secundaria en cada intersección. En la Tabla 15 se presentan los resultados obtenidos.

Tabla 15. Resultados de precisión del algoritmo en la detección de vehículos

Número de escenario	Información de la detección de vehículos											
	Calle primaria						Calle secundaria					
	Nombre calle	Vehículos detectados	Falsos positivos	Falsos negativos	Vehículos totales	Precisión	Nombre calle	Detectados	Falsos positivos	Falsos negativos	Vehículos totales	Precisión
Escenario 1	J. J. Peña	0	0	0	0	100,0 %	Mercadillo	5	0	0	5	100,0 %
Escenario 1	J. J. de Olmedo	0	0	0	0	100,0 %	Azuay	0	0	0	0	100,0 %
Escenario 1	Sucre	0	0	0	0	100,0 %	Miguel Riofrío	0	0	0	0	100,0 %
Escenario 1	Simón Bolívar	0	0	0	0	100,0 %	V. Rocafuerte	0	0	0	0	100,0 %
Escenario 1	J. J. Peña	0	0	0	0	100,0 %	10 de Agosto	3	0	0	0	100,0 %
Escenario 2	Simón Bolívar	11	0	3	14	78,6 %	Mercadillo	14	0	2	16	87,5 %
Escenario 2	B. Valdivieso	4	1	1	4	75,0 %	Azuay	15	5	2	12	83,3 %
Escenario 2	J. J. de Olmedo	9	0	4	13	69,2 %	Miguel Riofrío	9	0	2	11	81,8 %
Escenario 2	Simón Bolívar	3	0	0	3	100,0 %	V. Rocafuerte	15	2	1	14	92,9 %
Escenario 2	Sucre	8	0	2	10	80,0 %	10 de Agosto	7	0	4	11	63,6 %
Escenario 3	B. Valdivieso	14	0	4	18	77,8 %	Mercadillo	7	0	0	7	100,0 %
Escenario 3	J. J. de Olmedo	0	0	0	0	100,0 %	Azuay	18	6	1	13	92,3 %
Escenario 3	Simón Bolívar	0	0	0	0	100,0 %	Miguel Riofrío	0	0	0	0	100,0 %
Escenario 3	18 de Noviembre	15	0	3	18	83,3 %	V. Rocafuerte	16	2	1	15	93,3 %
Escenario 3	Sucre	13	0	5	18	72,2 %	10 de Agosto	17	1	2	18	88,9 %
Escenario 4	18 de Noviembre	12	0	6	18	66,7 %	Mercadillo	13	0	3	16	81,3 %
Escenario 4	Simón Bolívar	10	0	5	15	66,7 %	Azuay	14	5	1	10	90,0 %
Escenario 4	J. J. Peña	12	0	5	17	70,6 %	Miguel Riofrío	2	0	0	2	100,0 %
Escenario 4	Sucre	14	7	2	9	77,8 %	V. Rocafuerte	15	3	1	13	92,3 %
Escenario 4	J.J. de Olmedo	13	6	0	7	85,7 %	10 de Agosto	11	0	3	14	78,6 %
Escenario 5	Simón Bolívar	14	0	4	18	77,8 %	10 de Agosto	13	0	4	17	76,5 %
Escenario 5	18 de Noviembre	12	0	7	19	63,2 %	Azuay	19	5	1	15	93,3 %
Escenario 5	J. J. de Olmedo	12	0	8	20	60,0 %	V. Rocafuerte	13	1	1	14	85,7 %
Escenario 5	J. J. Peña	7	0	2	9	77,8 %	10 de Agosto	11	0	2	13	84,6 %
Escenario 5	B. Valdivieso	14	0	1	15	93,3 %	Miguel Riofrío	19	0	3	22	86,4 %
Escenario 6	Juan José Peña	7	0	6	13	53,8 %	Azuay	11	3	1	9	88,9 %
Escenario 6	18 de Noviembre	12	0	7	19	63,2 %	V. Rocafuerte	16	3	1	14	92,9 %
Escenario 6	Simón Bolívar	15	0	3	18	83,3 %	Miguel Riofrío	30	0	0	30	100,0 %
Escenario 6	Juan José Peña	12	0	7	19	63,2 %	Miguel Riofrío	2	0	0	2	100,0 %
Escenario 6	Sucre	12	0	7	19	63,2 %	10 de Agosto	9	0	3	12	75,0 %
Escenario 7	B. Valdivieso	9	1	2	10	80,0 %	Azuay	15	7	2	14	57,1 %
Escenario 7	Simón Bolívar	20	0	3	23	87,0 %	Mercadillo	13	0	2	15	86,7 %
Escenario 7	J. J. de Olmedo	18	0	6	24	75,0 %	Miguel Riofrío	14	0	3	17	82,4 %
Escenario 7	18 de Noviembre	15	0	4	19	78,9 %	V. Rocafuerte	16	3	1	14	92,9 %
Escenario 7	Sucre	16	0	5	21	76,2 %	10 de Agosto	17	0	3	20	85,0 %
Escenario 8	B. Valdivieso	14	0	6	20	70,0 %	Mercadillo	15	0	3	18	83,3 %
Escenario 8	Simón Bolívar	15	0	8	23	65,2 %	Azuay	31	6	1	26	96,2 %
Escenario 8	J. J. de Olmedo	18	0	5	23	78,3 %	Miguel Riofrío	25	0	2	27	92,6 %
Escenario 8	J. J. Peña	16	0	7	23	69,6 %	V. Rocafuerte	31	1	2	32	93,8 %
Escenario 8	Sucre	11	1	2	12	83,3 %	Azuay	26	4	2	24	91,7 %
Escenario 9	Azuay	20	1	5	24	79,2 %	Mercadillo	26	0	2	28	92,9 %
Escenario 9	B. Valdivieso	12	1	1	12	91,7 %	Azuay	26	5	3	24	87,5 %
Escenario 9	Simón Bolívar	21	0	2	23	91,3 %	Miguel Riofrío	27	0	0	27	100,0 %
Escenario 9	18 de Noviembre	17	0	6	23	73,9 %	V. Rocafuerte	22	1	2	23	91,3 %
Escenario 9	J. J. de Olmedo	7	0	2	9	77,8 %	10 de Agosto	28	0	1	29	96,6 %
Escenario 10	J. J. Peña	16	1	5	20	75,0 %	Mercadillo	28	0	0	28	100,0 %
Escenario 10	Simón Bolívar	17	0	6	23	73,9 %	Azuay	26	5	1	22	92,8 %
Escenario 10	B. Valdivieso	24	0	0	24	100,0 %	Miguel Riofrío	26	0	2	28	92,9 %
Escenario 10	Simón Bolívar	13	0	7	20	65,0 %	10 de Agosto	27	0	0	27	100,0 %
Escenario 10	Sucre	16	0	6	22	72,7 %	10 de Agosto	26	0	2	28	92,9 %

Fuente: Propia del autor.
Elaboración: Autor.

5.2.5. Tiempo de procesamiento al realizar acercamiento en cada intersección.

El tiempo de procesamiento que tardó el programa en presentar la imagen desde el momento que se seleccionó la opción para realizar un acercamiento en las calles en cada semáforo dentro de la GUI en MATLAB se presenta en la Tabla 16.

Tabla 16. Tiempo de procesamiento para el acercamiento del mapa.

Procesador	Tiempo (ms)
Intel Core i5 – 4210U (CPU)	3600
Intel Core i7 - 6700HQ (CPU)	3030
Intel Core i7 - 6700HQ (GPU)	2980

Fuente: Propia del autor.
Elaboración: Autor.

Como se puede observar en la Figura 68, en la comparación realizada existió una diferencia de milisegundos entre las tres pruebas realizadas en cada ordenador, donde el algoritmo tardó más tiempo en realizar el proceso en el primer ordenador.

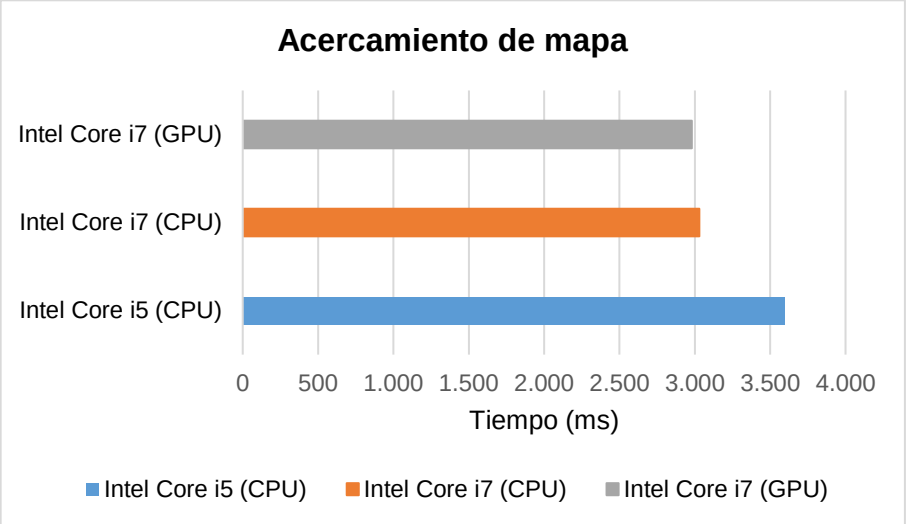


Figura 68. Comparación tiempo de procesamiento entre los ordenadores al realizar el acercamiento al mapa.
Fuente: Propia del autor
Elaboración: Autor.

5.2.6. Tiempo de procesamiento de la búsqueda la ruta con menor congestión.

También se ha realizado pruebas en lo que respecta al promedio del tiempo necesario que toman los algoritmos de búsqueda de la ruta más corta para presentar la o las rutas con menor congestión en pantalla en los dos ordenadores. Estos tiempos se presentan en la Tabla 17.

Tabla 17. Tiempo de procesamiento para la presentación de la ruta con menor congestión.

Procesador	Tiempo (ms)
Intel Core i5 - 4210U (CPU)	2720
Intel Core i7 - 6700HQ (CPU)	2180
Intel Core i7 - 6700HQ (GPU)	2050

Fuente: Propia del autor.
Elaboración: Autor.

En la Figura 69 se presenta una comparación mediante una gráfica entre el tiempo requerido por cada ordenador para la presentación en pantalla la ruta con menor congestión. El menor tiempo se registró en el ordenador Dell utilizando la tarjeta de vídeo Nvidia, y se pudo demostrar que trabajar con la GPU permite realizar operaciones de manera más rápida y en menos tiempo.

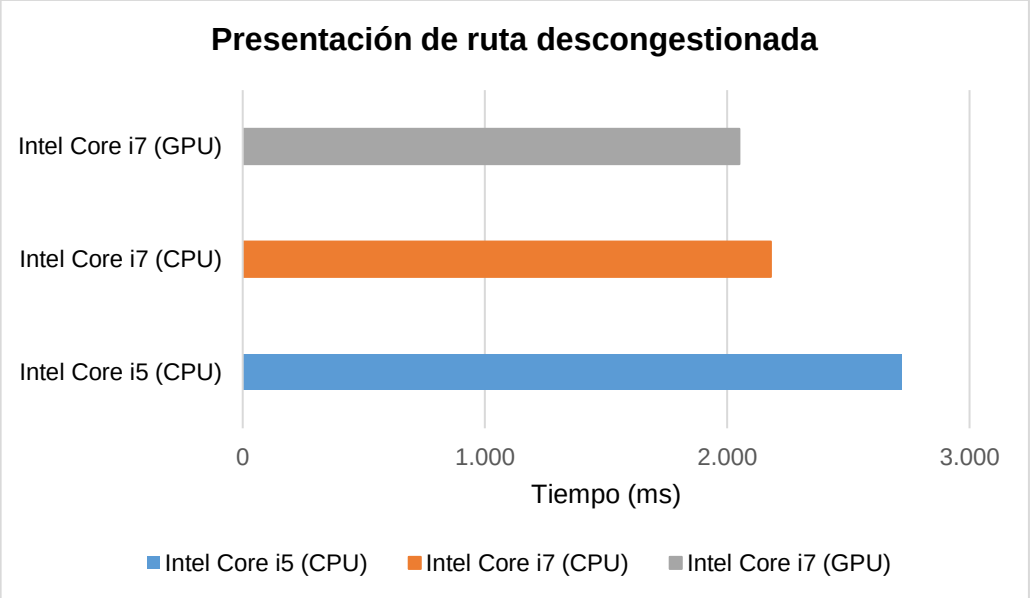


Figura 69. Tiempo de procesamiento entre los dos ordenadores al presentar ruta descongestionada.
Fuente: Propia del autor
Elaboración: Autor.

CAPITULO VI.
DISCUSIÓN

6.1. Análisis de la inversión y costos

Una parte esencial para una futura implementación de las soluciones para el conteo vehicular, es la estimación de costos, ya que esto es el recurso con el que toda empresa cuenta para la ejecución de un proyecto.

Este capítulo tiene como finalidad realizar un análisis de costos del proyecto tanto en su implementación como en su desarrollo. Para ello se ha visto la necesidad de realizar un análisis por cada posible solución y evaluar la más factible desde un punto de vista técnico como económico.

6.1.1. Costos de implementación del proyecto.

Para la selección de los equipos se tomó en cuenta consideraciones y aspectos importantes tales como:

- Características técnicas de cada equipo requeridas para la implementación del proyecto.
- Variación de los costos presupuestados.
- Garantía de todos los equipos.
- Compatibilidad de los equipos para comunicarse entre sí.
- Asesoramiento acerca del funcionamiento de cada equipo.
- Prestigio de las marcas.

6.1.2. Costos de inversión.

Estos costos corresponden a los costos pre-operativos que se dan desde la concepción de la idea para emprender el proyecto hasta poco antes de la producción del servicio o producto. Para ello se contemplaron los siguientes costos:

- **Equipos:** incluye todos los equipos y accesorios requeridos en el diseño del sistema.
- **Infraestructura:** dentro de estos costos se presentan torres, casetas y la energía para el funcionamiento.
- **Instalación:** comprende los precios por la instalación y puesta a punto de las estaciones.
- **Ingeniería:** se contempla el estudio, diseño, inspección y todo lo necesario para la realización del proyecto.

6.1.2.1. Costo de equipos.

Se consideró los valores referenciales de los equipos y accesorios necesarios para la implementación cada posible solución. A los equipos y accesorios importados se sumaron los

impuestos que se debe pagar para ingresar los productos al país. Estos valores dependen del artículo a importar y se encuentran en la página del Servicio Nacional de Aduana del Ecuador, como son: AD-VALOREM (Arancel Cobrado a las Mercancías), el FODINFA (Fondo de Desarrollo para la Infancia), ICE (Impuesto a los Consumos Especiales) y el IVA (Impuesto al Valor Agregado).

A continuación se presenta una descripción y las tablas de los costos de las cuatro soluciones posibles que se ha visto conveniente para la ejecución del proyecto:

6.1.2.1.1. Solución mediante la utilización del drone DJI Phanthom 3 Standard.

Esta solución presenta como ventaja un menor costo de los equipos y accesorios necesarios, que en comparación a las demás opciones resultó ser más económica (Tabla 18) y con la utilización de Arduino se permite reducir la complejidad de los circuitos electrónicos. Pero teniendo como desventaja una menor autonomía de vuelo, un rango de alcance menor, factores como la lluvia y el viento, así como la necesidad de un piloto que ayude a sobrevolar cada uno de los drones sobre el casco urbano.

La arquitectura que se pretende utilizar es la siguiente: una vez que se han captado las imágenes aéreas, estas se suben a la nube para que el ordenador procese las imágenes y los datos obtenidos del procesamiento son enviados nuevamente a la nube para que cada tarjeta Arduino obtenga los datos de la vía con mayor congestión y tome una decisión para el control del tiempo de duración de las fases del semáforo.

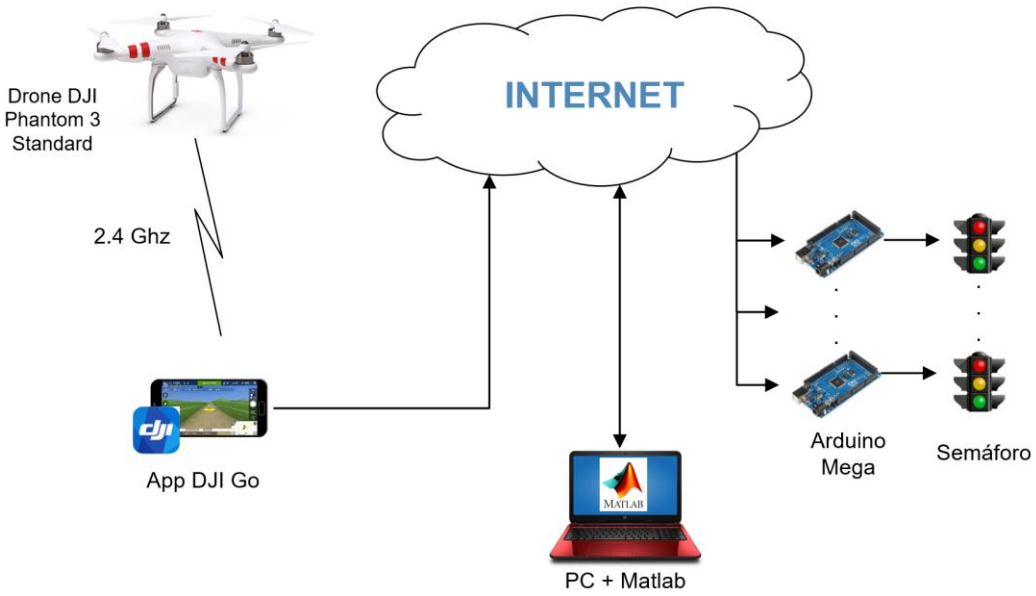


Figura 70. Arquitectura primera solución con drone DJI Phanthom 3.
Fuente: Propia del autor.
Elaboración: Autor.

Tabla 18. Costo de los equipos y accesorios para la primera solución

Equipo	Especificaciones	Cantidad	Costo Unitario (USD$)	Costo Total (USD$)
Drone DJI Phantom 3 Standard	- Tiempo de vuelo: 23 minutos - Rango de alcance: 1 Km - Resolución de la cámara: 12 MP	10	593,81	5.938,10
Batería de repuesto de vuelo inteligente	- Capacidad: 4480 mAh - Tipo de batería: LiPo 4S - Voltaje: 15,2 V	10	177,31	1.773,10
Hélices autoajustables	- Peso de cada hélice: 13 g - Diámetro y paso de tornillo: 24×12,7 cm.	10 pares	7,14	71,40
Protector para hélices	- Elemento adicional de seguridad	10	22,61	190,00
Corrosión X	- Spray - Evita corrosión	48	20,53	985,44
Revestimiento protector	- Aerosol - Protección circuitos electrónicos	30	35,65	1.069,50
Semáforo Vehicular LED (200 mm)	- Policarbonato - Colores: Verde-Ámbar-Rojo - 220/110 Vca - 50/60 Hz	60	494,93	29.635,80
Computador portátil Genérico	- Procesador Intel Core i7 (7ma Generación) - 16 GB de memoria RAM - Tarjeta de vídeo: Nvidia GTX 1060	1	1.800,00	1.800,00
Arduino Mega 2560	- Placa de adquisición de datos - Voltaje de operación: 5 V - 54 pines E/S - 16 pines análogos	32	54,36	1.739,52
Arduino WiFi Shield	- Antena integrada. - Encriptación: WEP y WPA2 Personal - Estándar: 802.11 b/g - Frecuencia : 2,4 GHz	32	95,18	3.045,76
Imprevistos 10 %				4.624,86
TOTAL				**50.873,48**

Fuente: Propia del autor
Elaboración: Autor

6.1.2.1.2. Solución mediante la utilización del drone DJI Mavic PRO.

Una de las ventajas que posee esta solución es la detección de obstáculos que viene incorporado en el vehículo aéreo no tripulado, un peso menor, un rango de alcance y un tiempo de vuelo mayor en comparación a la solución anterior. Como desventaja existe un mayor costo a cambio de las mejores funcionalidades que presenta el vehículo (Tabla 19).

La arquitectura de esta solución es semejante a la presentada en la Figura 70, realizándose el mismo proceso que la solución anterior con la única diferencia de que presenta otro modelo de drone que es el DJI Mavic Pro (Figura 71).

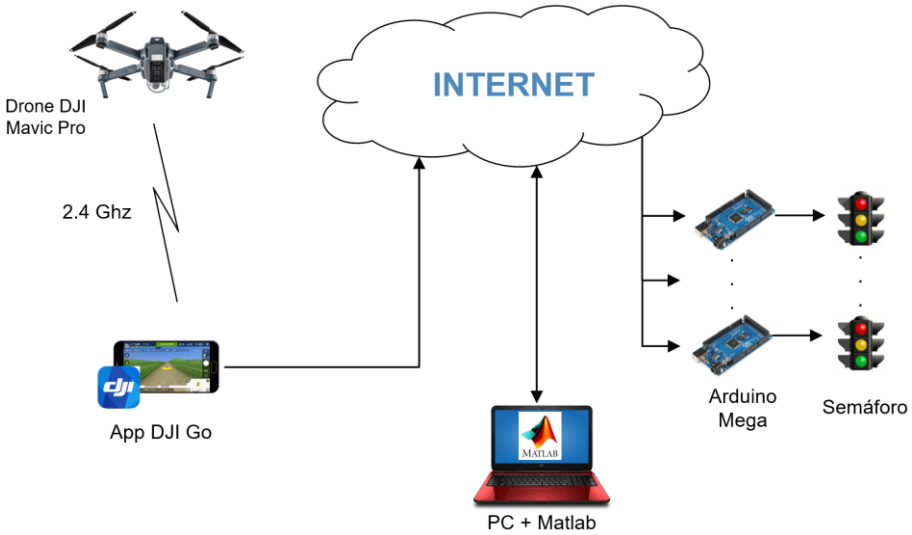


Figura 71. Arquitectura primera solución con drone DJI Mavic Pro.
Fuente: Propia del autor.
Elaboración: Autor.

Tabla 19. Costo de los equipos y accesorios para la segunda solución.

Equipo	Especificaciones	Cantidad	Costo Unitario (USD$)	Costo Total (USD$)
Drone DJI Mavic Pro	- Tiempo de vuelo: 27 minutos - Rango: 7 Km - Resolución de la cámara: 12 MP - Detección de obstáculos	10	1.188,00	11.880,00
Batería de repuesto de vuelo inteligente	- Capacidad: 3830 mAh - Voltaje: 11,4 V - Tipo de batería: LiPo 3S - Energía: 43,6 Wh - Peso neto: Aprox. 0.5 lbs (240 g) - Temperatura de operación: 5 °C - 40 °C	10	105,91	1.059,10
Hélices plegables	- Diámetro: 21,1 × 7,6 cm	12 pares	10,71	128,52
Protector para hélices	- Elemento adicional de seguridad	12	17,85	214,20
Corrosión X	- Evita corrosión	48	20,53	985,44
Revestimiento protector	- Protección circuitos electrónicos	30	35,65	1.069,50
Semáforo Vehicular LED (200 mm)	- Policarbonato - Colores: Verde-Ámbar-Rojo - 220/110 Vca - 50/60 Hz	60	494,93	29.635,80
Computador Portátil Genérico	- Procesador Intel Core i7 (7ma Generación) - 16 GB de memoria RAM - Tarjeta de vídeo: Nvidia GTX 1060	1	1.800,00	1.800,00
Arduino Mega 2560	- Voltaje de operación: 5 V - 54 pines E/S - 16 pines análogos	32	54,36	1.739,52
Arduino WiFi Shield	- Antena integrada. - Encriptación: WEP y WPA2 Personal - Estándar: 802.11 b/g - Frecuencia: 2,4 GHz	32	95,18	3.045,76
Imprevistos 10 %				5.155,78
TOTAL				**56.713,62**

Fuente: Propia del autor
Elaboración: Autor

6.1.2.1.3. Solución mediante la utilización del drone DJI Phanthom 4 Pro.

El vehículo aéreo no tripulado en esta solución en comparación a los anteriores posee mejores funcionalidades en cuanto al tiempo de vuelo y la resolución de la cámara que viene integrada en el vehículo al igual que los anteriores drones de la marca DJI.

El costo de esta solución es mucho mayor debido al modelo de vehículo y sus accesorios que poseen mejores prestaciones (Tabla 20). Al igual que las soluciones anteriores, se realiza el mismo procedimiento, pero con distinto modelo de drone, tal como se presenta en la Figura 72.

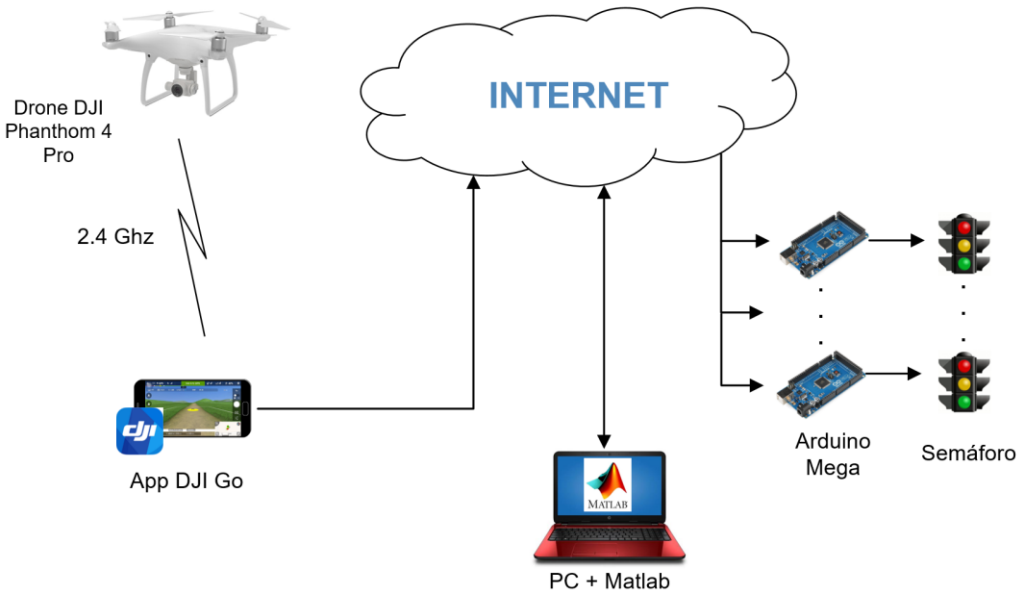


Figura 72. Arquitectura primera solución con drone DJI Phanthom 4 Pro.
Fuente: Propia del autor.
Elaboración: Autor.

Tabla 20. Costo de los equipos y accesorios para la tercera solución

Equipo	Especificaciones	Cantidad	Costo Unitario (USD$)	Costo Total (USD$)
Drone DJI Mavic Pro	- Tiempo de vuelo: 30 minutos - Rango: 7 Km - Cámara incorporada - Resolución de la cámara: 20 MP	10	1.783,81	17.838,10
Batería de repuesto de vuelo inteligente	- Capacidad: 5870 mAh - Tipo de batería: LiPo 4S	10	201,11	2.011,10
Hélices plegables	- Peso: 13 g - Diámetro: 24 × 12,7 cm.	12 pares	10,71	128,52
Protector para hélices	- Elemento adicional de seguridad	12	22,61	271,32

Corrosión X	- Spray - Evita corrosión	48	20,53	985,44
Revestimiento protector para circuitos electrónicos	- Aerosol - Protección circuitos electrónicos	30	35,65	1.069,50
Semáforo Vehicular LED (200 mm)	- Policarbonato - Colores: Verde-Ámbar-Rojo - 220/110 Vca - 50/60 Hz	60	494,93	29.635,80
Computador Portátil Genérico	- Procesador Intel Core i7 (7ma Generación) - 16 GB de memoria RAM - Tarjeta de vídeo: Nvidia GTX 1060	1	1.800,00	1.800,00
Arduino Mega 2560	- Placa de adquisición de datos - Voltaje de operación: 5 V - 54 pines E/S - 16 pines análogos	32	54,36	1.739,52
Arduino WiFi Shield	- Antena integrada. - Encriptación: WEP y WPA2 Personal - Estándar: 802.11 b/g - Frecuencia: 2,4 GHz	32	95,18	3.045,76
Imprevistos 10%				5.852,51
TOTAL				**64.377,57**

Fuente: Propia del autor.
Elaboración: Autor.

6.1.2.1.4. Solución mediante la utilización de globos cautivos con helio.

Esta solución se diferencia de las anteriores ya que permite sobrevolar el casco urbano y por ende la captación de las imágenes aéreas mediante cámaras montadas en globos cautivos inflados con helio, evitando el uso de personal que pilotee los anteriores vehículos, por lo que se necesita distintos equipos y accesorios. Sus costos no difieren de gran manera que las soluciones anteriores (Tabla 21).

Las desventajas que se presentan son la desestabilización al momento de captar las imágenes que se puede dar debido a situaciones donde exista mayor velocidad de viento, la cantidad de helio que se necesita para inflar los globos y la mayor cantidad de globos y cámaras (en este caso un globo y una cámara por cada semáforo) para cubrir el casco urbano.

La arquitectura que se pretende utilizar se presenta en la Figura 73 y se describe a continuación: las fotos que son captadas mediante la cámara se almacenan en la nube para que el ordenador las obtenga y realice el procesamiento, luego los datos obtenidos mediante el procesamiento de imágenes son enviados nuevamente a la nube para que cada Arduino encienda el semáforo en función de estos datos.

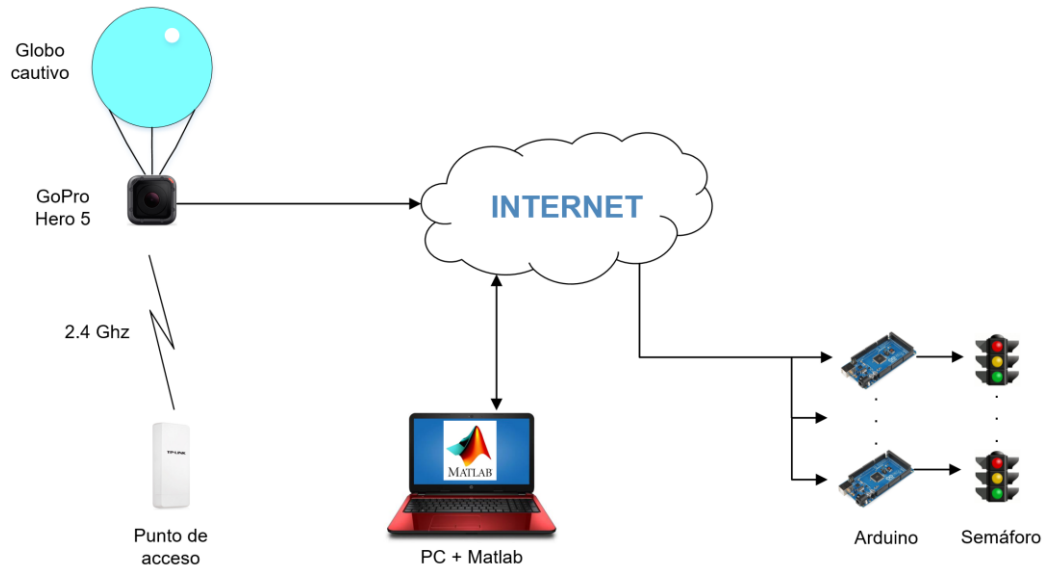


Figura 73. Arquitectura primera solución con globo cautivo.
Fuente: Propia del autor.
Elaboración: Autor.

Tabla 21. Costo de los equipos y accesorios para la cuarta solución

Equipo	Especificaciones	Cantidad	Costo Unitario (USD$)	Costo Total (USD$)
Globo cautivo o esferas cautivas	- Globo 2 m de diámetro - Capacidad: 4,19 m^3	30	408,00	12.240,00
Computador portátil genérico	- Procesador Intel Core i7 (7ma Generación) - 16 GB de memoria RAM - Tarjeta de vídeo: Nvidia GTX 1060	1	1.700,00	1.700,00
Arduino Mega 2560	- Placa de adquisición de datos	32	54,36	1.739,52
Cámara GoPro Hero 5 Session	- Resistente al agua - Resolución: 10 MP - GPS incorporado	30	343,85	10.315,50
Tarjetas micro SDHC SanDisk	- Capacidad: 32 GB	30	28,00	840,00
Arnés para amarre	- Sujeción de la cámara al globo	30	18,51	555,30
Cuerda trenzada	- Diámetro 2mm - Polipropileno - Carretes de 1 km	10 carretes	39,90	399,00
Válvula de inflado con manguera para helio	- Válvula reguladora	10	112,00	1.120,00
Punto de acceso TP-Link TL-WA7210N	- Punto de acceso a 2,4 GHz - Antena 12dBi	15	74,90	1.123,50
Arduino WiFi Shield	- Antena integrada. - Encriptación: WEP y WPA2 Personal - Estándar: 802.11 b/g - Frecuencia: 2,4 GHz	32	95,18	3.045,76
Semáforo Vehicular LED (200 mm)	- Policarbonato - Colores: Verde-Ámbar-Rojo - 220/110 Vca - 50/60 Hz	60	494,93	29.635,80
Imprevistos 10 %				6.271,44
TOTAL				**68.985,82**

Fuente: Propia del autor
Elaboración: Autor

6.1.2.2. Costos de infraestructura.

Abarcan el campo de obra civil y lo que incurre en cuanto a instalaciones eléctricas. Para la implementación de este proyecto, se necesita colocar infraestructura en la parte de la semaforización, presentando los materiales necesarios y sus costos en la Tabla 22.

Tabla 22. Costos de infraestructura.

Descripción	Cantidad	Costo Unitario (USD$)	Costo Total (USD$)
Instalación a tierra	30	80,76	2.422,80
Acera de hormigón e = 12 cm	40	32,36	1.294,40
Rotura de acera existente e = 20 cm	40	4,89	195,60
Rotura de asfalto existente e = 20 cm	30	4,71	188,40
Reposición de asfalto	30	6,23	186,90
Báculo tronco cónico	30	859,46	25.783,80
Extensión alargada de 3” (4 m)	30	134,48	4.034,40
Bajante de báculo	30	61,48	1.844,40
Basamento para báculo tronco cónico (0,80x0,80x0,80 m)	30	306,20	9.186,00
Cable 4x14 AWG para instalación entre postes	5000 m	5,87	29.350,00
Kit de acometida	10	89,75	897,5
Imprevistos 10 %			8.292,262
TOTAL			**91.214,88**

Fuente: Propia del autor
Elaboración: Autor

6.1.2.3. Costos de instalación.

Contempla los rubros para la mano de obra requerida en la puesta a tierra, rotura y reposición de asfalto, instalación de los semáforos, entre otros. Los costos son presentados en la Tabla 23.

Tabla 23. Costos de instalación

Cantidad	Descripción	Costo Unitario (USD$)	Costo Total (USD$)
1	Mano de obra	3.000,00	3.000,00
Imprevistos 10 %			300,00
TOTAL			**3.300,00**

Fuente: Propia del autor
Elaboración: Autor

6.1.2.4. Costos de ingeniería.

Considera los costos de diseño del proyecto como son: el estudio de la situación actual, diseño de los algoritmos, licencia del programa para el sistema y demás aspectos considerados en el diseño del trabajo de titulación (Tabla 24).

Tabla 24. Costos de ingeniería

Cantidad	Descripción	Costo Unitario (USD$)	Costo Total (USD$)
1	Licencia anual del sistema	1.800,00	1.800,00
TOTAL			**1.800,00**

Fuente: Propia del autor
Elaboración: Autor

6.1.3. Costos operativos.

Estos costos son los que se dan desde que el proyecto se pone en marcha y a diferencia de los costos de inversión que se dan una sola vez, los costos operativos son periódicos, permitiendo que el proyecto funcione día a día.

En el caso de las soluciones que utilizan de drones se necesita pilotos para el manejo de los mismos (Tabla 25), mientras que en la cuarta solución se necesita de personal para el mantenimiento de los globos (Tabla 26).

Tabla 25. Costos operativos para la solución con drones.

Personal	Cantidad	Costo Mensual (USD$)	Costo Anual (USD$)
Capacitación del personal	1	200,00	2.400,00
Pilotos para el vuelo de los drones	8	375,00	36.000,00
Imprevistos 10 %			3.840,00
TOTAL			**42.240,00**

Fuente: Propia del autor
Elaboración: Autor

Tabla 26. Costos operativos para la solución con globos cautivos.

Personal	Cantidad	Costo Mensual (USD$)	Costo Anual (USD$)
Bombona de helio recargable (9,10 m^3)	20	300,00	72.000,00
Capacitación del personal	1	200,00	2.400,00
Personal para el mantenimiento de los globos	5	375,00	22.500,00
Imprevistos 10 %			7.440,00
TOTAL			**81.840,00**

Fuente: Propia del autor
Elaboración: Autor

6.1.4. Costos totales.

En este apartado se presenta el resumen de los costos de inversión por cada posible solución (Tabla 27), mientras que en la Tabla 28 se presentan los costos totales operativos y de ingeniería en cada solución.

Tabla 27. Costos de inversión por cada solución.

Descripción	Primera solución	Segunda solución	Tercera solución	Cuarta solución
	Valor (USD$)	Valor (USD$)	Valor (USD$)	Valor (USD$)
Costos de equipos	50.873,48	56.713,62	64.377,57	68.985,82
Costos de infraestructura	91.214,88	91.214,88	91.214,88	91.214,88
Costos de instalación	3.300,00	3.300,00	3.300,00	3.300,00
COSTO TOTAL	**145.388,36**	**151.228,50**	**158.892,45**	**163.500,70**

Fuente: Propia del autor
Elaboración: Autor

En la Tabla 28 se debe tomar en cuenta que el costo de la cuarta solución varía respecto de a las demás soluciones, debido a que con los globos cautivos se puede tener mayor tiempo de servicio en comparación a los drones.

Tabla 28. Costos operativos y de ingeniería anuales por cada solución.

Descripción	Primera solución	Segunda solución	Tercera solución	Cuarta solución
	Valor Anual (USD$)	Valor Anual (USD$)	Valor Anual (USD$)	Valor Anual (USD$)
Costos operativos	42.240,00	42.240,00	42.240,00	81.840,00
Costos de ingeniería	1.800,00	1.800,00	1.800,00	1.800,00
COSTO TOTAL	**44.040,00**	**44.040,00**	**44.040,00**	**83.640,00**

Fuente: Propia del autor
Elaboración: Autor

6.1.5. Análisis de las soluciones planteadas.

En las soluciones expuestas en los apartados anteriores se ha recurrido a la utilización de dos tipos de vehículos aéreos no tripulados como son drones y globos o esferas cautivas infladas con helio, ya que poseen ventajas como un vuelo estacionario, es decir, mantener una posición constante sobre un punto fijo, su menor costo en comparación a los demás tipos de vehículos, así como también la ventaja de no estar tripulados y un fácil mantenimiento. En la Tabla 29, se presenta de manera resumida una comparación de las características de cada solución.

Tabla 29. Comparación de características entre las cuatro posibles soluciones.

Características	Vehículo			
	Drone DJI Phanthom 3 Standard	**DJI Mavic PRO**	**Drone DJI Phanthom 4 Pro**	**Globos cautivos**
Tiempo de vuelo	23 minutos	27 minutos	30 minutos	Con 4190 litros de helio flota alrededor de dos días
Resolución de la cámara	12 MP	12 MP	20 MP	10 MP
Cámara integrada	Si	Si	Si	No
Número de vehículos utilizados	8	8	8	30
Alimentación	Batería	Batería	Batería	Helio
Capacidad de la batería	4480 mAh	3830 mAh	5870 mAh	-
Piloto	Si	Si	Si	No
Nivel de capacitación del personal	Medio	Medio	Medio	Bajo

Fuente: Propia del autor
Elaboración: Autor

Con relación al costo que presentaron las soluciones, cada una tiene ventajas y desventajas respecto la una de la otra.

- Una ventaja que poseen los drones es el poder movilizarse de un lugar a otro rápidamente y regular su altura de una manera sencilla a través de aplicaciones desarolladas por la misma empresa para dispositivos celulares.
- Como ventaja que presenta el uso de globos cautivos es la posibilidad de sobrevolar el casco urbano de la ciudad mucho más tiempo (en algunos casos hasta días en condiciones óptimas).
- Con el uso de drones se tiene un punto a favor, el cual es el traer una cámara incorporada en el vehículo y un estabilizador para captar imágenes de mayor calidad.
- Los globos cautivos no necesitan de piloto para poder sobrevolar, necesitando solamente de personal de mantenimiento en caso de existir algún problema con el globo o ya sea para inflar nuevamente con helio los mismos.

6.2. Factibilidad del proyecto

6.2.1. Técnica.

La implementación de las posibles soluciones para la entidad interesada es técnicamente factible, debido a que los equipos y accesorios necesarios para la adquisición de las imágenes aéreas, así como para la implementación de la semaforización son de fácil adquisición dentro y fuera del país.

En cuanto a los problemas generados por factores climáticos como la lluvia, humedad y viento que son muy habituales en la ciudad de Loja. Se pueden resolver mediante el uso de aditamentos para los componentes y accesorios que se pueden colocar en el vehículo aéreo.

6.2.2. Económica.

El desarrollo del proyecto no representa costos elevados para alguna institución que desease implementar cualquiera de las posibles soluciones. Los equipos y accesorios son fácilmente accesibles tanto en el país como fuera del mismo.

Si se desearía calcular indicadores de rentabilidad como la TIR (Tasa Interna de Retorno) y el VAN (Valor Actual Neto), a la entidad contratante le corresponde evaluar si le resulta rentable el trabajo propuesto. Esto en vista de no tener conocimientos sobre el valor de ingresos que le generarían a la empresa que desee implementar el sistema.

6.3. Ventajas y desventajas del proyecto en su desarrollo

Gracias a la ayuda del vehículo aéreo no tripulado se pudo obtener las imágenes aéreas con gran calidad, de una manera eficiente, rápida y con costos reducidos, siendo una gran alternativa para cambiar o mejorar el sistema de semáforos tradicional que se tiene en la ciudad de Loja.

Otra ventaja que se observó en este tipo de vehículos es la facilidad para realizar un vuelo estacionario estabilizado, es decir, se puede mantener estático el vehículo sobre un punto y a una altura establecida (en este caso 100 metros), facilitando de esta manera la obtención de las imágenes aéreas y ayudando a una mejor contabilización de los vehículos.

Una de las desventajas que se presentaron al momento de realizar las pruebas para el presente trabajo de titulación ha sido la velocidad de cómputo de los ordenadores con los que se trabajó y la cantidad de memoria RAM utilizada, ya que al momento de crear el mosaico con las imágenes aéreas y la contabilización de los vehículos, el proceso llevó alrededor de 5 minutos cuando se trabajó con la CPU del ordenador con procesador core i5 y cerca de 2 minutos con la GPU con procesador core i7. Esto debido a la resolución del mosaico resultante, aunque la velocidad de procesamiento puede mejorar al utilizar un cómputo paralelo mediante un servidor con mejores características.

Otro factor que influyó para la realización del trabajo de titulación fue el clima, ya que, teniendo un clima adverso como la lluvia y el viento en la ciudad de Loja, fue imposible volar el vehículo aéreo no tripulado en estas condiciones ya que existen problemas como la desestabilización de la cámara y con ello imágenes de mala calidad, como también el daño que pueda tener el vehículo por el agua ya que no se dispuso de aditamentos y recubrimientos en los circuitos electrónicos para la resistencia al agua.

Como último factor, la autonomía de vuelo fue muy limitada en el vehículo aéreo no tripulado, que cómo se sabe demanda un gran consumo de energía en caso de los drones y con ello un

menor tiempo de vuelo, dificultando así la realización de las pruebas para el presente proyecto, siendo esta una de las mayores desventajas que se pudo observar al utilizar un solo vehículo aéreo no tripulado.

6.4. Objetivos futuros

A futuro se plantea una mejora en cuanto a la velocidad de procesamiento del algoritmo sin tener que reducir el tamaño de la imagen y con ello evitando pérdidas de información, una mejor detección de vehículos y aprovechar de mejor manera la capacidad de cómputo de la tarjeta de vídeo Nvidia incorporada en el ordenador para acelerar el tiempo y la eficiencia del algoritmo mediante el procesamiento paralelo.

Con el adelanto a pasos agigantados de la tecnología, la mejora de la capacidad de las baterías y la eficiencia de los motores, se espera que la autonomía de vuelo del drone, como también su velocidad mejore, con el fin de poder trasladarse de un punto a otro en el menor tiempo posible, aumentando su cobertura y reduciendo el número de vehículos aéreos utilizados en el proyecto.

Se espera realizar pruebas a mayor altitud para de esta manera mejorar la velocidad de procesamiento, evitando problemas debido a los vehículos en movimiento que causaron inconvenientes al momento de crear la vista panorámica tipo mosaico.

Se plantea implementar el sistema en tiempo real mediante cualquiera de las cuatro posibles soluciones presentadas anteriormente, haciendo un algoritmo mucho más robusto, colocando los distintos dispositivos en red, como también una aplicación para dispositivos móviles en dónde el usuario pueda observar en su smartphone la ruta con la menor congestión hacia su destino.

Hoy en día se dispone de aditamentos (Corrosion Technologies, 2017) y recubrimientos para los componentes y circuitos electrónicos (HumiSeal, 2017) que ayudan a proteger al vehículo aéreo no tripulado ante condiciones ambientales tales como la lluvia, humedad y polvo. Al realizar las pruebas a futuro se tendrá en cuenta este tipo de aditamentos para que no exista mayor problema con respecto a los factores antes mencionados y observar el factor de disponibilidad del sistema que se tendría si se lo implementara.

Finalmente, se espera poder detectar los vehículos de una manera más eficaz y con mayor precisión, ya que como se vio anteriormente en las tablas de resultados, se tiene problemas con vehículos ubicados bajo sombras, dificultando su detección en este tipo de escenarios con baja iluminación.

CONCLUSIONES

Para la creación de la vista panorámica tipo mosaico en el presente trabajo de titulación se ha optado por trabajar con el algoritmo detector SIFT para encontrar la mayor cantidad de puntos característicos y el modelo de estimación de parámetros RANSAC para el cálculo de las mejores correspondencias entre las imágenes. Para ello se utilizaron herramientas del Toolbox VLFeat y Toolbox Balu para lograr el resultado deseado mediante la utilización de cuatro imágenes.

Según los resultados obtenidos en las pruebas realizadas con los tres canales de color de la imagen y escala de grises, resultó mejor trabajar con imágenes en formato a escala de grises para la creación de vistas panorámicas tipo mosaico, ya que se obtuvo mayor cantidad de correspondencias y un mejor resultado con interpolación por vecinos más cercanos

Se pudo observar que resulta mejor captar las imágenes a mayor altitud, ya que si se captan a menor altura puede existir problemas al momento de fusionar las imágenes debido al movimiento de los vehículos y al bajo número de correspondencias que se encuentren. De la misma manera se logra cubrir más área y reduce el tiempo de procesamiento, sin embargo, una mayor altitud no garantiza que los vehículos sean detectados de mejor manera ya que se debe tomar en consideración que los vehículos puedan observarse en la imagen.

El tiempo de procesamiento depende del procesador, y con la utilización de la GPU se obtuvo un incremento del rendimiento el sistema. Esto debido a la arquitectura de cálculo paralelo de la tarjeta de vídeo Nvidia instalada en el ordenador.

El sistema está diseñado para trabajar en horas de la mañana y tarde (07h00 hasta las 18h00), ya que si se quiere trabajar en horas de la noche se debe recurrir a otro tipo de procesamiento, así como también otro tipo de equipos especializados para captar imágenes nocturnas. Además, de que en horario nocturno no se presenta tanta afluencia vehicular como en horas de la mañana y tarde.

RECOMENDACIONES

Se recomienda volar el vehículo aéreo no tripulado a una altura de 100 metros. Esto con el fin de utilizar menor cantidad de vehículos y cubrir una mayor área; como también evitar dificultades producidas al momento de la creación de vistas panorámicas y la detección de los vehículos ya que al existir vehículos en movimiento produce problemas al momento de encontrar las correspondencias y producir mayor cantidad de errores de píxeles.

Otra recomendación al volar el vehículo aéreo no tripulado es realizar las pruebas en zonas con baja interferencia o buscar un canal con menor saturación debido a que las bandas ISM al no ser licenciadas, por lo general presentan una gran congestión. Para la búsqueda de los canales con menor congestión se puede recurrir a varias aplicaciones en Android, una de estas es Wifi Analyzer.

Antes de la creación de la vista panorámica tipo mosaico es recomendable reducir el tamaño de las imágenes. Esto con el fin de mejorar el tiempo que tarda en procesar las imágenes y a su vez evitar errores por falta de memoria en el ordenador.

Al momento de cargar la aplicación en el programa MATLAB, se tiene que instalar con anterioridad los Toolbox VLFeat y Toolbox Balu, como también la instalación de compiladores para funciones en C. De preferencia, se debe utilizar una versión a partir de MATLAB 2015a, ya que posee herramientas para el cálculo de la ruta con menor congestión mediante la utilización de grafos dirigidos.

Por último, si se desea verificar que existe comunicación entre el programa MATLAB y la tarjeta Arduino, se necesita instalar el Toolbox ArduinoIO que se encuentra disponible en la página de MathWorks y trabajar con las funciones *arduino* y *encenderLEDS,* las cuales se encuentran en el Anexo 2.

BIBLIOGRAFÍA

AGN. (2010). En Loja presentan plan para evitar la Congestión Vehicular. *El Mercurio*. Retrieved from http://www.elmercurio.com.ec/254301-en-loja-presentan-plan-para-evitar-la-congestion-vehicular/

Agrawal, A., & Hickman, M. (2004). Automated extraction of queue lengths from airborne imagery. In *Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on* (pp. 297–302). IEEE.

Aguirre, D. (2012). Construcción de vistas panorámicas tipo “mosaico”. Tarea Presentada en la asignatura de Visión por Computador. .

Arduino. (2016). Arduino Blog. Retrieved March 23, 2017, from https://blog.arduino.cc/

AVB. (2012). El tráfico de Loja preocupa. *Diario El Mercurio*. Cuenca. Retrieved from http://www.elmercurio.com.ec/351163-el-trafico-de-loja-preocupa/#.U462Uvl5OE8

Barragán, D. (2015). Shape recognition using 2-D correlation. Retrieved from http://matpic.com/

Bay, H., Tuytelaars, T., & Van Gool, L. (2006). Surf: Speeded up robust features. In *European conference on computer vision* (pp. 404–417). Springer.

Beis, J. S., & Lowe, D. G. (1997). Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on* (pp. 1000–1006). IEEE.

Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, *60*(6), 503–516. https://doi.org/10.1090/S0002-9904-1954-09848-8

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324

Calonder, M., Lepetit, V., Strecha, C., & Fua, P. (2010). BRIEF: Binary Robust Independent Elementary Features (pp. 778–792). Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-15561-1_56

Chávez, C. (2015). *Sistema de semaforización inteligente para el control de flujo vehicular mediante el Procesamiento Digital de Imágenes*. Universidad Técnica de Ambato. Retrieved from http://repositorio.uta.edu.ec/bitstream/123456789/13061/1/Tesis_t1033ec.pdf

Chen, X., & Meng, Q. (2013). Vehicle detection from UAVs by using SIFT with implicit shape model. In *2013 IEEE International Conference on Systems, Man, and Cybernetics* (pp. 3139–3144). IEEE.

Choi, S., Kim, T., & Yu, W. (1997). Performance evaluation of RANSAC family. *Journal of Computer Vision*, *24*(3), 271–300.

Chum, O., & Matas, J. (2005). Matching with PROSAC — Progressive Sample Consensus. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 220–226). IEEE. https://doi.org/10.1109/CVPR.2005.221

Cormen, T. H., Leirserson, C. E., & Rivest, R. L. (2001). *Introduction to algorithms*. MIT Press. Retrieved from https://es.wikipedia.org/wiki/Búsqueda_en_profundidad

Corrosion Technologies, L. (2017). CorrosionX. Retrieved April 22, 2017, from http://www.corrosionx.com/corrosionx.html

Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13*(1), 21–27. https://doi.org/10.1109/TIT.1967.1053964

Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 886–893). IEEE. https://doi.org/10.1109/CVPR.2005.177

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik*, *1*(1), 269–271. https://doi.org/10.1007/BF01386390

Evans, C. (2009). Notes on the opensurf library. *University of Bristol, Tech. Rep. CSTR-09-001, January*.

Fischler, M. A., & Bolles, R. C. (1981). Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, *24*(6), 381–395.

Ford, L. R., & Fulkerson, D. R. (1962). *Flows in networks*.

Gao, H. W., Li, D., & Chen, F. G. (2011). Image mosaics algorithm based on PSO and SIFT. In *Advanced Materials Research* (Vol. 225, pp. 150–153). Trans Tech Publ.

García, I. D. (2008). Vision artificial y Procesamiento Digital de Imágenes usando Matlab. Ibarra: Pontificia Universidad Católica del Ecuador.

Gleason, J., Nefian, A. V, Bouyssounousse, X., Fong, T., & Bebis, G. (2011). Vehicle detection from aerial imagery. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on* (pp. 2065–2070). IEEE.

González, A., Martínez de Pisón, F., Pernía, A., Alba, F., Castejón, M., Ordieres, J., & Vergara, E. (2006). *Técnicas y algoritmos básicos de visión artificial*. España. Retrieved from https://publicaciones.unirioja.es/catalogo/online/VisionArtificial.pdf

González, R. C., & Woods, R. E. (1996). *Tratamiento digital de imágenes*. Addison-Wesley Iberoamericana. Retrieved from https://books.google.com.ec/books/about/Tratamiento_Digital_de_Imagenes.html?id=TQT6DF3_YgkC&redir_esc=y

Guamán, J. G. (2012). *Estudio y Análisis de Soluciones al Congestionamiento Vehicular en el Centro Histórico de la Ciudad de Loja* . *Titulación de Ingeniería Civil*. Universidad Técnica Particular de Loja. Retrieved from

http://dspace.utpl.edu.ec/bitstream/123456789/4021/1/JUAN GABRIEL GUAMAN MOROCHO-ETRANLOJA.pdf

Hearst, M. A., Dumais, S. T., Osuna, E., Platt, J., & Scholkopf, B. (1998). Support vector machines. *IEEE Intelligent Systems and Their Applications*, *13*(4), 18–28. https://doi.org/10.1109/5254.708428

Hong, J., Lin, W., Zhang, H., & Li, L. (2009). Image mosaic based on surf feature matching. In *2009 First International Conference on Information Science and Engineering* (pp. 1287–1290). IEEE.

HumiSeal. (2017). What is conformal coating? | HumiSeal®. Retrieved April 22, 2017, from http://www.humiseal.com/conformal-coating/

Kennedy, J. (2011). Particle swarm optimization. In *Encyclopedia of machine learning* (pp. 760–766). Springer.

Koch, T., Zhuo, X., Reinartz, P., & Fraundorfer, F. (2016). A new paradigm for matching uav-and aerial images. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, 2016*, *3*, 83–90.

Lei, F., & Wang, W. (2014). A fast method for image mosaic based on SURF. In *2014 9th IEEE Conference on Industrial Electronics and Applications* (pp. 79–82). IEEE.

Liu, M., & Wen, D. (2017). Automatic seamless image mosaic method based on SIFT features. In C. Zhang & A. Asundi (Eds.) (p. 1025636). https://doi.org/10.1117/12.2257792

Long, X. L., Chen, Q., & Bao, J. W. (2013). Improvement of image mosaic algorithm based on SURF. *Applied Mechanics and Materials*. Trans Tech Publ.

Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on* (Vol. 2, pp. 1150–1157). Ieee.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110.

Martínez, M. (2014). *Semáforos inteligentes*. Asunción. Retrieved from http://jeuazarru.com/wp-content/uploads/2014/10/semaforos_inteligentes.pdf

Matas, J., Chum, O., Urban, M., & Pajdla, T. (2004). Robust wide-baseline stereo from maximally stable extremal regions. *Image and Vision Computing*, *22*(10), 761–767. https://doi.org/10.1016/j.imavis.2004.02.006

MathWorks Simulink Team. (2014). Simulink Support Package for Arduino Due Hardware - File Exchange - MATLAB Central. Retrieved March 23, 2017, from http://www.mathworks.com/matlabcentral/fileexchange/45071-simulink-support-package-for-arduino-due-hardware

Mery, D. (2015). BALU: A Matlab Toolbox for Computer Vision, Pattern Recognition and Image Processing.

Ming-Kuei Hu. (1962). Visual pattern recognition by moment invariants. *IEEE Transactions on Information Theory*, *8*(2), 179–187. https://doi.org/10.1109/TIT.1962.1057692

Molina, R. (1998). *Introducción al procesamiento y análisis de imágenes digitales*.

Morel, J.-M., & Yu, G. (2009). ASIFT: A new framework for fully affine invariant image comparison. *SIAM Journal on Imaging Sciences*, *2*(2), 438–469.

Nora, D., Serna Palomino, L., Ulises, L., & Concha, R. (2009). Técnicas de Segmentación en Procesamiento Digital de Imágenes. Retrieved from http://sisbib.unmsm.edu.pe/bibvirtual/publicaciones/risi/2009_n2/v6n2/a02v6n2.pdf

Rosten, E., & Drummond, T. (2006). Machine Learning for High-Speed Corner Detection (pp. 430–443). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11744023_34

Rublee, E., Rabaud, V., Konolige, K., & Bradski, G. (2011). ORB: An efficient alternative to SIFT or SURF. In *2011 International conference on computer vision* (pp. 2564–2571). IEEE.

Ruiz, Á., Sánchez, K., & Arteaga, L. (2016). Modelos de Ordenación del Tráfico de la Ciudad de Loja. *IV Congreso REDU*, 6.

Santamaría, M. V. B., & Moscol, M. F. R. (2015). Semáforos Inteligentes para la Regulación Del Tráfico Vehicular. *Revista Científica INGENIERÍA: Ciencia, Tecnología E Innovación*, *1*(1), 37.

Shi, G., Xu, X., & Dai, Y. (2013). SIFT feature point matching based on improved RANSAC Algorithm. In *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 2013 5th International Conference on* (Vol. 1, pp. 474–477). IEEE.

Stanković, R. S., & Falkowski, B. J. (2003). The Haar wavelet transform: its status and achievements. *Computers & Electrical Engineering*, *29*(1), 25–44.

Stewart, C. V. (1995). MINPRAN: a new robust estimator for computer vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *17*(10), 925–938. https://doi.org/10.1109/34.464558

Su, J., & Ai, M. (2011). Unmanned Airship Based Multiple Spectrum Image Mosaic with SIFT Feature Matching. In *Multi-Platform/Multi-Sensor Remote Sensing and Mapping (M2RSM), 2011 International Workshop on* (pp. 1–5). IEEE.

Sun, S., & Zeng, Z. (2013). UAV image mosaic based on adaptive SIFT algorithm. In *2013 21st International Conference on Geoinformatics* (pp. 1–6). IEEE.

Sun, Y., Zhao, L., Huang, S., Yan, L., & Dissanayake, G. (2014). L2-SIFT: SIFT feature extraction and matching for large images in large-scale aerial photogrammetry. *ISPRS Journal of Photogrammetry and Remote Sensing*, *91*, 1–16.

Szeliski, R. (1996). Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, *16*(2), 22–30.

Torr, P. H. S., & Zisserman, A. (2000). MLESAC: A new robust estimator with application to

estimating image geometry. *Computer Vision and Image Understanding*, *78*(1), 138–156.

Vedaldi, A., & Fulkerson, B. (2010). VLFeat: An open and portable library of computer vision algorithms. In *Proceedings of the 18th ACM international conference on Multimedia* (pp. 1469–1472). ACM.

Viola, P., & Jones, M. (2001). Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on* (Vol. 1, p. I-511-I-518 vol. 1). IEEE.

Wang, J., & Watada, J. (2015). Panoramic image mosaic based on SURF algorithm using OpenCV. In *Intelligent Signal Processing (WISP), 2015 IEEE 9th International Symposium on* (pp. 1–6). IEEE.

Wen, W. (2008). A dynamic and automatic traffic light control expert system for solving the road congestion problem. *Expert Systems with Applications*, *34*(4), 2370–2381.

Wu, C. (2007). A GPU Implementation of Scale Invariant Feature Transform (SIFT).

Xu, Y., Yu, G., Wu, X., Wang, Y., & Ma, Y. (2016). An Enhanced Viola-Jones Vehicle Detection Method From Unmanned Aerial Vehicles Imagery. *IEEE Transactions on Intelligent Transportation Systems*.

Yang, Z.-L., & Guo, B.-L. (2008). Image mosaic based on SIFT. In *Intelligent Information Hiding and Multimedia Signal Processing, 2008. IIHMSP'08 International Conference on* (pp. 1422–1425). IEEE.

Yousef, K., Al-Karaki, M., & Shatnawi, A. (2010). Intelligent traffic light flow control system using wireless sensors networks.

Zehang Sun, Bebis, G., & Miller, R. (2006). Monocular precrash vehicle detection: features and classifiers. *IEEE Transactions on Image Processing*, *15*(7), 2019–2034. https://doi.org/10.1109/TIP.2006.877062

Zhao, T., & Nevatia, R. (2003). Car detection in low resolution aerial images. *Image and Vision Computing*, *21*(8), 693–703.

Zhou, B., Cao, J., Zeng, X., & Wu, H. (2010). Adaptive traffic light control in wireless sensor network-based intelligent transportation system. In *Vehicular technology conference fall (VTC 2010-Fall), 2010 IEEE 72nd* (pp. 1–5). IEEE.

ANEXOS

ANEXO 1. CÓDIGO PARA LA CREACIÓN DE VISTAS PANORÁMICAS TIPO “MOSAICO”

```
%% CÓDIGO PARA LA CREACIÓN DE VISTAS PANORÁMICAS TIPO "MOSAICOS" %%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Autor: Daniel F. Aguirre R.
%% 08/09/2012
%% ------------------------------------------------------------------------
%% CARGA BALU Y VLFEAT - LIMPIO REGISTROS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clc
clear all
close all
warning off

%% CARGA DE IMÁGENES, CAMBIO DE TAMAÑO Y FORMATO A SINGLE %%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ic = imread('DJI_0050.jpg');          % Carga de imágenes
Icd1 = imread('DJI_0051.jpg');        % Carga de imágenes
Ici1 = imread('DJI_0053.jpg');        % Carga de imágenes
Icd2 = imread('DJI_0052.jpg');        % Carga de imágenes

G1 = Ic(:,:,3);                       %Descomposición de la imagen en canales de colores
G2 = Icd1(:,:,3);
G3 = Ici1(:,:,3);
G4 = Icd2(:,:,3);

Ic_g = single(imresize(G1,0.3));      % Cambio a formato single
Icd1_g = single(imresize(G2,0.3));    % Cambio a formato single
Ici1_g = single(imresize(G3,0.3));    % Cambio a formato single
Icd2_g = single(imresize(G4,0.3));    % Cambio a formato single

%% PRIMERA CORRESPONDENCIA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Cargo imagen 1
figure(1),
subplot(2,2,1), imshow(Ic_g,[])
title('Primera Imagen');
hold on
[fc,dc] = vl_sift(Ic_g);  % Calculo puntos SIFT y descriptores

% La matriz fc tiene una columna por cada punto.
% Cada punto tiene la información X,Y en fc(1:2), escala fc(3) y orientación fc(4).

perm_c = randperm(size(fc,2));           % Posicionamiento aleatorio de los puntos SIFT
sel_c = perm_c(1:50);                    % Escojo los 100 valores aleatorios
h1_c = vl_plotframe(fc(:,sel_c));        % Grafica los posiciones de los puntos SIFT
h2_c = vl_plotframe(fc(:,sel_c));        % Grafica los posiciones de los puntos SIFT
set(h1_c,'color','r','linewidth',3);     % Grafica color línea externa del punto SIFT
set(h2_c,'color','y','linewidth',2);     % Grafica color interno del punto SIFT
h3_c = vl_plotsiftdescriptor(dc(:,sel_c),fc(:,sel_c));  % Superpongo descriptores
set(h3_c,'color','g');                   % Establezco color del descriptor
hold off

% Cargo imagen 2 con el mismo procedimiento
figure(1),
subplot(2,2,2),imshow(Icd1_g,[])
title('Segunda Imagen');
hold on
[fcd1,dcd1] = vl_sift(Icd1_g);
perm_cd1 = randperm(size(fcd1,2));
sel_cd1 = perm_cd1(1:50);
h1_cd1 = vl_plotframe(fcd1(:,sel_cd1));
h2_cd1 = vl_plotframe(fcd1(:,sel_cd1));
set(h1_cd1,'color','r','linewidth',3);
set(h2_cd1,'color','y','linewidth',2);
h3_cd1 = vl_plotsiftdescriptor(dcd1(:,sel_cd1),fcd1(:,sel_cd1));
set(h3_cd1,'color','g');
hold off

%% Visualización correspondencias entre pares de imágenes %%%%%%%%%%%%%%%%%
% Presenta las imágenes en pares para establecer la correspondencia
dosimagenes_c_cd1 = [Ic_g,Icd1_g];
[matches_c_cd1, scores_c_cd1] = vl_ubcmatch(dc,dcd1,10);     % Imagen centro y primera
figure(1), subplot(2,2,[3,4]),imshow(dosimagenes_c_cd1,[]);
```

```
o = size(Ic_g,2);                               % Longitud en eje x de la imagen
a = length(matches_c_cd1(1,:));                 % Número de correspondencias
title(sprintf('Hay %d puntos SIFT entre ambas imágenes', a));
X1 = zeros(1,a);                                % posiciones en X de los puntos en c
X2 = zeros(1,a);                                % posiciones en X de los puntos en cd1
Y1 = zeros(1,a);                                % posiciones en Y de los puntos en c
Y2 = zeros(1,a);                                % posiciones en Y de los puntos en cd1
for i = 1:a                                     % Traza las rectas de correspondencia
    X1(1,i) = fc(1,matches_c_cd1(1,i));         % asigna los valores encontrados en fc a X
    X2(1,i) = fcd1(1,matches_c_cd1(2,i))+o;     % asigna los valores encontrados en fcd1 a X
    Y1(1,i) = fc(2,matches_c_cd1(1,i));         % asigna los valores encontrados en fc a X
    Y2(1,i) = fcd1(2,matches_c_cd1(2,i));       % asigna los valores encontrados en fcd1 a Y
end
line([X1;X2],[Y1;Y2]);                          % Traza líneas

% La correspondencia original y su descriptor más cercano es almacenado en
% cada columna de matches y la distancia entre el par de puntos en scores.
X1 = fc(1:2,matches_c_cd1(1,:));X1(3,:) = 1;    % Datos de puntos correspondientes
X2 = fcd1(1:2,matches_c_cd1(2,:));X2(3,:) = 1;  % Datos de puntos correspondientes

%% Encuentro matriz H
H = Bmv_homographyRANSAC(X1,X2);        % Estimación de H
H = H/H(3,3)                            % Normalización
X2s = H*X1;X2s = X2s./(ones(3,1)*X2s(3,:));
e = X2s-X2;
e1 = max(max(e))                        % Máximo error en pixeles.

%% Aplicación de H sobre la imagen
imagen1 = Ic_g;                         % Cargo imagen 1
imagen2 = Icd1_g;                       % Cargo imagen 2

% Establezco los cuatro puntos de la imagen a aplicarse la homografía.
[f2,c2] = size(imagen2);
esquinas2 = [1 c2 c2  1;
             1  1 f2 f2;
             1  1  1  1 ];

% Encuentro los puntos de las esquinas de la imagen 2 en el dominio de la imagen 1.
esquinas2_1 = inv(H)*esquinas2; %#ok<MINV>

% Normalizo los resultados estableciendo 1 en la tercera fila
esquinas2_1(1,:) = esquinas2_1(1,:)./esquinas2_1(3,:);
esquinas2_1(2,:) = esquinas2_1(2,:)./esquinas2_1(3,:);

% Tamaño del nuevo lienzo. Establezco límites mínimos y máximos
% Establezco el menor y el mayor valor de los puntos encontrados en x
[f1,c1] = size(imagen1);
[im1_u,im1_v] = meshgrid(1:c1,1:f1);                    % malla para imagen 1

% u_size y v_size definen el lienzo final donde se fusionan ambas imágenes
u_size = min([1 esquinas2_1(1,:)]):max([c1 esquinas2_1(1,:)]);
v_size = min([1 esquinas2_1(2,:)]):max([f1 esquinas2_1(2,:)]);

% Establezco el lienzo donde se interpola imagen 1
[u,v] = meshgrid(u_size,v_size);
im1_int = interp2(im1_u,im1_v,imagen1,u,v,'bicubic');
[im2_u,im2_v] = meshgrid(1:c2,1:f2);                    % malla para imagen 2

% Establezco el lienzo donde se interpola imagen 2
% Busco la rejilla transformada para la imagen 2
u_size_2 = H(1,1)*u + H(1,2)*v + H(1,3);
v_size_2 = H(2,1)*u + H(2,2)*v + H(2,3);
w_size_2 = H(3,1)*u + H(3,2)*v + H(3,3);

% Normalizo a coordenadas homogéneas
u_size_2 = u_size_2./w_size_2;
v_size_2 = v_size_2./w_size_2;
im2_int = interp2(im2_u,im2_v,imagen2,u_size_2,v_size_2,'bicubic');

% Establezco la mitad del valor donde se superponen ambas imágenes
L1 = superponer(im1_int,im2_int);
figure(2)
imshow(L1), title('Primera Correspondencia');
```

```
%% SEGUNDA CORRESPONDENCIA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Corresponde a la fusión entre la primera correspondencia y la primera
% imagen desde el centro hacia la izquierda
Ib = single(L1);
figure(3),
subplot(2,2,1),imshow(Ib,[])
title('Primera Correspondencia');
hold on
[fb,db] = vl_sift(Ib,'PeakThresh', 0);
perm_b = randperm(size(fb,2));
sel_b = perm_b(1:50);
h1_b = vl_plotframe(fb(:,sel_b));
h2_b = vl_plotframe(fb(:,sel_b));
set(h1_b,'color','r','linewidth',3);
set(h2_b,'color','y','linewidth',2);
h3_b = vl_plotsiftdescriptor(db(:,sel_b),fb(:,sel_b));
set(h3_b,'color','g');
hold off

figure(3),
subplot(2,2,2),imshow(Ici1_g,[])
title('Tercera Imagen');
hold on
[fci1,dci1] = vl_sift(Ici1_g,'PeakThresh', 0);
perm_ci1 = randperm(size(fci1,2));
sel_ci1 = perm_ci1(1:50);
h1_ci1 = vl_plotframe(fci1(:,sel_ci1));
h2_ci1 = vl_plotframe(fci1(:,sel_ci1));
set(h1_ci1,'color','r','linewidth',3);
set(h2_ci1,'color','y','linewidth',2);
h3_ci1 = vl_plotsiftdescriptor(dci1(:,sel_ci1),fci1(:,sel_ci1));
set(h3_ci1,'color','g');
hold off

%% Visualización correspondencias entre pares de imágenes
% Presenta las imágenes en pares para establecer la correspondencia
dosimagenes_ci1_c = [Ici1_g,Ic_g];
[matches_ci1_c, scores_ci1_c] = vl_ubcmatch(dci1,dc,10);        % izquierda 1 y centro
figure(3), subplot(2,2,[3,4]),imshow(dosimagenes_ci1_c,[]);
a3 = length(matches_ci1_c(1,:));           % número de correspondencias
title(sprintf('Hay %d puntos SIFT entre ambas imágenes', a3));
X13 = zeros(1,a3);                         % posiciones en X de los puntos en c
X23 = zeros(1,a3);                         % posiciones en X de los puntos en cd1
Y13 = zeros(1,a3);                         % posiciones en Y de los puntos en c
Y23 = zeros(1,a3);                         % posiciones en Y de los puntos en cd1
for i = 1:a3                               % Traza las rectas de correspondencia
    X13(1,i) = fci1(1,matches_ci1_c(1,i)); % asigna los valores encontrados en fc a X
    X23(1,i) = fc(1,matches_ci1_c(2,i))+o; % asigna los valores encontrados en fcd1 a X
    Y13(1,i) = fci1(2,matches_ci1_c(1,i)); % asigna los valores enocntrados en fc a X
    Y23(1,i) = fc(2,matches_ci1_c(2,i));   % asigna los valores enocntrados en fcd1 a Y
end
line([X13;X23],[Y13;Y23]);
[matches_b_ci1, scores_b_ci1] = vl_ubcmatch(db,dci1,10);
X1 = fb(1:2,matches_b_ci1(1,:));X1(3,:) = 1;
X2 = fci1(1:2,matches_b_ci1(2,:));X2(3,:) = 1;

%% Encuentro matriz H
H = Bmv_homographyRANSAC(X1,X2);           % Estimación de H
H = H/H(3,3)                               % Normalización
X2s = H*X1;X2s = X2s./(ones(3,1)*X2s(3,:));
e = X2s-X2;
e2 = max(max(e))                           % Máximo error en pixeles.

%% Aplicación de H sobre la imagen
imagen1 = Ib;
imagen2 = Ici1_g;

% Establezco los cuatro puntos de la imagen a aplicarse la homografía.
[f2,c2] = size(imagen2);
esquinas2 = [1 c2 c2  1;
             1  1 f2 f2;
             1  1  1  1 ];
% Encuentro los puntos de las esquinas de la imagen 2 en el dominio de la
% imagen 1.
esquinas2_1 = inv(H)*esquinas2;         %#ok<MINV>
```

```
% Normalizo los resultados estableciendo 1 en la tercera fila (coordenadas
% homogéneas)
esquinas2_1(1,:) = esquinas2_1(1,:)./esquinas2_1(3,:);
esquinas2_1(2,:) = esquinas2_1(2,:)./esquinas2_1(3,:);

% Tamaño del nuevo lienzo. Establezco límites mínimos y máximos
% Establezco el menor y el mayor valor de los puntos encontrados
[f1,c1] = size(imagen1);
[im1_u,im1_v] = meshgrid(1:c1,1:f1); % malla para imagen 1

% u_size y v_size definen el lienzo final donde se fusionan ambas imágenes
% define de la misma forma pero para los puntos y.
u_size = min([1 esquinas2_1(1,:)]):max([c1 esquinas2_1(1,:)]);
v_size = min([1 esquinas2_1(2,:)]):max([f1 esquinas2_1(2,:)]);

% Establezco el lienzo donde se interpola imagen 1
[u,v] = meshgrid(u_size,v_size) ;
im1_int = interp2(im1_u,im1_v,imagen1,u,v,'bicubic');
[im2_u,im2_v] = meshgrid(1:c2,1:f2);                    % malla para imagen 2

% Establezco el lienzo donde se interpola imagen 2
% Busco la rejilla transformada para la imagen 2
u_size_2 = H(1,1)*u + H(1,2)*v + H(1,3);
v_size_2 = H(2,1)*u + H(2,2)*v + H(2,3);
w_size_2 = H(3,1)*u + H(3,2)*v + H(3,3);

% Normalizo a coordenadas homogéneas
u_size_2 = u_size_2./w_size_2;
v_size_2 = v_size_2./w_size_2;
im2_int = interp2(im2_u,im2_v,imagen2,u_size_2,v_size_2,'bicubic');

% Establezco la mitad del valor donde se superponen ambas imágenes
L1 = superponer(im1_int,im2_int);
figure(4)
imshow(L1),title('Segunda Correspondencia');

%% TERCERA CORRESPONDENCIA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Fusión entre la segunda correspondencia y la segunda imagen
Ib1 = single(L1);
figure(5),
subplot(2,2,1),imshow(Ib1,[])
title('Segunda Correspondencia');
hold on
[fb1,db1] = vl_sift(Ib1,'PeakThresh', 0);
perm_b1 = randperm(size(fb1,2));
sel_b1 = perm_b1(1:50);
h1_b1 = vl_plotframe(fb1(:,sel_b1));
h2_b1 = vl_plotframe(fb1(:,sel_b1));
set(h1_b1,'color','r','linewidth',3);
set(h2_b1,'color','y','linewidth',2);
h3_b1 = vl_plotsiftdescriptor(db1(:,sel_b1),fb1(:,sel_b1));
set(h3_b1,'color','g');
hold off

figure(5),
subplot(2,2,2),imshow(Icd2_g,[])
title('Segunda Imagen desde el centro hacia la derecha');
hold on
[fcd2,dcd2] = vl_sift(Icd2_g,'PeakThresh', 0);
perm_cd2 = randperm(size(fcd2,2));
sel_cd2 = perm_cd2(1:50);
h1_cd2 = vl_plotframe(fcd2(:,sel_cd2));
h2_cd2 = vl_plotframe(fcd2(:,sel_cd2));
set(h1_cd2,'color','r','linewidth',3);
set(h2_cd2,'color','y','linewidth',2);
h3_cd2 = vl_plotsiftdescriptor(dcd2(:,sel_cd2),fcd2(:,sel_cd2));
set(h3_cd2,'color','g');
hold off

%% Visualización correspondencias entre pares de imágenes
% Presenta las imágenes en pares para establecer la correspondencia
dosimagenes_cd1_cd2 = [Icd1_g,Icd2_g];

[matches_cd1_cd2, scores_cd1_cd2] = vl_ubcmatch(dcd1,dcd2,10);  % derecha 1 y derecha 2
```

```
figure(5), subplot(2,2,[3,4]),imshow(dosimagenes_cd1_cd2,[]);
a1 = length(matches_cd1_cd2(1,:));              % número de correspondencias
title(sprintf('Hay %d puntos SIFT entre ambas imágenes', a1));
X11 = zeros(1,a1);                              % posiciones en X de los puntos en c
X21 = zeros(1,a1);                              % posiciones en X de los puntos en cd1
Y11 = zeros(1,a1);                              % posiciones en Y de los puntos en c
Y21 = zeros(1,a1);                              % posiciones en Y de los puntos en cd1
for i = 1:a1                                    % Traza las rectas de correspondencia
    X11(1,i) = fcd1(1,matches_cd1_cd2(1,i));  %asigna valores encontrados en fc a X
    X21(1,i) = fcd2(1,matches_cd1_cd2(2,i))+o;%asigna valores encontrados en fcd1 a X
    Y11(1,i) = fcd1(2,matches_cd1_cd2(1,i));  % asigna los valores enocntrados en fc a X
    Y21(1,i) = fcd2(2,matches_cd1_cd2(2,i));  % asigna valores enocntrados en fcd1 a Y
end
line([X11;X21],[Y11;Y21]);
[matches_b1_cd2, scores_b1_cd2] = vl_ubcmatch(db1,dcd2,10);
X1 = fb1(1:2,matches_b1_cd2(1,:));X1(3,:) = 1;
X2 = fcd2(1:2,matches_b1_cd2(2,:));X2(3,:) = 1;

%% Encuentro matriz H
H = Bmv_homographyRANSAC(X1,X2);        % Estimación de H
H = H/H(3,3)                            % Normalización
X2s = H*X1;X2s = X2s./(ones(3,1)*X2s(3,:));
e = X2s-X2;
e3 = max(max(e))                        % Máximo error en pixeles.

%% Aplicación de H sobre la imagen
imagen1 = Ib1;
imagen2 = Icd2_g;

% Establezco los cuatro puntos de la imagen a aplicarse la homografía.
[f2,c2] = size(imagen2);
esquinas2 = [1 c2 c2  1;
             1  1 f2 f2;
             1  1  1  1 ];

% Encuentro los puntos de las esquinas de la imagen 2 en el dominio de la
% imagen 1.
esquinas2_1 = inv(H)*esquinas2;                             %#ok<MINV>

% Normalizo los resultados estableciendo 1 en la tercera fila (coordenadas
% homogéneas)
esquinas2_1(1,:) = esquinas2_1(1,:)./esquinas2_1(3,:);
esquinas2_1(2,:) = esquinas2_1(2,:)./esquinas2_1(3,:);

% Tamaño del nuevo lienzo. Establezco límites mínimos y máximos
% Establezco el menor y el mayor valor de los puntos encontrados
[f1,c1] = size(imagen1);
[im1_u,im1_v] = meshgrid(1:c1,1:f1);                        % malla para imagen 1

% u_size y v_size definen el lienzo final donde se fusionan ambas imágenes
u_size = min([1 esquinas2_1(1,:)]):max([c1 esquinas2_1(1,:)]);
v_size = min([1 esquinas2_1(2,:)]):max([f1 esquinas2_1(2,:)]);

% Establezco el lienzo donde se interpola imagen 1
[u,v] = meshgrid(u_size,v_size);
im1_int = interp2(im1_u,im1_v,imagen1,u,v,'bicubic');

[im2_u,im2_v] = meshgrid(1:c2,1:f2);                        % malla para imagen 2

% Establezco el lienzo donde se interpola imagen 2
% Busco la rejilla transformada para la imagen 2
u_size_2 = H(1,1)*u + H(1,2)*v + H(1,3);
v_size_2 = H(2,1)*u + H(2,2)*v + H(2,3);
w_size_2 = H(3,1)*u + H(3,2)*v + H(3,3);

% Normalizo a coordenadas homogéneas
u_size_2 = u_size_2./w_size_2;
v_size_2 = v_size_2./w_size_2;
im2_int = interp2(im2_u,im2_v,imagen2,u_size_2,v_size_2,'bicubic');

% Establezco la mitad del valor donde se superponen ambas imágenes
mosaico = superponer(im1_int,im2_int);
figure(6)
imshow(mosaico),title('Tercera Correspondencia');
```

ANEXO 2. CÓDIGO PARA LA IMPLEMENTACIÓN DE LA SOLUCIÓN PARA EL CONTEO VEHICULAR A PARTIR DE IMÁGENES AÉREAS

```
%% CÓDIGO PARA LA IMPLEMENTACIÓN DE LA SOLUCIÓN PARA EL CONTEO VEHICULAR A PARTIR DE IMÁGENES
%% AÉREAS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Autor: Cristian F. Pérez S.
%% 17/03/2017
%% Referenciass:
%% "Código Para La Creación De Vistas Panorámicas Tipo "Mosaico" - Aguirre Daniel (2012)
%% "Shape recognition using 2-D correlation" - Barragán Diego (2015)
%% ------------------------------------------------------------------------------------------

function varargout = CONTEOVEHICULAR_V2(varargin)
% Edit the above text to modify the response to help CONTEOVEHICULAR_V2
% Last Modified by GUIDE v2.5 05-Mar-2017 23:36:50
% Begin initialization code - DO NOT EDIT

gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @CONTEOVEHICULAR_V2_OpeningFcn, ...
                   'gui_OutputFcn',  @CONTEOVEHICULAR_V2_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT


% --- Executes just before CONTEOVEHICULAR_V2 is made visible.
function CONTEOVEHICULAR_V2_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to CONTEOVEHICULAR_V2 (see VARARGIN)

% Choose default command line output for CONTEOVEHICULAR_V2
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes CONTEOVEHICULAR_V2 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = CONTEOVEHICULAR_V2_OutputFcn(hObject, eventdata, handles)

varargout{1} = handles.output;
warning off

% --- Executes when selected object is changed in uipanel1.
function uipanel1_SelectionChangeFcn(hObject, eventdata, handles)
tic
%% Mostrar mapa sin zoom (En caso de que se seleccione por segunda vez)
axes(handles.mosaico)
zoom out
zoom off
%% Resetear textfield y pop-up menú
set([handles.salida1 handles.salida5],'String','(Calle Principal)','BackgroundColor',[0.94
0.94 0.94]);
set([handles.salida2 handles.salida6],'String','(Calle Secundaria)','BackgroundColor',[0.94
0.94 0.94]);
set([handles.salida3 handles.salida7],'String','','BackgroundColor',[0.94 0.94 0.94]);
set([handles.salida4 handles.salida8],'String','','BackgroundColor',[0.94 0.94 0.94]);
set(handles.salida9,'String','(Calle Recomendada)');
set([handles.text_inicio handles.text_fin],'String','','BackgroundColor',[0.94 0.94 0.94]);
```

```
set(handles.salida_proc,'String','','BackgroundColor',[0.94 0.94 0.94]);
set(handles.popupmenu1, 'Value', 1);
set(handles.popupmenu2, 'Value', 1);
set(handles.popupmenu3, 'Value', 1);

%% SELECCIONAR MAPA (CADA MAPA CONTIENE UN ESCENARIO DISTINTO) %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if hObject == handles.mapa1
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaVacioVacio.mat;                        %Cargo Mapa
    MapaLoja = MapaLojaVacioVacio;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 1)');

elseif hObject == handles.mapa2
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaVacio.mat;                             %Cargo Mapa
    MapaLoja = MapaLojaVacio;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 2)');

elseif hObject == handles.mapa3
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaCentroVacio.mat;                       %Cargo Mapa
    MapaLoja = MapaLojaCentroVacio;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 3)');

elseif hObject == handles.mapa4
    hold off
    cla;                                                %Limpia el axis
    load MapaLoja.mat;                                  %Carga mapa
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 4)');

elseif hObject == handles.mapa5
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaFull.mat;                              %Carga mapa
    MapaLoja = MapaLojaFull;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 5)');

elseif hObject == handles.mapa6
    hold off
    cla;                                                 %Limpia el axis
    load MapaLojaCentro.mat;                             %Cargo Mapa
    MapaLoja = MapaLojaCentro;
    axes(handles.mosaico);                               %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 6)');

elseif hObject == handles.mapa7
    hold off
    cla;                                                 %Limpia el axis
    load MapaLojaFullFull.mat;                           %Carga Mapa
    MapaLoja = MapaLojaFullFull;
    axes(handles.mosaico);                               %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 7)');

elseif hObject == handles.mapa8
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaFullFullFull.mat;                      %Cargo Mapa
    MapaLoja = MapaLojaFullFullFull;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 8)');
```

```
elseif hObject == handles.mapa9
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaCasiLleno.mat;                         %Cargo Mapa
    MapaLoja = MapaLojaCasiLleno;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 9)');

elseif hObject == handles.mapa10
    hold off
    cla;                                                %Limpia el axis
    load MapaLojaLlenoTotal.mat;                        %Cargo Mapa
    MapaLoja = MapaLojaLlenoTotal;
    axes(handles.mosaico);                              %Presenta el mapa cargado en el axis
    imshow(MapaLoja,'initialmagnification','fit'), title('CASCO URBANO DE LA CIUDAD DE LOJA
(Escenario 10)');

end

handles.mapa = MapaLoja;                                %Guarda la variable MapaLoja
guidata(hObject,handles);                               %Guarda información del axis

%% Presentar Calles
if size(MapaLoja) > 1
   nombres_calles(MapaLoja);
end

%% Barra de progreso
progreso_fig =
sbprogress([],0,'Msg','Procesando...','Parent',CONTEOVEHICULAR_V2,'DisplayMode',2);
%% Cargo imagen de semáforos
semaforoimagen = imread('semaforo.jpg');
for i = 1:30
    axis_semaforo = sprintf('semaforo%d',i);
    axes(handles.(axis_semaforo)),imshow(semaforoimagen,'initialmagnification','fit');
end
handles.img_semaforo = semaforoimagen;
axis off
pause(4)
axes(handles.mosaico);
%% Barra de progreso
progreso_fig = sbprogress(progreso_fig,2);

%% Desentrelazador
hdint = vision.Deinterlacer;
% Aplico el desentrelazador mediante el método "step"
MapaLoja = step(hdint,MapaLoja);

%% PRE PROCESAR Y PROCESAR LA IMAGEN %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
imshow(MapaLoja,'initialmagnification','fit');
PreProcesada_mapa = prepocesar_mex(MapaLoja);
background = imopen(PreProcesada_mapa,strel('disk',25));
background = imadjust(background,[],[],0.4);
Resta = PreProcesada_mapa - background;
Procesada = procesar(Resta);
Procesada = gpuArray(Procesada);
clearvars PreProcesada_mapa background Resta
hold off

%% CONTEO DE VEHICULOS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Semáforo 1 %%%
progreso_fig = sbprogress(progreso_fig,5,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Juan José Peña');                   %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');             %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit');                    %Presenta mapa
hold on
zoom_semaforo1 = [233 3400 99 2546];                              %Región para acercamiento
axis(zoom_semaforo1);                                             %Región acercada
text(1986,1292,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(1795,973,'Juan José Peña','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
```

```
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit');title('Mapa procesado');%Mapa procesado
hold on
%% Calle 1
ROI_calle1 = impoly(gca,[1771,1190; 1785,190; 1863,190; 1850,1190]);%ROI calle 1
[total_carros1,propied1] = contabilizarGPU(ROI_calle1,Procesada);   %Conteo de vehículos
%% Calle 14
ROI_calle14 = impoly(gca,[1783,1390; 1861,1390; 1840,2340; 1771,2340]);%ROI tercera calle
x = [1783 1861 1840 1771 1783];                                 %Coordenadas del punto el eje x
y = [1390 1390 2340 2340 1390];                                 %Coordenadas del punto el eje y
area_calle14 = area_calc_mex(x,y);                              %Cálculo de área calle 14
[total_carros14,propied14] = contabilizarGPU(ROI_calle14,Procesada);%Conteo de vehículos
%% Calle 7
ROI_calle7 = impoly(gca,[1730,1340; 300,1340; 300,1250; 1730,1250]); %ROI calle 7
x = [1730 300 300 1730 1730];                                   %Coordenadas del punto el eje x
y = [1340 1340 1250 1250 1340];                                 %Coordenadas del punto el eje y
area_calle7 = area_calc_mex(x,y);                               %Cálculo de área calle 7
[total_carros7,propied7] = contabilizarGPU(ROI_calle7,Procesada);  %Conteo de vehículos
clearvars zoom_semaforo1
hold off

%% Semáforo 2 %%%
progreso_fig = sbprogress(progreso_fig,8,'DisplayMode',2);          %Barra de progreso
set(handles.salida1,'String','José Joaquín de Olmedo');            %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');              %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit');                      %Presenta mapa
hold on
zoom_semaforo2 = [1864 5015 99 2466];                               %Región para acercamiento
axis(zoom_semaforo2);                                               %Región acercada
text(3616,1291,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(3423,2331,'J. J. de Olmedo','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
hold off
pause(0.001);
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'),title('Mapa procesado');%Mapa procesado
hold on
%% Calle 2
ROI_calle2 = impoly(gca,[3387,1140; 3407,190; 3485,190; 3460,1140]);%ROI calle 2
x = [3387 3407 3485 3460 3387];                                 %Coordenadas del punto el eje x
y = [1140 190 190 1140 1140];                                   %Coordenadas del punto el eje y
area_calle2 = area_calc_mex(x,y);                               %Cálculo de área calle 2
[total_carros2,propied2] = contabilizarGPU(ROI_calle2,Procesada);  %Conteo de vehículos
%% Calle 8
ROI_calle8 = impoly(gca,[1930,1250; 3340,1250; 3340,1340; 1930,1340]);  %ROI calle 8
x = [1930 3340 3340 1930 1930];                                 %Coordenadas del punto el eje x
y = [1250 1250 1340 1340 1250];                                 %Coordenadas del punto el eje y
area_calle8 = area_calc_mex(x,y);                               %Cálculo de área calle 8
[total_carros8,propied8] = contabilizarGPU(ROI_calle8,Procesada);  %Conteo de vehículos
clearvars zoom_semaforo2
hold off

%% Semáforo 3 %%
progreso_fig = sbprogress(progreso_fig,11,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','Bernardo Valdivieso');               %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');              %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit');                      %Presenta mapa
hold on
zoom_semaforo3 = [3463 6646 83 2466];                               %Región para acercamiento
axis(zoom_semaforo3);                                                       %Región acercada
text(3618,1279,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(5050,1121,'B. Valdivieso','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001);
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'),title('Mapa procesado');%Mapa procesado
hold on
%% Calle 3
ROI_calle3 = impoly(gca,[5010,1140; 5025,185; 5105,185; 5080,1140]);%ROI calle 3
[total_carros3,propied3] = contabilizarGPU(ROI_calle3,Procesada);   %Cálculo de área calle 3
clearvars zoom_semaforo3
%% Calle 16
ROI_calle16 = impoly(gca,[5025,1388; 5005,2335; 5045,2335; 5068,1388]); %ROI calle 16
x = [5025 5005 5045 5068 5025];                                 %Coordenadas del punto el eje x
y = [1388 2335 2335 1388 1388];                                 %Coordenadas del punto el eje y
```

```
area_calle16 = area_calc_mex(x,y);                              %Cálculo de área calle 16
[total_carros16,propied16] = contabilizarGPU(ROI_calle16,Procesada);    %Conteo de vehículos
%% Calle 9
ROI_calle9 = impoly(gca,[3540,1240; 4960,1250; 4960,1340; 3540,1340]);  %ROI calle 9
x = [3540 4960 4960 3540 3540];                                 %Coordenadas del punto el eje x
y = [1240 1250 1340 1340 1240];                                 %Coordenadas del punto el eje y
area_calle9 = area_calc_mex(x,y);                               %Cálculo de área calle 9
[total_carros9,propied9] = contabilizarGPU(ROI_calle9,Procesada);       %Conteo de vehículos
hold off

%% Semáforo 4 %%
progreso_fig = sbprogress(progreso_fig,14,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                      %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');               %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit');                      %Presenta mapa
hold on
zoom_semaforo4 = [5129 8254 107 2482];                              %Región para acercamiento
axis(zoom_semaforo4);                                               %Región acercada
text(5376,1279,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(6685,2345,'Simón Bolívar','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado'); %Mapa procesado
hold on
%% Calle 4
ROI_calle4 = impoly(gca,[6630,1150; 6645,190; 6720,190; 6700,1150]);     %ROI calle 4
x = [6630 6645 6720 6700 6630];                                 %Coordenadas del punto el eje x
y = [1150 190 190 1150 1150];                                   %Coordenadas del punto el eje y
area_calle4 = area_calc_mex(x,y);                               %Cálculo de área calle 4
[total_carros4,propied4] = contabilizarGPU(ROI_calle4,Procesada);  %Conteo de vehículos
%% Calle 10
ROI_calle10 = impoly(gca,[5150,1240; 6580,1250; 6580,1340; 5150,1340]); %ROI calle 10
x = [5150 6580 6580 5150 5150];                                 %Coordenadas del punto el eje x
y = [1240 1250 1340 1340 1240];                                 %Coordenadas del punto el eje y
area_calle10 = area_calc_mex(x,y);                              %Cálculo de área calle 10
[total_carros10,propied10] = contabilizarGPU(ROI_calle10,Procesada);%Conteo de vehículos
clearvars zoom_semaforo4
hold off

%% Semáforo 5 %%
progreso_fig = sbprogress(progreso_fig,17,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','Sucre');                              %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');               %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on              %Presenta mapa
zoom_semaforo5 = [6726 9893 83 2466];                               %Región para acercamiento
axis(zoom_semaforo5);                                                   %Región acercada
text(6962,1279,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(8290,2345,'Antonio José de
Sucre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001);
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 5
ROI_calle5 = impoly(gca,[8247,1145; 8265,190; 8340,190; 8320,1154]);    %ROI calle 5
[total_carros5,propied5] = contabilizarGPU(ROI_calle5,Procesada);       %Conteo de vehículos
clearvars zoom_semaforo5
%% Calle 18
ROI_calle18 = impoly(gca,[8267,1385; 8247,2335; 8285,2335; 8311,1385]); %ROI calle 18
x = [8267 8247 8285 8311 8267];                                 %Coordenadas del punto el eje x
y = [1385 2335 2335 1385 1385];                                 %Coordenadas del punto el eje y
area_calle18 = area_calc_mex(x,y);                              %Cálculo de área calle 18
[total_carros18,propied18] = contabilizarGPU(ROI_calle18,Procesada);    %Conteo de vehículos
%% Calle 11
ROI_calle11 = impoly(gca,[6717,1240; 8210,1250; 8210,1330; 6717,1340]); %ROI calle 11
x = [6717 8210 8210 6717 6717];                                 %Coordenadas del punto el eje x
y = [1240 1250 1330 1340 1240];                                 %Coordenadas del punto el eje y
area_calle11 = area_calc_mex(x,y);                              %Cálculo de área calle 11
[total_carros11,propied11] = contabilizarGPU(ROI_calle11,Procesada);    %Conteo de vehículos
hold off

%% Semáforo 6 %%
progreso_fig = sbprogress(progreso_fig,20,'DisplayMode',2);             %Barra de progreso
```

```
set(handles.salida1,'String','18 de Noviembre');                        %Calle principal
set(handles.salida2,'String','Alonso de Mercadillo');                   %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'); hold on              %Presenta mapa
zoom_semaforo6 = [8341 11556 115 2498];                             %Región para acercamiento
axis(zoom_semaforo6);                                               %Región acercada
text(8575,1279,'Alonso de Mercadillo','color','y','FontSize',5,'FontWeight','bold')
text(9910,2200,'18 de Noviembre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 13
ROI_calle13 = impoly(gca,[10050,1240; 11450,1250; 11450,1340; 10050,1340]);%ROI calle 13
[total_carros13,propied13] = contabilizarGPU(ROI_calle13,Procesada);    %Conteo de vehículos
%% Calle 6
ROI_calle6 = impoly(gca,[9870,1130; 9890,185; 9965,185; 9940,1130]);    %ROI calle 6
x = [9870 9890 9965 9940 9870];                                 %Coordenadas del punto el eje x
y = [1130 185 185 1130 1130];                                   %Coordenadas del punto el eje y
area_calle6 = area_calc_mex(x,y);                               %Cálculo de área calle 6
[total_carros6,propied6] = contabilizarGPU(ROI_calle6,Procesada);       %Conteo de vehículos
%% Calle 12
ROI_calle12 = impoly(gca,[8375,1245; 9830,1250; 9830,1340; 8375,1340]); %ROI calle 12
x = [8375 9830 9830 8375 8375];                                 %Coordenadas del punto el eje x
y = [1245 1250 1340 1340 1245];                                 %Coordenadas del punto el eje y
area_calle12 = area_calc_mex(x,y);                              %Cálculo de área calle 12
[total_carros12,propied12] = contabilizarGPU(ROI_calle12,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo6
hold off

%% Semáforo 7 %%
progreso_fig = sbprogress(progreso_fig,23,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','Juan José Peña');                     %Calle principal
set(handles.salida2,'String','Azuay');                              %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'),hold on               %Presenta mapa
zoom_semaforo7 = [249 3416 1330 3649];                              %Región para acercamiento
axis(zoom_semaforo7);                                               %Región acercada
text(2275,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(1815,2141,'Juan José Peña','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');  %Mapa procesado
hold on
%% Calle 20
ROI_calle20 = impoly(gca,[265,2435; 1715,2455; 1715,2530; 265,2530]);   %ROI calle 20
[total_carros20,propied20] = contabilizarGPU(ROI_calle20,Procesada);    %Conteo de vehículos
%% Calle 21
ROI_calle21 = impoly(gca,[1885,2435; 3330,2455; 3330,2530; 1885,2530]); %ROI calle 21
x = [1885 3330 3330 1885 1885];                                 %Coordenadas del punto el eje x
y = [2435 2455 2530 2530 2435];                                 %Coordenadas del punto el eje y
area_calle21 = area_calc_mex(x,y);                              %Cálculo de área calle 21
[total_carros21,propied21] = contabilizarGPU(ROI_calle21,Procesada);    %Conteo de vehículos
%% Calle 27
ROI_calle27 = impoly(gca,[1785,2585; 1770,3535; 1840,3535; 1860,2585]); %ROI calle 27
x = [1785 1770 1840 1860 1785];                                 %Coordenadas del punto el eje x
y = [2585 3535 3535 2585 2585];                                 %Coordenadas del punto el eje y
area_calle27 = area_calc_mex(x,y);                              %Cálculo de área calle 27
[total_carros27,propied27] = contabilizarGPU(ROI_calle27,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo7
hold off

%% Semáforo 8 %%
progreso_fig = sbprogress(progreso_fig,26,'DisplayMode',2);  %Barra de progreso
set(handles.salida1,'String','José Joaquín de Olmedo');      %Calle principal
set(handles.salida2,'String','Azuay');                       %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on       %Presentar mapa
zoom_semaforo8 = [1858 5036 1341 3657];                      %Región para acercamiento
axis(zoom_semaforo8);                                        %Región acercada
text(3864,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(3425,3284,'J. J. de Olmedo','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
```

```
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 15
ROI_calle15 = impoly(gca,[3405,1385; 3387,2335; 3455,2335; 3480,1385]); %ROI calle 15
x = [3405 3387 3455 3480 3405];                              %Coordenadas del punto el eje x
y = [1385 2335 2335 1385 1385];                              %Coordenadas del punto el eje y
area_calle15 = area_calc_mex(x,y);                           %Cálculo de área calle 15
[total_carros15,propied15] = contabilizarGPU(ROI_calle15,Procesada);    %Conteo de vehículos
%% Calle 22
ROI_calle22 = impoly(gca,[3545,2455; 4955,2455; 4955,2530; 3545,2530]); %ROI calle 22
x = [3545 4955 4955 3545 3545];                              %Coordenadas del punto el eje x
y = [2455 2455 2530 2530 2455];                              %Coordenadas del punto el eje y
area_calle22 = area_calc_mex(x,y);                           %Cálculo de área calle 22
[total_carros22,propied22] = contabilizarGPU(ROI_calle22,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo8
hold off

%% Semáforo 9 %%
progreso_fig = sbprogress(progreso_fig,29,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Bernardo Valdivieso');               %Calle principal
set(handles.salida2,'String','Azuay');                             %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presenta mapa
zoom_semaforo9 = [3495 6678 1298 3681];                            %Región para acercamiento
axis(zoom_semaforo9);                                              %Región acercada
text(5488,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(5050,2305,'B. Valdivieso','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 29
ROI_calle29 = impoly(gca,[5030,2585; 5010,3530; 5080,3530; 5100,2585]);    %ROI calle 29
x = [5030 5010 5080 5100 5030];                              %Coordenadas del punto el eje x
y = [2585 3530 3530 2585 2585];                              %Coordenadas del punto el eje y
area_calle29 = area_calc_mex(x,y);                           %Cálculo de área calle 29
[total_carros29,propied29] = contabilizarGPU(ROI_calle29,Procesada);   %Conteo de vehículos
%% Calle 23
ROI_calle23 = impoly(gca,[5125,2450; 6560,2455; 6560,2530; 5125,2530]);    %ROI calle 23
x = [5125 6560 6560 5125 5125];                              %Coordenadas del punto el eje x
y = [2450 2455 2530 2530 2450];                              %Coordenadas del punto el eje y
area_calle23 = area_calc(x,y);                               %Cálculo de área calle 23
[total_carros23,propied23] = contabilizarGPU(ROI_calle23,Procesada);   %Conteo de vehículos
clearvars zoom_semaforo9
hold off

%% Semáforo 10 %%
progreso_fig = sbprogress(progreso_fig,32,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                     %Calle principal
set(handles.salida2,'String','Azuay');                             %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presentar mapa
zoom_semaforo10 = [5079 8262 1314 3665];                           %Región para acercamiento
axis(zoom_semaforo10);                                             %Región acercada
text(7427,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(6675,3145,'Simón Bolívar','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 17
ROI_calle17 = impoly(gca,[6625,2330; 6635,1385; 6720,1385; 6720,2330]); %ROI calle 17
x = [6625 6635 6720 6720 6625];                              %Coordenadas del punto el eje x
y = [2330 1385 1385 2330 2330];                              %Coordenadas del punto el eje y
area_calle17 = area_calc(x,y);                               %Cálculo de área calle 17
[total_carros17,propied17] = contabilizarGPU(ROI_calle17,Procesada);    %Conteo de vehículos
%% Calle 24
ROI_calle24 = impoly(gca,[6795,2445; 8185,2455; 8185,2530; 6795,2530]); %ROI calle 24
x = [6795 8185 8185 6795 6795];                              %Coordenadas del punto el eje x
y = [2445 2455 2530 2530 2445];                              %Coordenadas del punto el eje y
area_calle24 = area_calc_mex(x,y);                           %Cálculo de área calle 24
[total_carros24,propied24] = contabilizarGPU(ROI_calle24,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo10
hold off
```

```
%% Semáforo 11
progreso_fig = sbprogress(progreso_fig,35,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Sucre');                             %Calle principal
set(handles.salida2,'String','Azuay');                             %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presentar mapa
zoom_semaforo11 = [6742 9925 1330 3665];                           %Región para acercamiento
axis(zoom_semaforo11);                                             %Región acercada
text(8658,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(8285,2305,'Sucre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'); title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 31
ROI_calle31 = impoly(gca,[8270,2585; 8250,3530; 8320,3530; 8340,2585]); %ROI calle 31
x = [8270 8250 8320 8340 8270];                                    %Coordenadas del punto el eje x
y = [2585 3530 3530 2585 2585];                                    %Coordenadas del punto el eje y
area_calle31 = area_calc_mex(x,y);                                 %Cálculo de área calle 31
[total_carros31,propied31] = contabilizarGPU(ROI_calle31,Procesada);    %Conteo de vehículos
%% Calle 25
ROI_calle25 = impoly(gca,[8365,2445; 9800,2455; 9800,2530; 8365,2530]);    %ROI calle 25
x = [8365 9800 9800 8365 8365];                                    %Coordenadas del punto el eje x
y = [2445 2455 2530 2530 2445];                                    %Coordenadas del punto el eje y
area_calle25 = area_calc_mex(x,y);                                 %Cálculo de área calle 25
[total_carros25,propied25] = contabilizarGPU(ROI_calle25,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo11
hold off

%% Semáforo 12 %%%
progreso_fig = sbprogress(progreso_fig,38,'DisplayMode',2);            %Barra de progreso
set(handles.salida1,'String','18 de Noviembre');                       %Calle principal
set(handles.salida2,'String','Azuay');                                 %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit');
hold on
zoom_semaforo12 = [8341 11508 1346 3649];
axis(zoom_semaforo12);
text(10540,2482,'Azuay','color','y','FontSize',5,'FontWeight','bold')
text(9910,3145,'18 de Noviembre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');
hold on
%% Calle 19
ROI_calle19 = impoly(gca,[9870,2335; 9890,1385; 9960,1385; 9940,2335]); %ROI calle 19
x = [9870 9890 9960 9940 9870];                                    %Coordenadas del punto el eje x
y = [2335 1385 1385 2335 2335];                                    %Coordenadas del punto el eje y
area_calle19 = area_calc_mex(x,y);                                 %Cálculo de área calle 24
[total_carros19,propied19] = contabilizarGPU(ROI_calle19,Procesada);    %Conteo de vehículos
%% Calle 26
ROI_calle26 = impoly(gca,[9990,2445; 11430,2455; 11430,2530; 9990,2530]);  %ROI calle 26
x = [9990 11430 11430 9990 9990];                                  %Coordenadas del punto el eje x
y = [2445 2455 2530 2530 2445];                                    %Coordenadas del punto el eje y
area_calle26 = area_calc_mex(x,y);                                 %Cálculo de área calle 26
[total_carros26,propied26] = contabilizarGPU(ROI_calle26,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo12
hold off

%% Semáforo 13
progreso_fig = sbprogress(progreso_fig,42,'DisplayMode',2);  %Barra de progreso
set(handles.salida1,'String','Juan José Peña');              %Calle principal
set(handles.salida2,'String','Miguel Riofrío');              %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on       %Presenta mapa
zoom_semaforo13 = [233 3416 2498 4897];                      %Región para acercamiento
axis(zoom_semaforo13);                                       %Región acercada
text(2400,3684,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
text(1815,3427,'Juan José Peña','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 40
```

```
ROI_calle40 = impoly(gca,[1790,3785; 1765,4740; 1840,4740; 1860,3783]);    %ROI calle 40
x = [1790 1765 1840 1860 1790];                                  %Coordenadas del punto el eje x
y = [3785 4740 4740 3783 3785];                                  %Coordenadas del punto el eje y
area_calle40 = area_calc_mex(x,y);                               %Cálculo de área calle 40
[total_carros40,propied40] = contabilizarGPU(ROI_calle40,Procesada);    %Conteo de vehículos
%% Calle 33
ROI_calle33 = impoly(gca,[300,3660; 1700,3660; 1700,3745; 300,3745]);   %ROI calle 33
x = [300 1700 1700 300 300];                                     %Coordenadas del punto el eje x
y = [3660 3660 3745 3745 3660];                                  %Coordenadas del punto el eje y
area_calle33 = area_calc_mex(x,y);                               %Cálculo de área calle 33
[total_carros33,propied33] = contabilizarGPU(ROI_calle33,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo13
hold off

%% Semáforo 14 %%
progreso_fig = sbprogress(progreso_fig,45,'DisplayMode',2); %Barra de progreso
set(handles.salida1,'String','José Joaquín de Olmedo');     %Calle principal
set(handles.salida2,'String','Miguel Riofrío');             %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on      %Presenta mapa
zoom_semaforo14 = [1864 5031 2498 4865];                    %Región para acercamiento
axis(zoom_semaforo14);                                      %Región acercada
text(3971,3706,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
text(3423,3291,'J. J. de Olmedo','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 28
ROI_calle28 = impoly(gca,[3390,3540; 3410,2585; 3480,2585; 3460,3540]); %ROI calle 28
x = [3390 3410 3480 3460 3390];                                  %Coordenadas del punto el eje x
y = [3540 2585 2585 3540 3540];                                  %Coordenadas del punto el eje y
area_calle28 = area_calc_mex(x,y);                               %Cálculo de área calle 28
[total_carros28,propied28] = contabilizarGPU(ROI_calle28,Procesada);    %Conteo de vehículos
%% Calle 34
ROI_calle34 = impoly(gca,[1930,3660; 3325,3660; 3325,3740; 1930,3740]); %ROI calle 34
x = [1930 3325 3325 1930 1930];                                  %Coordenadas del punto el eje x
y = [3660 3660 3740 3740 3660];                                  %Coordenadas del punto el eje y
area_calle34 = area_calc_mex(x,y);                               %Cálculo de área calle 34
[total_carros34,propied34] = contabilizarGPU(ROI_calle34,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo14
hold off

%% Semáforo 15 %%
progreso_fig = sbprogress(progreso_fig,48,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Bernardo Valdivieso');               %Calle principal
set(handles.salida2,'String','Miguel Riofrío');                    %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presenta mapa
zoom_semaforo15 = [3448 6710 2466 4913];                           %Región para acercamiento
axis(zoom_semaforo15);                                             %Región acercada
text(5562,3703,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
text(5050,4654,'B. Valdivieso','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001);
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado'); %Mapa procesado
hold on
%% Calle 42
ROI_calle42 = impoly(gca,[5030,3790; 5010,4740; 5050,4740; 5060,3790]);    %ROI calle 42
x = [5030 5010 5050 5075 5030];                                  %Coordenadas del punto el eje x
y = [3790 4740 4740 3790 3790];                                  %Coordenadas del punto el eje y
area_calle42 = area_calc_mex(x,y);                               %Cálculo de área calle 42
[total_carros42,propied42] = contabilizarGPU(ROI_calle42,Procesada);    %Conteo de vehículos
%% Calle 35
ROI_calle35 = impoly(gca,[3565,3660; 4950,3660; 4950,3740; 3565,3740]);    %ROI calle 35
x = [3565 4950 4950 3565 3565];                                  %Coordenadas del punto el eje x
y = [3660 3660 3740 3740 3660];                                  %Coordenadas del punto el eje y
area_calle35 = area_calc_mex(x,y);                               %Cálculo de área calle 35
[total_carros35,propied35] = contabilizarGPU(ROI_calle35,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo15
hold off

%% Semáforo 16 %%
progreso_fig = sbprogress(progreso_fig,51,'DisplayMode',2);             %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                          %Calle principal
```

```
set(handles.salida2,'String','Miguel Riofrío');                          %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on              %Presenta mapa
zoom_semaforo16 = [5095 8277 2498 4881];                            %Región para acercamiento
axis(zoom_semaforo16);                                              %Región acercada
text(7216,3706,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
text(6663,3291,'Simón Bolívar','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 30
ROI_calle30 = impoly(gca,[6630,3540; 6650,2585; 6735,2585; 6700,3540]); %ROI calle 30
x = [6630 6650 6735 6700 6630];                                  %Coordenadas del punto el eje x
y = [3540 2585 2585 3540 3540];                                  %Coordenadas del punto el eje y
area_calle30 = area_calc_mex(x,y);                               %Cálculo de área calle 30
[total_carros30,propied30] = contabilizarGPU(ROI_calle30,Procesada);    %Conteo de vehículos
%% Calle 36
ROI_calle36 = impoly(gca,[5160,3660; 6570,3660; 6570,3740; 5160,3740]);    %ROI calle 36
x = [5160 6570 6570 5160 5160];                                  %Coordenadas del punto el eje x
y = [3660 3660 3740 3740 3660];                                  %Coordenadas del punto el eje y
area_calle36 = area_calc_mex(x,y);                               %Cálculo de área calle 36
[total_carros36,propied36] = contabilizarGPU(ROI_calle36,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo16
hold off

%% Semáforo 17 %%
progreso_fig = sbprogress(progreso_fig,54,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','Sucre');                              %Calle principal
set(handles.salida2,'String','Miguel Riofrío');                     %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on              %Presenta mapa
zoom_semaforo17 = [6678 9893 2498 4897];                            %Región para acercamiento
axis(zoom_semaforo17);                                              %Región acercada
text(8290,3380,'Sucre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(7127,3695,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 44
ROI_calle44 = impoly(gca,[8270,3785; 8250,4750; 8290,4750; 8310,3785]);    %ROI calle 44
x = [8270 8250 8290 8310 8270];                                  %Coordenadas del punto el eje x
y = [3785 4750 4750 3785 3785];                                  %Coordenadas del punto el eje y
area_calle44 = area_calc_mex(x,y);                               %Cálculo de área calle 36
[total_carros44,propied44] = contabilizarGPU(ROI_calle44,Procesada);   %Conteo de vehículos
clearvars zoom_semaforo17
%% Calle 37
ROI_calle37 = impoly(gca,[6780,3660; 8200,3660; 8200,3740; 6780,3740]);    %ROI calle 37
x = [6780 8200 8200 6780 6780];                                  %Coordenadas del punto el eje x
y = [3660 3660 3740 3740 3660];                                  %Coordenadas del punto el eje y
area_calle37 = area_calc_mex(x,y);                               %Cálculo de área calle 36
[total_carros37,propied37] = contabilizarGPU(ROI_calle37,Procesada);    %Conteo de vehículos
hold off

%% Semáforo 18 %%
progreso_fig = sbprogress(progreso_fig,57,'DisplayMode',2);         %Barra de progreso
set(handles.salida1,'String','18 de Noviembre');                    %Calle principal
set(handles.salida2,'String','Miguel Riofrío');                     %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on              %Presenta mapa
zoom_semaforo18 = [8325 11540 2498 4880];                           %Región para acercamiento
axis(zoom_semaforo18);                                              %Región acercada
text(9912,3285,'18 de Noviembre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(10290,3695,'Miguel Riofrío','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 39
ROI_calle39 = impoly(gca,[10000,3660; 11450,3660; 11450,3740; 10000,3740]);%ROI calle 39
[total_carros39,propied39] = contabilizarGPU(ROI_calle39,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo18
%% Calle 32
```

```
ROI_calle32 = impoly(gca,[9870,3540; 9890,2585; 9960,2585; 9940,3540]); %ROI calle 32
x = [9870 9890 9960 9940 9870];                                  %Coordenadas del punto el eje x
y = [3540 2585 2585 3540 3540];                                  %Coordenadas del punto el eje y
area_calle32 = area_calc_mex(x,y);                               %Cálculo de área calle 32
[total_carros32,propied32] = contabilizarGPU(ROI_calle32,Procesada);    %Conteo de vehículos
%% Calle 38
ROI_calle38 = impoly(gca,[8410,3660; 9815,3660; 9815,3740; 8410,3740]); %ROI calle 39
x = [8410 9815 9815 8410 8410];                                  %Coordenadas del punto el eje x
y = [3660 3660 3740 3740 3660];                                  %Coordenadas del punto el eje y
area_calle38 = area_calc_mex(x,y);                               %Cálculo de área calle 36
[total_carros38,propied38] = contabilizarGPU(ROI_calle38,Procesada);    %Conteo de vehículos
hold off

%% Semáforo 19 %%
progreso_fig = sbprogress(progreso_fig,60,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Juan José Peña');                    %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');                %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presentar mapa
zoom_semaforo19 = [256 3407 3738 6054];                            %Región para acercamiento
axis(zoom_semaforo19);                                             %Región acercada
text(1815,4700,'Juan José Peña','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(2370,4885,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 46
ROI_calle46 = impoly(gca,[265,4860; 1700,4860; 1700,4930; 265,4930]);  %ROI calle 46
[total_carros46,propied46] = contabilizarGPU(ROI_calle46,Procesada);   %Conteo de vehículos
%% Calle 53
ROI_calle53 = impoly(gca,[1790,4987; 1770,5940; 1840,5940; 1860,4987]);%ROI calle 53
x = [1790 1700 1840 1860 1790];                                  %Coordenadas del punto el eje x
y = [4987 5940 5940 4987 4987];                                  %Coordenadas del punto el eje y
area_calle53 = area_calc_mex(x,y);                               %Cálculo de área calle 53
[total_carros53,propied53] = contabilizarGPU(ROI_calle53,Procesada);   %Conteo de vehículos
%% Calle 47
ROI_calle47 = impoly(gca,[1890,4860; 3330,4860; 3330,4930; 1890,4930]);%ROI calle 47
x = [1890 3330 3330 1890 1890];                                  %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                  %Coordenadas del punto el eje y
area_calle47 = area_calc_mex(x,y);                               %Cálculo de área calle 47
[total_carros47,propied47] = contabilizarGPU(ROI_calle47,Procesada);   %Conteo de vehículos
clearvars zoom_semaforo19
hold off

%% Semáforo 20 %%
progreso_fig = sbprogress(progreso_fig,63,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','José Joaquín de Olmedo');            %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');                %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presentar mapa
zoom_semaforo20 = [1864 5015 3729 6048];                           %Región para acercamiento
axis(zoom_semaforo20);                                             %Región acercada
text(3425,4650,'José Joaquín de
Olmedo','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(3920,4880,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 41
ROI_calle41 = impoly(gca,[3390,4750; 3410,3788; 3480,3785; 3460,4750]);    %ROI calle 41
x = [3390 3410 3480 3460 3390];                                  %Coordenadas del punto el eje x
y = [4750 3788 3785 4750 4750];                                  %Coordenadas del punto el eje y
area_calle41 = area_calc_mex(x,y);                               %Cálculo de área calle 41
[total_carros41,propied41] = contabilizarGPU(ROI_calle41,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo20
%% Calle 48
ROI_calle48 = impoly(gca,[3505,4860; 4935,4860; 4935,4930; 3468,4930]); %ROI calle 48
x = [3505 4935 4935 3468 3505];                                  %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                  %Coordenadas del punto el eje y
area_calle48 = area_calc_mex(x,y);                               %Cálculo de área calle 48
[total_carros48,propied48] = contabilizarGPU(ROI_calle48,Procesada);    %Conteo de vehículos
hold off
```

```
%% Semáforo 21 %%
progreso_fig = sbprogress(progreso_fig,66,'DisplayMode',2);       %Barra de progreso
set(handles.salida1,'String','Bernardo Valdivieso');              %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');               %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on            %Presenta mapa
zoom_semaforo21 = [3495 6662 3761 6064];                          %Región para acercamiento
axis(zoom_semaforo21);                                            %Región acercada
text(5050,5680,'Bernardo
Valdivieso','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(5655,4890,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'),title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 55
ROI_calle55 = impoly(gca,[5030,4990; 5010,5945; 5070,5945; 5100,4990]);   %ROI calle 55
x = [5030 5010 5080 5100 5030];                                   %Coordenadas del punto el eje x
y = [4990 5945 5945 4990 4990];                                   %Coordenadas del punto el eje y
area_calle55 = area_calc_mex(x,y);                                %Cálculo de área calle 55
[total_carros55,propied55] = contabilizarGPU(ROI_calle55,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo21
%% Calle 49
ROI_calle49 = impoly(gca,[5125,4860; 6565,4860; 6565,4930; 5125,4930]); %ROI calle 49
x = [5125 6565 6565 5125 5125];                                   %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                   %Coordenadas del punto el eje y
area_calle49 = area_calc_mex(x,y);                                %Cálculo de área calle 49
[total_carros49,propied49] = contabilizarGPU(ROI_calle49,Procesada);    %Conteo de vehículos
hold off

%% Semáforo 22 %%
progreso_fig = sbprogress(progreso_fig,69,'DisplayMode',2);       %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                    %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');               %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on            %Presenta mapa
zoom_semaforo22 = [5079 8277 3729 6080];                          %Región para acercamiento
axis(zoom_semaforo22);                                            %Región acercada
text(6670,4680,'Simón Bolívar','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(7205,4890,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'),title('Mapa procesado'); %Mapa procesado
hold on
%% Calle 43
ROI_calle43 = impoly(gca,[6630,4750; 6650,3788; 6720,3788; 6700,4750]);   %ROI calle 43
x = [6630 6650 6720 6700 6630];                                   %Coordenadas del punto el eje x
y = [4750 3788 3788 4750 4750];                                   %Coordenadas del punto el eje y
area_calle43 = area_calc_mex(x,y);                                %Cálculo de área calle 43
[total_carros43,propied43] = contabilizarGPU(ROI_calle43,Procesada);    %Conteo de vehículos
%% Calle 50
ROI_calle50 = impoly(gca,[6745,4860; 8180,4860; 8180,4930; 6745,4930]); %ROI calle 50
x = [6745 8180 8180 6745 6745];                                   %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                   %Coordenadas del punto el eje y
area_calle50 = area_calc_mex(x,y);                                %Cálculo de área calle 50
[total_carros50,propied50] = contabilizarGPU(ROI_calle50,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo22
hold off

%% Semáforo 23 %%
progreso_fig = sbprogress(progreso_fig,72,'DisplayMode',2);       %Barra de progreso
set(handles.salida1,'String','Sucre');                            %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');               %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on            %Presenta mapa
zoom_semaforo23 = [6747 9913 3751 6088];                          %Región para acercamiento
axis(zoom_semaforo23);                                            %Región acercada
text(8285,5744,'Sucre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(8794,4890,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 57
```

```
ROI_calle57 = impoly(gca,[8270,4990; 8250,5945; 8320,5945; 8340,4990]); %ROI calle 57
x = [8270 8250 8320 8340 8270];                                  %Coordenadas del punto el eje x
y = [4990 5945 5945 4990 4990];                                  %Coordenadas del punto el eje y
area_calle57 = area_calc_mex(x,y);                               %Cálculo de área calle 57
[total_carros57,propied57] = contabilizarGPU(ROI_calle57,Procesada);    %Conteo de vehículos
%% Calle 51
ROI_calle51 = impoly(gca,[8365,4860; 9815,4860; 9815,4930; 8365,4930]); %ROI calle 51
x = [8365 9815 9815 8365 8365];                                  %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                  %Coordenadas del punto el eje y
area_calle51 = area_calc_mex(x,y);                               %Cálculo de área calle 51
[total_carros51,propied51] = contabilizarGPU(ROI_calle51,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo23
hold off

%% Semáforo 24 %%
progreso_fig = sbprogress(progreso_fig,75,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','18 de Noviembre');                   %Calle principal
set(handles.salida2,'String','Vicente Rocafuerte');                %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presenta mapa
zoom_semaforo24 = [8341 11524 3729 6032];                          %Región para acercamiento
axis(zoom_semaforo24);                                             %Región acercada
text(9910,4465,'18 de Noviembre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(10540,4890,'Vicente Rocafuerte','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Imagen procesada
hold on
%% Calle 45
ROI_calle45 = impoly(gca,[9870,4735; 9890,3790; 9960,3790; 9940,4735]); %ROI calle 45
x = [9870 9890 9960 9940 9870];                                  %Coordenadas del punto el eje x
y = [4735 3790 3790 4735 4735];                                  %Coordenadas del punto el eje y
area_calle45 = area_calc_mex(x,y);                               %Cálculo de área calle 45
[total_carros45,propied45] = contabilizarGPU(ROI_calle45,Procesada);    %Conteo de vehículos
%% Calle 52
ROI_calle52 = impoly(gca,[9990,4860; 11430,4860; 11430,4930; 9990,4930]);  %ROI calle 52
x = [9990 11430 11430 9990 9990];                                %Coordenadas del punto el eje x
y = [4860 4860 4930 4930 4860];                                  %Coordenadas del punto el eje y
area_calle52 = area_calc_mex(x,y);                               %Cálculo de área calle 52
[total_carros52,propied52] = contabilizarGPU(ROI_calle52,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo24
hold off

%% Semáforo 25 %%
progreso_fig = sbprogress(progreso_fig,78,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Juan José Peña');                    %Calle principal
set(handles.salida2,'String','10 de Agosto');                      %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presenta mapa
zoom_semaforo25 = [233 3432 4881 7296];                            %Región para acercamiento
axis(zoom_semaforo25);                                             %Región acercada
text(1815,5800,'Juan José Peña','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(2340,6100,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');%Mapa procesado
hold on
%% Calle 66
ROI_calle66 = impoly(gca,[1790,6190; 1770,7150; 1840,7150; 1860,6190]);    %ROI calle 66
x = [1790 1770 1840 1860 1790];                                  %Coordenadas del punto el eje x
y = [6190 7150 7150 6190 6190];                                  %Coordenadas del punto el eje y
area_calle66 = area_calc_mex(x,y);                               %Cálculo de área calle 66
[total_carros66,propied66] = contabilizarGPU(ROI_calle66,Procesada);    %Conteo de vehículos
%% Calle 59
ROI_calle59 = impoly(gca,[280,6060; 1725,6060; 1725,6140; 280,6140]);   %ROI calle 59
x = [280 1725 1725 280 280];                                     %Coordenadas del punto el eje x
y = [6060 6060 6140 6140 6060];                                  %Coordenadas del punto el eje Y
area_calle59 = area_calc_mex(x,y);                               %Cálculo de área calle 59
[total_carros59,propied59] = contabilizarGPU(ROI_calle59,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo25
hold off

%% Semáforo 26 %%
progreso_fig = sbprogress(progreso_fig,81,'DisplayMode',2);        %Barra de proceso
set(handles.salida1,'String','José Joaquín de Olmedo');            %Calle principal
```

```
set(handles.salida2,'String','10 de Agosto');                        %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on               %Presentar mapa
zoom_semaforo26 = [1848 5031 4913 7296];                             %Región para acercamiento
axis(zoom_semaforo26);                                               %Región acercada
pause(0.001)
text(3425,7030,'José Joaquín de
Olmedo','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(2400,6100,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesada
hold on
%% Calle 67
ROI_calle67 = impoly(gca,[3410,6190; 3390,7145; 3460,7145; 3480,6190]); %ROI calle 67
[total_carros67,propied67] = contabilizarGPU(ROI_calle67,Procesada);    %Conteo de vehículos
%% Calle 54
ROI_calle54 = impoly(gca,[3390,5940; 3410,4990; 3480,4990; 3460,5940]); %ROI calle 54
x = [3390 3410 3480 3460 3390];                                  %Coordenadas del punto el eje x
y = [5940 4990 4990 5940 5940];                                  %Coordenadas del punto el eje y
area_calle54 = area_calc_mex(x,y);                               %Cálculo de área calle 54
[total_carros54,propied54] = contabilizarGPU(ROI_calle54,Procesada);    %Conteo de vehículos
%% Calle 60
ROI_calle60 = impoly(gca,[1900,6060; 3345,6060; 3345,6140; 1900,6140]); %ROI calle 60
x = [1900 3345 3345 1900 1900];                                  %Coordenadas del punto el eje x
y = [6060 6060 6140 6140 6060];                                  %Coordenadas del punto el eje y
area_calle60 = area_calc_mex(x,y);                               %Cálculo de área calle 60
[total_carros60,propied60] = contabilizarGPU(ROI_calle60,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo26
hold off

%% Semáforo 27 %%
progreso_fig = sbprogress(progreso_fig,84,'DisplayMode',2);              %Barra de progreso
set(handles.salida1,'String','Bernardo Valdivieso');                     %Calle principal
set(handles.salida2,'String','10 de Agosto');                            %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on               %Presentar mapa
zoom_semaforo27 = [3480 6678 4913 7280];                             %Región para acercamiento
axis(zoom_semaforo27);                                               %Región acercada
text(5066,5780,'Bernardo
Valdivieso','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(5492,6100,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 68
ROI_calle68 = impoly(gca,[5030,6190; 5010,7150; 5080,7150; 5105,6190]);    %ROI calle 68
x = [5030 5010 5080 5105 5030];                                  %Coordenadas del punto el eje x
y = [6190 7150 7150 7150 6190];                                  %Coordenadas del punto el eje y
area_calle68 = area_calc_mex(x,y);                               %Cálculo de área calle 68
[total_carros68,propied68] = contabilizarGPU(ROI_calle68,Procesada);    %Conteo de vehículos
%% Calle 61
ROI_calle61 = impoly(gca,[3555,6060; 4955,6060; 4955,6140; 3555,6140]); %ROI calle 61
x = [3555 4955 4955 3555 3555];                                  %Coordenadas del punto el eje x
y = [6060 6060 6140 6140 6060];                                  %Coordenadas del punto el eje y
area_calle61 = area_calc_mex(x,y);                               %Cálculo de área calle 61
[total_carros61,propied61] = contabilizarGPU(ROI_calle61,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo27
hold off

%% Semáforo 28 %%
progreso_fig = sbprogress(progreso_fig,87,'DisplayMode',2);          %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                       %Calle principal
set(handles.salida2,'String','10 de Agosto');                        %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on               %Presentar mapa
zoom_semaforo28 = [5047 8309 4913 7280];                             %Región para acercamiento
axis(zoom_semaforo28);                                               %Acercamiento al mapa
text(6670,5495,'Simón Bolívar','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(7115,6100,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
```

```
%% Calle 69
ROI_calle69 = impoly(gca,[6650,6190; 6630,7150; 6700,7150; 6720,6190]); %ROI calle 69
[total_carros69,propied69] = contabilizarGPU(ROI_calle69,Procesada);    %Conteo de vehículos
%% Calle 56
ROI_calle56 = impoly(gca,[6630,5950; 6650,5005; 6690,5005; 6670,5950]);  %ROI calle 56
x = [6630 6650 6690 6670 6630];                                %Coordenadas del punto el eje x
y = [5950 5005 5005 5950 5950];                                %Coordenadas del punto el eje y
area_calle56 = area_calc_mex(x,y);                             %Cálculo de área calle 56
[total_carros56,propied56] = contabilizarGPU(ROI_calle56,Procesada);    %Conteo de vehículos
%% Calle 62
ROI_calle62 = impoly(gca,[5170,6060; 6580,6060; 6580,6140; 5170,6150]);    %ROI calle 62
x = [5170 6580 6580 5170 5170];                                %Coordenadas del punto el eje x
y = [6060 6060 6140 6150 6060];                                %Coordenadas del punto el eje y
area_calle62 = area_calc_mex(x,y);                             %Cálculo de área calle 62
[total_carros62,propied62] = contabilizarGPU(ROI_calle62,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo28
hold off

%% Semáforo 29
progreso_fig = sbprogress(progreso_fig,90,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','Simón Bolívar');                     %Calle principal
set(handles.salida2,'String','10 de Agosto');                      %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presentar mapa
zoom_semaforo29 = [6726 9893 4897 7296];                           %Región para acercamiento
axis(zoom_semaforo29);                                             %Presenta acercamiento
text(8290,6862,'Sucre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(6962,6076,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 70
ROI_calle70 = impoly(gca,[8270,6190; 8250,7140; 8330,7140; 8320,6190]); %ROI calle 70
x = [8270 8250 8330 8320 8270];                                 %Coordenadas del punto el eje x
y = [6190 7140 7140 6190 6190];                                 %Coordenadas del punto el eje y
area_calle70 = area_calc_mex(x,y);                              %Cálculo de área calle 70
[total_carros70,propied70] = contabilizarGPU(ROI_calle70,Procesada);    %Conteo de vehículos
%% Calle 63
ROI_calle63 = impoly(gca,[6780,6060; 8195,6060; 8195,6131; 6780,6135]); %ROI calle 63
x = [6780 8195 8195 6780 6780];                                 %Coordenadas del punto el eje x
y = [6060 6060 6131 6135 6060];                                 %Coordenadas del punto el eje y
area_calle63 = area_calc_mex(x,y);                              %Cálculo de área calle 63
[total_carros63,propied63] = contabilizarGPU(ROI_calle63,Procesada);    %Conteo de vehículos
clearvars zoom_semaforo29
hold off

%% Semáforo 30 %%
progreso_fig = sbprogress(progreso_fig,93,'DisplayMode',2);        %Barra de progreso
set(handles.salida1,'String','18 de Noviembre');                   %Calle principal
set(handles.salida2,'String','10 de Agosto');                      %Calle secundaria
%% Zoom calles
imshow(MapaLoja,'initialmagnification','fit'), hold on             %Presenta mapa
zoom_semaforo30 = [8357 11540 4928 7279];                          %Región para acercamiento
axis(zoom_semaforo30);                                             %Presenta acercamiento
text(9910,6825,'18 de Noviembre','color','y','FontSize',5,'FontWeight','bold','Rotation',90)
text(10520,6096,'10 de Agosto','color','y','FontSize',5,'FontWeight','bold')
pause(0.001)
hold off
%% Presentar imagen procesada para realizar el conteo
imshow(Procesada,'initialmagnification','fit'), title('Mapa procesado');   %Mapa procesado
hold on
%% Calle 71
ROI_calle71 = impoly(gca,[9890,6190; 9870,7150; 9940,7150; 9960,6190]); %ROI calle 71
[total_carros71,propied71] = contabilizarGPU(ROI_calle71,Procesada);    %Conteo de vehículos
%% Calle 65
ROI_calle65 = impoly(gca,[10030,6060; 11450,6060; 11450,6140; 10030,6140]);%ROI calle 65
[total_carros65,propied65] = contabilizarGPU(ROI_calle65,Procesada);    %Conteo de vehículos
%% Calle 58
ROI_calle58 = impoly(gca,[9870,5945; 9890,4990; 9960,4990; 9940,5945]); %ROI calle 58
x = [9870 9890 9960 9940 9870];                                 %Coordenadas del punto el eje x
y = [5945 4990 4990 5945 5945];                                 %Coordenadas del punto el eje y
area_calle58 = area_calc_mex(x,y);                              %Cálculo de área calle 58
[total_carros58,propied58] = contabilizarGPU(ROI_calle58,Procesada);    %Conteo de vehículos
%% Calle 64
ROI_calle64 = impoly(gca,[8370,6060; 9820,6060; 9820,6140; 8370,6140]); %ROI calle 64
```

```
x = [8370 9820 9820 8370 8370];                              %Coordenadas del punto el eje x
y = [6060 6060 6140 6140 6060];                              %Coordenadas del punto el eje y
area_calle64 = area_calc_mex(x,y);                           %Cálculo de área calle 64
[total_carros64,propied64] = contabilizarGPU(ROI_calle64,Procesada);    %Conteo de vehículos
clearvars Procesada semaforoimagen zoom_semaforo30

%% PROBABILIDAD OCUPACIÓN Y DISPONIBILIDAD SEMÁFOROS %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Presentar imagen de mapa
imshow(MapaLoja); title('Porcentaje de disponibilidad en cada calle');
hold off
%% Barra de progreso
progreso_fig = sbprogress(progreso_fig,96,'DisplayMode',2);

%% Semáforo 1
[ocup_S1_Princ,disp_S1_Princ] = porcentaje_mex(total_carros14, area_calle14);
[ocup_S1_Sec,disp_S1_Sec] = porcentaje_mex(total_carros7, area_calle7);
%% Semáforo 2
[ocup_S2_Princ,disp_S2_Princ] = porcentaje_mex(total_carros2, area_calle2);
[ocup_S2_Sec,disp_S2_Sec] = porcentaje_mex(total_carros8, area_calle8);
%% Semáforo 3
[ocup_S3_Princ,disp_S3_Princ] = porcentaje_mex(total_carros16, area_calle16);
[ocup_S3_Sec,disp_S3_Sec] = porcentaje_mex(total_carros9, area_calle9);
%% Semáforo 4
[ocup_S4_Princ,disp_S4_Princ] = porcentaje_mex(total_carros4, area_calle4);
[ocup_S4_Sec,disp_S4_Sec] = porcentaje_mex(total_carros10, area_calle10);
%% Semáforo 5
[ocup_S5_Princ,disp_S5_Princ] = porcentaje_mex(total_carros18, area_calle18);
[ocup_S5_Sec,disp_S5_Sec] = porcentaje_mex(total_carros11, area_calle11);
%% Semáforo 6
[ocup_S6_Princ,disp_S6_Princ] = porcentaje_mex(total_carros6, area_calle6);
[ocup_S6_Sec,disp_S6_Sec] = porcentaje_mex(total_carros12, area_calle12);
%% Semáforo 7
[ocup_S7_Princ,disp_S7_Princ] = porcentaje_mex(total_carros27, area_calle27);
[ocup_S7_Sec,disp_S7_Sec] = porcentaje_mex(total_carros21, area_calle21);
%% Semáforo 8
[ocup_S8_Princ,disp_S8_Princ] = porcentaje_mex(total_carros15, area_calle15);
[ocup_S8_Sec,disp_S8_Sec] = porcentaje_mex(total_carros22, area_calle22);
%% Semáforo 9
[ocup_S9_Princ,disp_S9_Princ] = porcentaje_mex(total_carros29, area_calle29);
[ocup_S9_Sec,disp_S9_Sec] = porcentaje_mex(total_carros23, area_calle23);
%% Semáforo 10
[ocup_S10_Princ,disp_S10_Princ] = porcentaje_mex(total_carros17, area_calle17);
[ocup_S10_Sec,disp_S10_Sec] = porcentaje_mex(total_carros24, area_calle24);
%% Semáforo 11
[ocup_S11_Princ,disp_S11_Princ] = porcentaje_mex(total_carros31, area_calle31);
[ocup_S11_Sec,disp_S11_Sec] = porcentaje_mex(total_carros25, area_calle25);
%% Semáforo 12
[ocup_S12_Princ,disp_S12_Princ] = porcentaje_mex(total_carros19, area_calle19);
[ocup_S12_Sec,disp_S12_Sec] = porcentaje_mex(total_carros26, area_calle26);
%% Semáforo 13
[ocup_S13_Princ,disp_S13_Princ] = porcentaje_mex(total_carros40, area_calle40);
[ocup_S13_Sec,disp_S13_Sec] = porcentaje_mex(total_carros33, area_calle33);
%% Semáforo 14
[ocup_S14_Princ,disp_S14_Princ] = porcentaje_mex(total_carros28, area_calle28);
[ocup_S14_Sec,disp_S14_Sec] = porcentaje_mex(total_carros34, area_calle34);
%% Semáforo 15
[ocup_S15_Princ,disp_S15_Princ] = porcentaje_mex(total_carros42, area_calle42);
[ocup_S15_Sec,disp_S15_Sec] = porcentaje_mex(total_carros35, area_calle35);
%% Semáforo 16
[ocup_S16_Princ,disp_S16_Princ] = porcentaje_mex(total_carros30, area_calle30);
[ocup_S16_Sec,disp_S16_Sec] = porcentaje_mex(total_carros36, area_calle36);
%% Semáforo 17
[ocup_S17_Princ,disp_S17_Princ] = porcentaje_mex(total_carros44, area_calle44);
[ocup_S17_Sec,disp_S17_Sec] = porcentaje_mex(total_carros37, area_calle37);
%% Semáforo 18
[ocup_S18_Princ,disp_S18_Princ] = porcentaje_mex(total_carros32, area_calle32);
[ocup_S18_Sec,disp_S18_Sec] = porcentaje_mex(total_carros38, area_calle38);
%% Semáforo 19
[ocup_S19_Princ,disp_S19_Princ] = porcentaje_mex(total_carros53, area_calle53);
[ocup_S19_Sec,disp_S19_Sec] = porcentaje_mex(total_carros47, area_calle47);
%% Semáforo 20
[ocup_S20_Princ,disp_S20_Princ] = porcentaje_mex(total_carros41, area_calle41);
[ocup_S20_Sec,disp_S20_Sec] = porcentaje_mex(total_carros48, area_calle48);
%% Semáforo 21
[ocup_S21_Princ,disp_S21_Princ] = porcentaje_mex(total_carros55, area_calle55);
```

```matlab
[ocup_S21_Sec,disp_S21_Sec] = porcentaje_mex(total_carros49, area_calle49);
%% Semáforo 22
[ocup_S22_Princ,disp_S22_Princ] = porcentaje_mex(total_carros43, area_calle43);
[ocup_S22_Sec,disp_S22_Sec] = porcentaje_mex(total_carros50, area_calle50);
%% Semáforo 23
[ocup_S23_Princ,disp_S23_Princ] = porcentaje_mex(total_carros57, area_calle57);
[ocup_S23_Sec,disp_S23_Sec] = porcentaje_mex(total_carros51, area_calle51);
%% Semáforo 24
[ocup_S24_Princ,disp_S24_Princ] = porcentaje_mex(total_carros45, area_calle45);
[ocup_S24_Sec,disp_S24_Sec] = porcentaje_mex(total_carros52, area_calle52);
%% Semáforo 25
[ocup_S25_Princ,disp_S25_Princ] = porcentaje_mex(total_carros66, area_calle66);
[ocup_S25_Sec,disp_S25_Sec] = porcentaje_mex(total_carros59, area_calle59);
%% Semáforo 26
[ocup_S26_Princ,disp_S26_Princ] = porcentaje_mex(total_carros54, area_calle54);
[ocup_S26_Sec,disp_S26_Sec] = porcentaje_mex(total_carros60, area_calle60);
%% Semáforo 27
[ocup_S27_Princ,disp_S27_Princ] = porcentaje_mex(total_carros68, area_calle68);
[ocup_S27_Sec,disp_S27_Sec] = porcentaje_mex(total_carros61, area_calle61);
%% Semáforo 28
[ocup_S28_Princ,disp_S28_Princ] = porcentaje_mex(total_carros56, area_calle56);
[ocup_S28_Sec,disp_S28_Sec] = porcentaje_mex(total_carros62, area_calle62);
%% Semáforo 29
[ocup_S29_Princ,disp_S29_Princ] = porcentaje_mex(total_carros70, area_calle70);
[ocup_S29_Sec,disp_S29_Sec] = porcentaje_mex(total_carros63, area_calle63);
%% Semáforo 30
[ocup_S30_Princ,disp_S30_Princ] = porcentaje_mex(total_carros58, area_calle58);
[ocup_S30_Sec,disp_S30_Sec] = porcentaje_mex(total_carros64, area_calle64);
%% Guardo variables
save VehiculosContabilizados.mat;

%% Guardo variables de porcentajes tanto de ocupación como disponibilidad
save PorcentajeOcupacion.mat ocup_S1_Princ disp_S1_Princ ocup_S1_Sec disp_S1_Sec ocup_S2_Princ
disp_S2_Princ ocup_S2_Sec disp_S2_Sec ocup_S3_Princ disp_S3_Princ ocup_S3_Sec disp_S3_Sec
ocup_S4_Princ disp_S4_Princ ocup_S4_Sec disp_S4_Sec ocup_S5_Princ disp_S5_Princ ocup_S5_Sec
disp_S5_Sec ocup_S6_Princ disp_S6_Princ ocup_S6_Sec disp_S6_Sec ocup_S7_Princ disp_S7_Princ
ocup_S7_Sec disp_S7_Sec ocup_S8_Princ disp_S8_Princ ocup_S8_Sec disp_S8_Sec ocup_S9_Princ
disp_S9_Princ ocup_S9_Sec disp_S9_Sec ocup_S10_Princ disp_S10_Princ ocup_S10_Sec disp_S10_Sec
ocup_S11_Princ disp_S11_Princ ocup_S11_Sec disp_S11_Sec ocup_S12_Princ disp_S12_Princ
ocup_S12_Sec disp_S12_Sec ocup_S13_Princ disp_S13_Princ ocup_S13_Sec disp_S13_Sec
ocup_S14_Princ disp_S14_Princ ocup_S14_Sec disp_S14_Sec ocup_S15_Princ disp_S15_Princ
ocup_S15_Sec disp_S15_Sec ocup_S16_Princ disp_S16_Princ ocup_S16_Sec disp_S16_Sec
ocup_S17_Princ disp_S17_Princ ocup_S17_Sec disp_S17_Sec ocup_S18_Princ disp_S18_Princ
ocup_S18_Sec disp_S18_Sec ocup_S19_Princ disp_S19_Princ ocup_S19_Sec disp_S19_Sec
ocup_S20_Princ disp_S20_Princ ocup_S20_Sec disp_S20_Sec ocup_S21_Princ disp_S21_Princ
ocup_S21_Sec disp_S21_Sec ocup_S22_Princ disp_S22_Princ ocup_S22_Sec disp_S22_Sec
ocup_S23_Princ disp_S23_Princ ocup_S23_Sec disp_S23_Sec ocup_S24_Princ disp_S24_Princ
ocup_S24_Sec disp_S24_Sec ocup_S25_Princ disp_S25_Princ ocup_S25_Sec disp_S25_Sec
ocup_S26_Princ disp_S26_Princ ocup_S26_Sec disp_S26_Sec ocup_S27_Princ disp_S27_Princ
ocup_S27_Sec disp_S27_Sec ocup_S28_Princ disp_S28_Princ ocup_S28_Sec disp_S28_Sec
ocup_S29_Princ disp_S29_Princ ocup_S29_Sec disp_S29_Sec ocup_S30_Princ disp_S30_Princ
ocup_S30_Sec disp_S30_Sec

%% Presento mapa de Loja
imshow(MapaLoja); title('Porcentaje de disponibilidad en cada calle');
hold on

%% PRESENTO FLECHAS DE DISPONIBILIDAD EN EL MAPA %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
calles_mapa(MapaLoja);

%% Barra de progreso
progreso_fig = sbprogress(progreso_fig,100,'Msg','Conteo completo','DisplayMode',2);
sbprogress(progreso_fig,'Close')
clearvars ans progreso_fig
%% Cargo imagen de semáforo
semaforoimagen = handles.img_semaforo;
for i = 1:30
    axis_semaforo = sprintf('semaforo%d',i);
    axes(handles.(axis_semaforo)),imshow(semaforoimagen);
end
axis off
hold off
%% Tiempo de procesamiento
tiempo = toc;
set(handles.salida_proc,'String',sprintf('%d segundos',round(tiempo)));
```

```
% --- Executes on selection change in popupmenu1.
function popupmenu1_Callback(hObject, eventdata, handles)

%% REALIZO UN ZOOM SOBRE CADA INTERSECCIÓN QUE CUBRE EL SEMÁFORO %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Resetear text_field y popupmenú
set([handles.text_inicio handles.text_fin],'String','','BackgroundColor',[0.94 0.94 0.94]);
set(handles.salida_proc,'String','','BackgroundColor',[0.94 0.94 0.94]);
set(handles.popupmenu2, 'Value', 1);
set(handles.popupmenu3, 'Value', 1);
%% Cargar imagen semáforo
semaforoimagen = imread('semaforo.jpg');
%% Selección de opciones del pop up menú
seleccion = get(handles.popupmenu1,'Value');
%% Barra de proceso
progreso_fig=sbprogress([],0,'Msg','Procesando...','Parent',CONTEOVEHICULAR_V2,'DisplayMode',2
);
progreso_fig = sbprogress(progreso_fig,50,'DisplayMode',2);
%% Cargar imagen mapa de Loja
MapaLoja = handles.mapa;
axes(handles.mosaico);

tic

switch seleccion

  case 1

    uwait(msgbox('Seleccione semáforo'));

  case 2

    %% Cargo variables
    load VehiculosContabilizados.mat total_carros1 total_carros8 propied1...
    propied8 disp_S1_Princ disp_S1_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Juan José Peña');
    set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
    set(handles.salida3,'String',total_carros1);
    set(handles.salida4,'String',total_carros8);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S1_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S1_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo1 = [233 3400 99 2546];
    axis(zoom_semaforo1);
    hold on
    %% Habilito conexión puerto serial
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied1,propied8,total_carros1,total_carros8,handles);
    %% Coordenadas flechas
    x_pare1 = 1810;    y_pare1 = 1870;                                    %Pare Principal
    x_pare2 = 950;     y_pare2 = 1300;                                    %Pare Secundaria
    x_flecha_prin = [1810 1520];     y_flecha_prin = [1810 2220];         %Flecha Secundaria
    x_flecha_sec = [1300 1300];     y_flecha_sec = [600 1300];            %Flecha Principal
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S1_Princ,disp_S1_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flecha_
    prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    hold on
    %% Enciendo LEDS
    %encenderLEDS(total_carros1,total_carros8,a)
    %% Presento en el text field calle recomendada
    if disp_S1_Princ >= disp_S1_Sec
       set(handles.salida9,'String','Juan José Peña');
    else
       set(handles.salida9,'String','Alonso de Mercadillo');
    end

  case 3

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros15 total_carros9 propied15 propied9...
```

```
  disp_S2_Princ disp_S2_Sec
  cla;
  %% Presento el número de vehículos y porcentaje en el text field
  set([handles.salida1 handles.salida5],'String','Jose Joaquín de Olmedo');
  set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
  set(handles.salida3,'String',total_carros15);
  set(handles.salida4,'String',total_carros9);
  set(handles.salida7,'String',sprintf('%d %c',round(disp_S2_Princ),'%'));
  set(handles.salida8,'String',sprintf('%d %c',round(disp_S2_Sec),'%'));
  %% Cargo mapa guardado en el handles
  imshow(MapaLoja);
  hold off
  %% Realizo zoom al mapa
  zoom_semaforo2 = [1864 5015 99 2466];
  axis(zoom_semaforo2);
  hold on
  %% Habilitar puerto serial
  %a = arduino('COM4');
  %% Grafico las cajas de frontera de los vehículos
  cajasfrontera(propied15,propied9,total_carros15,total_carros9,handles);
  %% Coordenadas flechas
  x_pare1 = 3430;   y_pare1 = 680;                                    %Pare Principal
  x_pare2 = 2570;   y_pare2 = 1300;                                   %Pare Secundaria
  x_flecha_prin = [3430 1030];   y_flecha_prin = [3430 330];          %Flecha Principal
  x_flecha_sec = [2920 1300];    y_flecha_sec = [2220 1300];          %Flecha Secundaria
  %% Presentar flechas
   flechas_disponibilidad(disp_S2_Princ,disp_S2_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flecha_
   prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
  %% Enciendo LEDS
  %encenderLEDS(total_carros15,total_carros9,a)
  %% Presento en el text field calle recomendada
  if disp_S2_Princ >= disp_S2_Sec
     set(handles.salida9,'String','José Joaquín de Olmedo');
  else
     set(handles.salida9,'String','Alonso de Mercadillo');
  end

case 4

  %% Cargo variables del archivo VehiculosContabilizados.mat
  load VehiculosContabilizados.mat total_carros3 total_carros10 propied3 propied10...
  disp_S3_Princ disp_S3_Sec
  cla;
  %% Presento el número de vehículos y porcentaje en el text field
  set([handles.salida1 handles.salida5],'String','Bernardo Valdivieso');
  set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
  set(handles.salida3,'String',total_carros3);
  set(handles.salida4,'String',total_carros10);
  set(handles.salida7,'String',sprintf('%d %c',round(disp_S3_Princ),'%'));
  set(handles.salida8,'String',sprintf('%d %c',round(disp_S3_Sec),'%'));
  %% Cargo mapa guardado en el handles
  imshow(MapaLoja);
  hold off
  %% Realizo zoom a las calles
  zoom_semaforo3 = [3463 6646 83 2466];
  axis(zoom_semaforo3);
  hold on
  %% Habilito conexión puerto serial
  %a = arduino('COM4');
  %% Grafico las cajas de frontera de los vehículos
  cajasfrontera(propied3,propied10,total_carros3,total_carros10,handles);
  %% Coordenadas flechas
  x_pare1 = 5050;   y_pare1 = 1870;                                   %Pare Principal
  x_pare2 = 4190;   y_pare2 = 1300;                                   %Pare Secundaria
  x_flecha_prin = [5050 1520];    y_flecha_prin = [5050 2220];        %Flecha Principal
  x_flecha_sec = [4540 1300];    y_flecha_sec = [3840 1300];          %Flecha Secundaria
  %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S3_Princ,disp_S3_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flecha_
   prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
  %% Enciendo LEDS
  %encenderLEDS(total_carros3,total_carros10,a)
  %% Presento en el text field calle recomendada
  if disp_S3_Princ >= disp_S3_Sec
     set(handles.salida9,'String','Bernardo Valdivieso');
  else
     set(handles.salida9,'String','Alonso de Mercadillo');
  end
```

```
case 5

  %% Cargo variables del archivo VehiculosContabilizados.mat
  load VehiculosContabilizados.mat total_carros17 total_carros11 propied17 propied11...
      disp_S4_Princ disp_S4_Sec
      cla;
      %% Presento el número de vehículos y porcentaje en el text field
      set([handles.salida1 handles.salida5],'String','Simón Bolívar');
      set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
      set(handles.salida3,'String',total_carros17);
      set(handles.salida4,'String',total_carros11);
      set(handles.salida7,'String',sprintf('%d %c',round(disp_S4_Princ),'%'));
      set(handles.salida8,'String',sprintf('%d %c',round(disp_S4_Sec),'%'));
      %% Cargo mapa guardado en el handles
      imshow(MapaLoja);
      hold off
      %% Realizo zoom a las calles
      zoom_semaforo4 = [5129 8254 107 2482];
      axis(zoom_semaforo4);
      hold on
      %% Habilitar conexión puerto serial
      %a = arduino('COM4');
      %% Grafico las cajas de frontera de los vehículos
      cajasfrontera(propied17,propied11,total_carros17,total_carros11,handles);
      %% Coordenadas flechas
      x_pare1 = 6670;   y_pare1 = 680;                                    %Pare Principal
      x_pare2 = 5800;   y_pare2 = 1300;                                   %Pare Secundaria
      x_flecha_prin = [6670 1030];    y_flecha_prin = [6670 330];         %Flecha Principal
      x_flecha_sec = [6150 1300];     y_flecha_sec = [5450 1300];         %Flecha Secundaria
      %% Presentar flechas de disponibilidad
     flechas_disponibilidad(disp_S4_Princ,disp_S4_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flec
     ha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
      %% Enciendo LEDS
      %encenderLEDS(total_carros17,total_carros11,a)
      %% Presento en el text field calle recomendada
      if disp_S4_Princ >= disp_S4_Sec
          set(handles.salida9,'String','Simón Bolívar');
      else
          set(handles.salida9,'String','Alonso de Mercadillo');
      end

  case 6

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros5 total_carros12 propied5 propied12...
    disp_S5_Princ disp_S5_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
    set(handles.salida3,'String',total_carros5);
    set(handles.salida4,'String',total_carros12);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S5_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S5_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo5 = [6726 9893 83 2466];
    axis(zoom_semaforo5);
    hold on
    %% Habilito comunicación puerto serial
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied5,propied12,total_carros5,total_carros12,handles);
    %% Coordenadas de las flechas
    x_pare1 = 8290;   y_pare1 = 1870;                                   %Pare Principal
    x_pare2 = 7430;    y_pare2 = 1300;                                  %Pare Secundaria
    x_flecha_prin = [8290 1520];   y_flecha_prin = [8290 2220];         %Flecha Principal
    x_flecha_sec = [7780 1300];    y_flecha_sec = [7080 1300];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S5_Princ,disp_S5_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flech
    a_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros5,total_carros12,a)
    %% Presento en el text field calle recomendada
```

```
  if disp_S5_Princ >= disp_S5_Sec
       set(handles.salida9,'String','Antonio José de Sucre');
  else
       set(handles.salida9,'String','Alonso de Mercadillo');
  end

case 7

  %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros13 total_carros19 propied13 propied19...
    disp_S6_Princ disp_S6_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','18 de Noviembre');
    set([handles.salida2 handles.salida6],'String','Alonso de Mercadillo');
    set(handles.salida3,'String',total_carros13);
    set(handles.salida4,'String',total_carros19);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S6_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S6_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo6 = [8341 11556 115 2498];
    axis(zoom_semaforo6);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied13,propied19,total_carros13,total_carros19,handles);
    %% Coordenadas de las flechas
    x_pare1 = 9920;   y_pare1 = 680;                                    %Pare Principal
    x_pare2 = 9060;   y_pare2 = 1300;                                   %Pare Secundaria
    x_flecha_prin = [9920 1030];    y_flecha_prin = [9920 330];         %Flecha Principal
    x_flecha_sec = [9410 1300];    y_flecha_sec = [8710 1300];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S6_Princ,disp_S6_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flec
   ha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros3,total_carros19,a)
    %% Presento en el text field calle recomendada
    if disp_S6_Princ >= disp_S6_Sec
        set(handles.salida9,'String','18 de Noviembre');
    else
        set(handles.salida9,'String','Alonso de Mercadillo');
    end

case 8

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros14 total_carros20 propied14...
    propied20 disp_S7_Princ disp_S7_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Juan José Peña');
    set([handles.salida2 handles.salida6],'String','Azuay');
    set(handles.salida3,'String',total_carros14);
    set(handles.salida4,'String',total_carros20);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S7_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S7_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo7 = [249 3416 1330 3649];
    axis(zoom_semaforo7);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied14,propied20,total_carros14,total_carros20,handles);
    %% Coordenadas de las flechas
    x_pare1 = 1810;   y_pare1 = 3050;                                   %Pare Principal
    x_pare2 = 2580;   y_pare2 = 2480;                                   %Pare Secundaria
    x_flecha_prin = [1810 2700];   y_flecha_prin = [1810 3400];         %Flecha Principal
    x_flecha_sec = [2230 2480];    y_flecha_sec = [2930 2480];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
```

```
    flechas_disponibilidad(disp_S7_Princ,disp_S7_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flec
    ha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
     %% Enciendo LEDS
     %encenderLEDS(total_carros14,total_carros20,a)
     %% Presento en el text field calle recomendada
     if disp_S7_Princ >= disp_S7_Sec
         set(handles.salida9,'String','Juan José Peña');
     else
         set(handles.salida9,'String','Azuay');
     end

case 9

     %% Cargo variables del archivo VehiculosContabilizados.mat
     load VehiculosContabilizados.mat total_carros28 total_carros21 propied28 propied21...
     disp_S8_Princ disp_S8_Sec
     cla;
     %% Presento el número de vehículos y porcentaje en el text field
     set([handles.salida1 handles.salida5],'String','José Joaquín de Olmedo');
     set([handles.salida2 handles.salida6],'String','Azuay');
     set(handles.salida3,'String',total_carros28);
     set(handles.salida4,'String',total_carros21);
     set(handles.salida7,'String',sprintf('%d %c',round(disp_S8_Princ),'%'));
     set(handles.salida8,'String',sprintf('%d %c',round(disp_S8_Sec),'%'));
     %% Cargo mapa guardado en el handles
     imshow(MapaLoja);
     hold off
     %% Realizo zoom a las calles
     zoom_semaforo8 = [1858 5036 1341 3657];
     axis(zoom_semaforo8);
     hold on
     %% Habilito puerto COM y pines de salida
     %a = arduino('COM4');
     %% Grafico las cajas de frontera de los vehículos
     cajasfrontera(propied28,propied21,total_carros28,total_carros21,handles);
     %% Coordenadas flechas
     x_pare1 = 3430;   y_pare1 = 1900;                                  %Pare Principal
     x_pare2 = 4200;   y_pare2 = 2480;                                  %Pare Secundaria
     x_flecha_prin = [3430 2250];    y_flecha_prin = [3430 1550];       %Flecha Principal
     x_flecha_sec = [3850 2480];    y_flecha_sec = [4550 2480];         %Flecha Secundaria
      %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S8_Princ,disp_S8_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flec
    ha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
     %% Enciendo LEDS
     %encenderLEDS(total_carros28,total_carros21,a)
     %% Presento en el text field calle recomendada
     if disp_S8_Princ >= disp_S8_Sec
         set(handles.salida9,'String','José Joaquín de Olmedo');
     else
         set(handles.salida9,'String','Azuay');
     end

case 10

     %% Cargo variables del archivo VehiculosContabilizados.mat
     load VehiculosContabilizados.mat total_carros16 total_carros22 propied16...
     propied22 disp_S9_Princ disp_S9_Sec
     cla;
     %% Presento el número de vehículos y porcentaje en el text field
     set([handles.salida1 handles.salida5],'String','Bernardo Valdivieso');
     set([handles.salida2 handles.salida6],'String','Azuay');
     set(handles.salida3,'String',total_carros16);
     set(handles.salida4,'String',total_carros22);
     set(handles.salida7,'String',sprintf('%d %c',round(disp_S9_Princ),'%'));
     set(handles.salida8,'String',sprintf('%d %c',round(disp_S9_Sec),'%'));
     %% Cargo mapa guardado en el handles
     imshow(MapaLoja);
     hold off
     %% Realizo zoom a las calles
     zoom_semaforo9 = [3495 6678 1298 3681];
     axis(zoom_semaforo9);
     hold on
     %% Habilito puerto COM y pines de salida
     %a = arduino('COM4');
     %% Grafico las cajas de frontera de los vehículos
     cajasfrontera(propied16,propied22,total_carros16,total_carros22,handles);
     %% Coordenadas de flechas
```

```
    x_pare1 = 5060;   y_pare1 = 3050;                                %Pare Principal
    x_pare2 = 5820;   y_pare2 = 2480;                                %Pare Secundaria
    x_flecha_prin = [5060 2700];    y_flecha_prin = [5060 3400];     %Flecha Principal
    x_flecha_sec = [5470 2480];    y_flecha_sec = [6170 2480];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S9_Princ,disp_S9_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_flec
    ha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros16,total_carros22,a)
    %% Presento en el text field calle recomendada
    if disp_S9_Princ >= disp_S9_Sec
        set(handles.salida9,'String','Bernardo Valdivieso');
    else
        set(handles.salida9,'String','Azuay');
    end

case 11

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros30 total_carros23 propied30...
    propied23 disp_S10_Princ disp_S10_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Simón Bolívar');
    set([handles.salida2 handles.salida6],'String','Azuay');
    set(handles.salida3,'String',total_carros30);
    set(handles.salida4,'String',total_carros23);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S10_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S10_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo10 = [5079 8262 1314 3665];
    axis(zoom_semaforo10);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied30,propied23,total_carros30,total_carros23,handles);
    %% Coordenadas flechas
    x_pare1 = 6670;   y_pare1 = 1900;                                %Pare Principal
    x_pare2 = 7430;   y_pare2 = 2480;                                %Pare Secundaria
    x_flecha_prin = [6670 2250];   y_flecha_prin = [6670 1550];      %Flecha Principal
    x_flecha_sec = [7080 2480];    y_flecha_sec = [7780 2480];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S10_Princ,disp_S10_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros30,total_carros23,a)
    %% Presento en el text field calle recomendada
    if disp_S10_Princ >= disp_S10_Sec
        set(handles.salida9,'String','Simón Bolívar');
    else
        set(handles.salida9,'String','Azuay');
    end

case 12

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros18 total_carros24 propied18...
    propied24 disp_S11_Princ disp_S11_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','Azuay');
    set(handles.salida3,'String',total_carros18);
    set(handles.salida4,'String',total_carros24);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S11_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S11_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo11 = [6742 9925 1330 3665];
    axis(zoom_semaforo11);
    hold on
```

```
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied18,propied24,total_carros18,total_carros24,handles);
    %% Coordenadas flechas
    x_pare1 = 8290;   y_pare1 = 3050;                              %Pare Principal
    x_pare2 = 9060;   y_pare2 = 2480;                              %Pare Secundaria
    x_flecha_prin = [8290 2700];    y_flecha_prin = [8290 3400];   %Flecha Principal
    x_flecha_sec = [8710 2480];    y_flecha_sec = [9410 2480];     %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S11_Princ,disp_S11_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros18,total_carros24,a)
    %% Presento en el text field calle recomendada
    if disp_S11_Princ >= disp_S11_Sec
        set(handles.salida9,'String','Antonio José de Sucre');
    else
        set(handles.salida9,'String','Azuay');
    end

case 13

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros32 total_carros25 propied32...
    propied25 disp_S12_Princ disp_S12_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','Azuay');
    set(handles.salida3,'String',total_carros32);
    set(handles.salida4,'String',total_carros25);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S12_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S12_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo12 = [8341 11508 1346 3649];
    axis(zoom_semaforo12);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied32,propied25,total_carros32,total_carros25,handles);
    %% Coordenadas flechas
    x_pare1 = 9920;   y_pare1 = 1900;                              %Pare Principal
    x_pare2 = 10660;   y_pare2 = 2480;                             %Pare Secundaria
    x_flecha_prin = [9920 2250];    y_flecha_prin = [9920 1550];   %Flecha Principal
    x_flecha_sec = [10310 2480];    y_flecha_sec = [11010 2480];   %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S12_Princ,disp_S12_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros32,total_carros25,a)
    %% Presento en el text field calle recomendada
    if disp_S12_Princ >= disp_S12_Sec
        set(handles.salida9,'String','18 de Noviembre');
    else
        set(handles.salida9,'String','Azuay');
    end

case 14

    %% Cargo variables del archivo VehiculosContabilizados.m
    load VehiculosContabilizados.mat total_carros27 total_carros34 propied27 propied34...
    disp_S13_Princ disp_S13_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Juan José Peña');
    set([handles.salida2 handles.salida6],'String','Miguel Riofrío');
    set(handles.salida3,'String',total_carros27);
    set(handles.salida4,'String',total_carros34);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S13_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S13_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
```

```
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo13 = [233 3416 2498 4897];
    axis(zoom_semaforo13);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied27,propied34,total_carros27,total_carros34,handles);
    %% Coordenadas flechas
    x_pare1 = 1810;   y_pare1 = 4290;                                   %Pare Principal
    x_pare2 = 950;   y_pare2 = 3700;                                    %Pare Secundaria
    x_flecha_prin = [1810 3940];    y_flecha_prin = [1810 4640];        %Flecha Principal
    x_flecha_sec = [1300 3700];    y_flecha_sec = [600 3700];
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S13_Princ,disp_S13_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros27,total_carros34,a)
    %% Presento en el text field calle recomendada
    if disp_S13_Princ >= disp_S13_Sec
        set(handles.salida9,'String','Juan José Peña');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 15

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros41 total_carros35 propied41 propied35...
    disp_S14_Princ disp_S14_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','José Joaquín de Olmedo');
    set([handles.salida2 handles.salida6],'String','Miguel Riofrío');
    set(handles.salida3,'String',total_carros41);
    set(handles.salida4,'String',total_carros35);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S14_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S14_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo14 = [1864 5031 2498 4865];
    axis(zoom_semaforo14);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied41,propied35,total_carros41,total_carros35,handles);
    %% Coordenadas flechas
    x_pare1 = 3430;   y_pare1 = 3050;                                   %Pare Principal
    x_pare2 = 2580;   y_pare2 = 3700;                                   %Pare Secundaria
    x_flecha_prin = [3430 3400];   y_flecha_prin = [3430 2700];         %Flecha Principal
    x_flecha_sec = [2930 3700];    y_flecha_sec = [2230 3700];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S14_Princ,disp_S14_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros41,total_carros35,a)
    %% Presento en el text field calle recomendada
    if disp_S14_Princ >= disp_S14_Sec
        set(handles.salida9,'String','José Joaquín de Olmedo');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 16

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros29 total_carros36 propied29 propied36...
    disp_S15_Princ disp_S15_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Bernardo Valdivieso');
    set([handles.salida2 handles.salida6],'String','Miguel Rifrío');
    set(handles.salida3,'String',total_carros29);
```

```
    set(handles.salida4,'String',total_carros36);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S15_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S15_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo15 = [3463 6678 2482 4881];
    axis(zoom_semaforo15);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied29,propied36,total_carros29,total_carros36,handles);
    %% Coordenadas flechas
    x_pare1 = 5050;   y_pare1 = 4290;                                   %Pare Principal
    x_pare2 = 4190;   y_pare2 = 3700;                                   %Pare Secundaria
    x_flecha_prin = [5050 3940];    y_flecha_prin = [5050 4640];        %Flecha Principal
    x_flecha_sec = [4540 3700];    y_flecha_sec = [3840 3700];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S15_Princ,disp_S15_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros29,total_carros36,a)
    %% Presento en el text field calle recomendada
    if disp_S15_Princ >= disp_S15_Sec
        set(handles.salida9,'String','Bernardo Valdivieso');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 17

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros43 total_carros37 propied43 propied37...
    disp_S16_Princ disp_S16_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Simón Bolívar');
    set([handles.salida2 handles.salida6],'String','Miguel Rifrío');
    set(handles.salida3,'String',total_carros43);
    set(handles.salida4,'String',total_carros37);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S16_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S16_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo16 = [5095 8277 2498 4881];
    axis(zoom_semaforo16);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied43,propied37,total_carros43,total_carros37,handles);
    %% Coordenadas flechas
    x_pare1 = 6670;   y_pare1 = 3050;                                   %Pare Principal
    x_pare2 = 5810;   y_pare2 = 3700;                                   %Pare Secundaria
    x_flecha_prin = [6670 3400];   y_flecha_prin = [6670 2700];         %Flecha Principal
    x_flecha_sec = [6160 3700];    y_flecha_sec = [5460 3700];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S16_Princ,disp_S16_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros43,total_carros37,a)
    %% Presento en el text field calle recomendada
    if disp_S16_Princ >= disp_S16_Sec
        set(handles.salida9,'String','Simón Bolívar');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 18
    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros31 total_carros38 propied31...
    propied38 disp_S17_Princ disp_S17_Sec
    cla;
```

```
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','Miguel Rifrío');
    set(handles.salida3,'String',total_carros31);
    set(handles.salida4,'String',total_carros38);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S17_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S17_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo17 = [6678 9893 2498 4897];
    axis(zoom_semaforo17);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied31,propied38,total_carros31,total_carros38,handles);
    %% Coordenadas flechas
    x_pare1 = 8290;   y_pare1 = 4290;                                   %Pare Principal
    x_pare2 = 7430;   y_pare2 = 3700;                                   %Pare Secundaria
    x_flecha_prin = [8290 3940];   y_flecha_prin = [8290 4640];         %Flecha Principal
    x_flecha_sec = [7780 3700];    y_flecha_sec = [7080 3700];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S17_Princ,disp_S17_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros31,total_carros38,a)
    %% Presento en el text field calle recomendada
    if disp_S17_Princ >= disp_S17_Sec
        set(handles.salida9,'String','Antonio José de Sucre');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 19

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros45 total_carros39 propied45...
    propied39 disp_S18_Princ disp_S18_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','18 de Noviembre');
    set([handles.salida2 handles.salida6],'String','Miguel Rifrío');
    set(handles.salida3,'String',total_carros45);
    set(handles.salida4,'String',total_carros39);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S18_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S18_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo18 = [8325 11540 2498 4880];
    axis(zoom_semaforo18);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied45,propied39,total_carros45,total_carros39,handles);
    %% Coordenadas flechas
    x_pare1 = 9910;   y_pare1 = 3050;                                 %Pare Principal
    x_pare2 = 9050;   y_pare2 = 3700;                                 %Pare Secundaria
    x_flecha_prin = [9910 3400];   y_flecha_prin = [9910 2700];       %Flecha Principal
    x_flecha_sec = [9400 3700];    y_flecha_sec = [8700 3700];        %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S18_Princ,disp_S18_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros45,total_carros39,a)
    %% Presento en el text field calle recomendada
    if disp_S18_Princ >= disp_S18_Sec
        set(handles.salida9,'String','18 de Noviembre');
    else
        set(handles.salida9,'String','Miguel Riofrío');
    end

case 20
```

```
    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros40 total_carros46 propied40...
    propied46 disp_S19_Princ disp_S19_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Juan José Peña');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros40);
    set(handles.salida4,'String',total_carros46);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S19_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S19_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo19 = [256 3407 3738 6054];
    axis(zoom_semaforo19);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied40,propied46,total_carros40,total_carros46,handles);
    %% Coordenadas flechas
    x_pare1 = 1810;   y_pare1 = 5450;                              %Pare Principal
    x_pare2 = 2570;   y_pare2 = 4890;                              %Pare Secundaria
    x_flecha_prin = [1810 5100];   y_flecha_prin = [1810 5800];    %Flecha Principal
    x_flecha_sec = [2220 4890];    y_flecha_sec = [2920 4890];     %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S19_Princ,disp_S19_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros40,total_carros46,a)
    %% Presento en el text field calle recomendada
    if disp_S19_Princ >= disp_S19_Sec
        set(handles.salida9,'String','Juan José Peña');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 21

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros54 total_carros47 propied54...
    propied47 disp_S20_Princ disp_S20_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','José Joaquín de Olmedo');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros54);
    set(handles.salida4,'String',total_carros47);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S20_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S20_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo20 = [1864 5015 3729 6048];
    axis(zoom_semaforo20);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied54,propied47,total_carros54,total_carros47,handles);
    %% Coordenadas flechas
    x_pare1 = 3430;   y_pare1 = 4260;                              %Pare Principal
    x_pare2 = 4190;   y_pare2 = 4890;                              %Pare Secundaria
    x_flecha_prin = [3430 4610];   y_flecha_prin = [3430 3910];    %Flecha Principal
    x_flecha_sec = [3840 4890];    y_flecha_sec = [4540 4890];     %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S20_Princ,disp_S20_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros54,total_carros47,a);
    %% Disponibilidad
    if disp_S20_Princ >= disp_S20_Sec
        set(handles.salida9,'String','José Joaquín de Olmedo');
```

```
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 22

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros42 total_carros48 propied42...
    propied48 disp_S21_Princ disp_S21_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Bernardo Valdivieso');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros42);
    set(handles.salida4,'String',total_carros48);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S21_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S21_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo21 = [3495 6662 3761 6064];
    axis(zoom_semaforo21);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied42,propied48,total_carros42,total_carros48,handles);
    %% Coordenadas de flechas
    x_pare1 = 5050;   y_pare1 = 5450;                                %Pare Principal
    x_pare2 = 5810;   y_pare2 = 4890;                                %Pare Secundaria
    x_flecha_prin = [5050 5100];    y_flecha_prin = [5050 5800];     %Flecha Principal
    x_flecha_sec = [5460 4890];    y_flecha_sec = [6160 4890];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S21_Princ,disp_S21_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros42,total_carros48,a)
    %% Presento disponibilidad
    if disp_S21_Princ >= disp_S21_Sec
        set(handles.salida9,'String','Bernardo Valdivieso');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 23

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros56 total_carros49 propied56...
    propied49 disp_S22_Princ disp_S22_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Simón Bolívar');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros56);
    set(handles.salida4,'String',total_carros49);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S22_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S22_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo22 = [5079 8277 3729 6080];
    axis(zoom_semaforo22);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied56,propied49,total_carros56,total_carros49,handles);
    %% Coordenadas flechas
    x_pare1 = 6670;   y_pare1 = 4260;                                %Pare Principal
    x_pare2 = 7430;   y_pare2 = 4890;                                %Pare Secundaria
    x_flecha_prin = [6670 4610];   y_flecha_prin = [6670 3910];      %Flecha Principal
    x_flecha_sec = [7080 4890];    y_flecha_sec = [7780 4890];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S22_Princ,disp_S22_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
```

```
    %% Enciendo LEDS
    %encenderLEDS(total_carros56,total_carros49,a)
    %% Presento disponibilidad
    if disp_S22_Princ >= disp_S22_Sec
        set(handles.salida9,'String','Simón Bolívar');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 24

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros44 total_carros50 propied44...
    propied50 disp_S23_Princ disp_S23_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros44);
    set(handles.salida4,'String',total_carros50);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S23_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S23_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo23 = [6747 9913 3751 6088];
    axis(zoom_semaforo23);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied44,propied50,total_carros44,total_carros50,handles);
    %% Coordenadas flechas
    x_pare1 = 8290;   y_pare1 = 5450;                                   %Pare Principal
    x_pare2 = 9050;   y_pare2 = 4890;                                   %Pare Secundaria
    x_flecha_prin = [8290 5100];   y_flecha_prin = [8290 5800];         %Flecha Principal
    x_flecha_sec = [8700 4890];    y_flecha_sec = [9400 4890];          %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S23_Princ,disp_S23_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros44,total_carros50,a)
    %% Presento disponibilidad
    if disp_S23_Princ >= disp_S23_Sec
        set(handles.salida9,'String','Antonio José de Sucre');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 25

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros58 total_carros51 propied58...
    propied51 disp_S24_Princ disp_S24_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','18 de Noviembre');
    set([handles.salida2 handles.salida6],'String','Vicente Rocafuerte');
    set(handles.salida3,'String',total_carros58);
    set(handles.salida4,'String',total_carros51);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S24_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S24_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo24 = [8341 11524 3729 6032];
    axis(zoom_semaforo24);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied58,propied51,total_carros58,total_carros51,handles);
    %% Coordenadas flechas
    x_pare1 = 9910;   y_pare1 = 4260;                                   %Pare Principal
    x_pare2 = 10680;   y_pare2 = 4890;                                  %Pare Secundaria
```

```
    x_flecha_prin = [9910 4610];   y_flecha_prin = [9910 3910];        %Flecha Principal
    x_flecha_sec = [10330 4890];    y_flecha_sec = [11030 4890];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S24_Princ,disp_S24_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros58,total_carros51,a)
    %% Presento disponibilidad
    if disp_S24_Princ >= disp_S24_Sec
        set(handles.salida9,'String','18 de Noviembre');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 26

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros53 total_carros60 propied53 propied60...
    disp_S25_Princ disp_S25_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Juan José Peña');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros53);
    set(handles.salida4,'String',total_carros60);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S25_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S25_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo25 = [233 3432 4881 7296];
    axis(zoom_semaforo25);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied53,propied60,total_carros53,total_carros60,handles);
    %% Coordenadas de flechas
    x_pare1 = 1810;   y_pare1 = 6650;                                %Pare Principal
    x_pare2 = 950;    y_pare2 = 6100;                                %Pare Secundaria
    x_flecha_prin = [1810 6300];    y_flecha_prin = [1810 7000];     %Flecha Principal
    x_flecha_sec = [1300 6100];    y_flecha_sec = [600 6100];        %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S25_Princ,disp_S25_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros53,total_carros60,a)
    %% Presento disponibilidad
    if disp_S25_Princ >= disp_S25_Sec
        set(handles.salida9,'String','Juan José Peña');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 27
    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros67 total_carros61 propied67...
    propied61 disp_S26_Princ disp_S26_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','José Joaquín de Olmedo');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros67);
    set(handles.salida4,'String',total_carros61);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S26_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S26_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo26 = [1848 5031 4913 7296];
    axis(zoom_semaforo26);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
```

```
    cajasfrontera(propied67,propied61,total_carros67,total_carros61,handles);
    %% Coordenadas flechas
    x_pare1 = 3430;   y_pare1 = 5450;                              %Pare Principal
    x_pare2 = 2580;   y_pare2 = 6100;                              %Pare Secundaria
    x_flecha_prin = [3430 5800];   y_flecha_prin = [3430 5100];    %Flecha Principal
    x_flecha_sec = [2930 6100];    y_flecha_sec = [2230 6100];     %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S26_Princ,disp_S26_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros67,total_carros61,a);
    %% Presento disponibilidad
    if disp_S26_Princ >= disp_S26_Sec
        set(handles.salida9,'String','José Joaquín de Olmedo');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 28

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros55 total_carros62 propied55...
    propied62 disp_S27_Princ disp_S27_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Bernardo Valdivieso');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros55);
    set(handles.salida4,'String',total_carros62);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S27_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S27_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo27 = [3480 6678 4913 7280];
    axis(zoom_semaforo27);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied55,propied62,total_carros55,total_carros62,handles);
    %% Coordenadas de flechas
    x_pare1 = 5050;   y_pare1 = 6650;                              %Pare Principal
    x_pare2 = 4220;   y_pare2 = 6100;                              %Pare Secundaria
    x_flecha_prin = [5050 6300];    y_flecha_prin = [5050 7000];   %Flecha Principal
    x_flecha_sec = [4570 6100];    y_flecha_sec = [3870 6100];     %Flecha Secundaria
    %% Presentar flechas de disponibilidad
    flechas_disponibilidad(disp_S27_Princ,disp_S27_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
    echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros55,total_carros62,a);
    %% Presento disponibilidad
    if disp_S27_Princ >= disp_S27_Sec
        set(handles.salida9,'String','José Joaquín de Olmedo');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 29
    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros69 total_carros63 propied69...
    propied63 disp_S28_Princ disp_S28_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Simón Bolívar');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros69);
    set(handles.salida4,'String',total_carros63);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S28_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S28_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo28 = [5095 8277 4913 7312];
    axis(zoom_semaforo28);
```

```
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied69,propied63,total_carros69,total_carros63,handles);
    %% Coordenadas flechas
    x_pare1 = 6670;   y_pare1 = 5450;                                %Pare Principal
    x_pare2 = 5810;   y_pare2 = 6100;                                %Pare Secundaria
    x_flecha_prin = [6670 5800];   y_flecha_prin = [6670 5100];      %Flecha Principal
    x_flecha_sec = [6160 6100];    y_flecha_sec = [5460 6100];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
   flechas_disponibilidad(disp_S28_Princ,disp_S28_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
   echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros69,total_carros63,a);
    %% Presento disponibilidad
    if disp_S28_Princ >= disp_S28_Sec
        set(handles.salida9,'String','Simón Bolívar');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 30

    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros57 total_carros64 propied57...
    propied64 disp_S29_Princ disp_S29_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','Antonio José de Sucre');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros57);
    set(handles.salida4,'String',total_carros64);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S29_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S29_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
    hold off
    %% Realizo zoom a las calles
    zoom_semaforo29 = [6726 9893 4897 7296];
    axis(zoom_semaforo29);
    hold on
    %% Habilito puerto COM y pines de salida
    %a = arduino('COM4');
    %% Grafico las cajas de frontera de los vehículos
    cajasfrontera(propied57,propied64,total_carros57,total_carros64,handles);
    %% Coordenadas de flechas
    x_pare1 = 8290;   y_pare1 = 6650;                                %Pare Principal
    x_pare2 = 7440;   y_pare2 = 6100;                                %Pare Secundaria
    x_flecha_prin = [8290 6300];    y_flecha_prin = [8290 7000];     %Flecha Principal
    x_flecha_sec = [7790 6100];    y_flecha_sec = [7090 6100];       %Flecha Secundaria
    %% Presentar flechas de disponibilidad
     flechas_disponibilidad(disp_S29_Princ,disp_S29_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_
     flecha_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
    %% Enciendo LEDS
    %encenderLEDS(total_carros57,total_carros64,a);
    %% Presento disponibilidad
    if disp_S29_Princ >= disp_S29_Sec
        set(handles.salida9,'String','Antonio José de Sucre');
    else
        set(handles.salida9,'String','Vicente Rocafuerte');
    end

case 31
    %% Cargo variables del archivo VehiculosContabilizados.mat
    load VehiculosContabilizados.mat total_carros71 total_carros65 propied71 propied65...
    disp_S30_Princ disp_S30_Sec
    cla;
    %% Presento el número de vehículos y porcentaje en el text field
    set([handles.salida1 handles.salida5],'String','18 de Noviembre');
    set([handles.salida2 handles.salida6],'String','10 de Agosto');
    set(handles.salida3,'String',total_carros71);
    set(handles.salida4,'String',total_carros65);
    set(handles.salida7,'String',sprintf('%d %c',round(disp_S30_Princ),'%'));
    set(handles.salida8,'String',sprintf('%d %c',round(disp_S30_Sec),'%'));
    %% Cargo mapa guardado en el handles
    imshow(MapaLoja);
```

```
        hold off
        %% Realizo zoom a las calles
        zoom_semaforo30 = [8357 11540 4928 7279];
        axis(zoom_semaforo30);
        hold on
        %% Habilito puerto COM y pines de salida
        %a = arduino('COM4');
        %% Grafico las cajas de frontera de los vehículos
        cajasfrontera(propied71,propied65,total_carros71,total_carros65,handles);
        %% Coordenadas flechas
        x_pare1 = 9910;   y_pare1 = 5450;                              %Pare Principal
        x_pare2 = 9070;   y_pare2 = 6100;                              %Pare Secundaria
        x_flecha_prin = [9910 5800];   y_flecha_prin = [9910 5100];    %Flecha Principal
        x_flecha_sec = [9420 6100];    y_flecha_sec = [8720 6100];     %Flecha Secundaria
        %% Presentar flechas de disponibilidad
       flechas_disponibilidad(disp_S30_Princ,disp_S30_Sec,x_pare1,y_pare1,x_pare2,y_pare2,x_fl
       echa_prin,y_flecha_prin,x_flecha_sec,y_flecha_sec);
        %% Enciendo LEDS
        %encenderLEDS(total_carros71,total_carros65,a);
        %% Presento
        if disp_S30_Princ >= disp_S30_Sec
            set(handles.salida9,'String','18 de Noviembre');
        else
            set(handles.salida9,'String','Vicente Rocafuerte');
        end
end

   %% Presento imagen de semáforo
   axes(handles.semaforozoom),imshow(semaforoimagen);
   axis off
   hold on
   %% Barra de progreso
   progreso_fig = sbprogress(progreso_fig,100,'Msg','Conteo completo','DisplayMode',2);
   sbprogress(progreso_fig,'Close')
   clearvars ans progreso_fig
   %% Presentar tiempo de proceso en text_field
   tiempo = toc;
   set(handles.salida_proc,'String',sprintf('%d segundos',round(tiempo)));
   disp(tiempo)

% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)

% --- Executes on selection change in popupmenu2.
function popupmenu2_Callback(hObject, eventdata, handles)

%% SELECCIONO NODO INICIAL %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Resetear pop-up menú
set(handles.popupmenu1, 'Value', 1);
%% Presentar opciones punto de inicio
seleccion1 = get(handles.popupmenu2,'Value');

switch seleccion1
    case 1
        inicio = 0;
        uwait(msgbox('Seleccione inicio'));
    case 2
        inicio = 1;
    case 3
        inicio = 2;
    case 4
        inicio = 3;
    case 5
        inicio = 4;
    case 6
        inicio = 5;
    case 7
        inicio = 6;
    case 8
        inicio = 7;
    case 9
        inicio = 8;
    case 10
        inicio = 9;
    case 11
```

```
        inicio = 10;
    case 12
        inicio = 11;
    case 13
        inicio = 12;
    case 14
        inicio = 13;
    case 15
        inicio = 14;
    case 16
        inicio = 15;
    case 17
        inicio = 16;
    case 18
        inicio = 17;
    case 19
        inicio = 18;
    case 20
        inicio = 19;
    case 21
        inicio = 20;
    case 22
        inicio = 21;
    case 23
        inicio = 22;
    case 24
        inicio = 23;
    case 25
        inicio = 24;
    case 26
        inicio = 26;
    case 27
        inicio = 27;
    case 28
        inicio = 28;
    case 29
        inicio = 29;
    case 30
        inicio = 30;
end
%% Presentar punto de inicio seleccionado
set(handles.text_inicio,'String',inicio);
handles.inicio = inicio;
%% Guardar variable
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)

% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)

%% SELECCIONO NODO FINAL %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Resetear pop-up menú
set(handles.popupmenu1, 'Value', 1);
%% Presentar opciones punto de inicio
seleccion2 = get(handles.popupmenu3,'Value');

switch seleccion2
    case 1
        fin = 0;
        uwait(msgbox('Seleccione fin'));
    case 2
        fin = 1;
    case 3
        fin = 2;
    case 4
        fin = 3;
    case 5
        fin = 4;
    case 6
        fin = 5;
    case 7
        fin = 6;
    case 8
```

```
        fin = 7;
    case 9
        fin = 8;
    case 10
        fin = 9;
    case 11
        fin = 10;
    case 12
        fin = 11;
    case 13
        fin = 12;
    case 14
        fin = 13;
    case 15
        fin = 14;
    case 16
        fin = 15;
    case 17
        fin = 16;
    case 18
        fin = 17;
    case 19
        fin = 18;
    case 20
        fin = 19;
    case 21
        fin = 20;
    case 22
        fin = 21;
    case 23
        fin = 22;
    case 24
        fin = 23;
    case 25
        fin = 24;
    case 26
        fin = 25;
    case 27
        fin = 26;
    case 28
        fin = 27;
    case 29
        fin = 28;
    case 30
        fin = 29;
end
%% Presentar punto de inicio seleccionado
set(handles.text_fin,'String',fin);
handles.fin = fin;
%% Guardar variables
guidata(hObject, handles);

% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% Hint: popupmenu controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
   set(hObject,'BackgroundColor','white');
end

% --- Executes on button press in presentarcamino.
function presentarcamino_Callback(hObject, eventdata, handles)

%% PRESENTACIÓN DE LA RUTA CON MENOR CONGESTIÓN %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

%% Barra de progreso
progreso_fig =
sbprogress([],0,'Msg','Procesando...','Parent',CONTEOVEHICULAR_V2,'DisplayMode',2);
progreso_fig = sbprogress(progreso_fig,50,'DisplayMode',2);
tic
%% Resetear text_field
set(handles.salida_proc,'String','','BackgroundColor',[0.94 0.94 0.94]);
%% Cargo imagen del mapa de Loja y presento
MapaLoja = handles.mapa;
axes(handles.mosaico), title('Ruta descongestionada');
MapaLoja = imadjust(MapaLoja, [], [], 0.25);
```

```
imshow(MapaLoja);
hold off
%% Reseteo text_field
set([handles.salida1 handles.salida5],'String','(Calle Principal)','BackgroundColor',[0.94
0.94 0.94]);
set([handles.salida2 handles.salida6],'String','(Calle Secundaria)','BackgroundColor',[0.94
0.94 0.94]);
set([handles.salida3 handles.salida7],'String','','BackgroundColor',[0.94 0.94 0.94]);
set([handles.salida4 handles.salida8],'String','','BackgroundColor',[0.94 0.94 0.94]);
set(handles.salida9,'String','(Calle Recomendada)');
%% Cargo variables de inicio y fin
inicio = handles.inicio;
fin = handles.fin;
%% Cargo variables de porcentaje
load PorcentajeOcupacion.mat
%% Conexión de los nodos
s = [1 2 3 3 4 5 5 6 7 7 8 8 9 9 10 10 11 11 12 13 14 14 15 15 16 16 17 17 ...
     18 18 19 19 20 20 21 21 22 22 23 23 24 26 26 27 28 28 29 30 30];
t = [7 1 2 9 3 4 11 5 8 13 2 9 10 15 4 11 12 17 6 19 8 13 14 21 10 15 16 23 ...
     12 17 20 25 14 21 22 27 16 23 24 29 18 20 25 26 22 27 28 24 29];
%% Pesos de las aristas
weights = [ocup_S1_Princ ocup_S2_Sec ocup_S3_Sec ocup_S3_Princ ocup_S4_Sec...
     ocup_S5_Sec ocup_S5_Princ ocup_S6_Sec ocup_S7_Sec ocup_S7_Princ...
     ocup_S8_Princ ocup_S8_Sec ocup_S9_Sec ocup_S9_Princ ocup_S10_Princ...
     ocup_S10_Sec ocup_S11_Sec ocup_S11_Princ ocup_S12_Princ ocup_S13_Princ...
     ocup_S14_Princ ocup_S14_Sec ocup_S15_Sec ocup_S15_Princ ocup_S16_Princ...
     ocup_S16_Sec ocup_S17_Sec ocup_S17_Princ ocup_S18_Princ ocup_S18_Sec...
     ocup_S19_Sec ocup_S19_Princ ocup_S20_Princ ocup_S20_Sec ocup_S21_Sec...
     ocup_S21_Princ ocup_S22_Princ ocup_S22_Sec ocup_S23_Sec ocup_S23_Princ...
     ocup_S24_Princ ocup_S26_Princ ocup_S26_Sec ocup_S27_Sec ocup_S28_Princ...
     ocup_S28_Sec ocup_S29_Sec ocup_S30_Princ ocup_S30_Sec];
%% Presento el peso en formato con dos decimales
weights = str2num(sprintf('%10.1f\n',weights));
%% Crear el grafo dirigido
G = digraph(s,t,weights);
%% Coordenadas de los nodos
x = [1827 3430 5046 6683 8299 9919 1827 3430 5046 6683 8299 9919 1827 3430 ...
     5046 6683 8299 9919 1827 3430 5046 6683 8299 9919 1827 3430 5046 6683 8299 9919];
y = [1272 1272 1272 1272 1272 1272 2465 2465 2465 2465 2465 2465 3683 3683 ...
     3683 3683 3683 3683 4887 4887 4887 4887 4887 4887 6085 6085 6085 6085 6085 6085];
%% Presento mapa de Loja
imshow(MapaLoja), title('Ruta descongestionada');
hold on
%% Grafico el digrafo
p = plot(G,'XData',x,'YData',y,'EdgeLabel',G.Edges.Weight,'ArrowSize',10.5,'LineWidth',0.1);
%% Encuentro el camino más  descongestionado y lo presento

%% Primera ruta
   [path1,d] = shortestpath(G,inicio,fin);
   gpuArray(path1);
   highlight(p,path1,'NodeColor','r','EdgeColor','g','LineWidth',4);
   hold on
%% Segundaa ruta
   [path2,d2] = shortestpath(G,inicio,fin,'Method','positive');
   gpuArray(path2);
   highlight(p,path2,'NodeColor','r','EdgeColor','c','LineWidth',4);
   hold on
%% Tercera ruta
   [path3,d3] = shortestpath(G,inicio,fin,'Method','unweighted');
   gpuArray(path3);
   highlight(p,path3,'NodeColor','r','EdgeColor','y','LineWidth',4);
   hold on
%% Cuarta ruta
   [path4,d4] = shortestpath(G,inicio,fin,'Method','mixed');
   gpuArray(path4);
   highlight(p,path4,'NodeColor','r','EdgeColor','m','LineWidth',4), hold on

%% Aumento tamaño del nodo inicial y nodo final
highlight(p,[inicio fin],'NodeColor','g','MarkerSize',7)
%% Un solo vector
ruta1 = sum(path1);
ruta2 = sum(path2);
ruta3 = sum(path3);
ruta4 = sum(path4);
vector_ruta = [ruta1 ruta2 ruta3 ruta4];
%% Longitud vector
longitud = length(vector_ruta);
```

```
contador = 0;
%% Conteo de rutas
for i=1:longitud
  if vector_ruta(i) == ruta1
      contador = contador + 1;
  end
end
%% Presento mensaje si escogió la misma ruta
if inicio == fin
  uwait(msgbox('Ha escogido el mismo punto'));
  hold off
end

%% Presento mensaje con número de rutas
if contador == 1
     if inicio ~= fin
     msgbox('Existen cuatro rutas disponibles');
     end
elseif contador == 2
     if inicio ~= fin
     msgbox('Existen tres rutas disponibles');
     end
elseif contador == 3
     if inicio ~= fin
     msgbox('Existen dos rutas disponibles');
     end
elseif contador == 4
    if inicio ~= fin
    msgbox('Existe solo una ruta disponible');
    end
end
%% Barra de progreso
progreso_fig = sbprogress(progreso_fig,100,'Msg','Conteo completo','DisplayMode',2);
sbprogress(progreso_fig,'Close')
clearvars ans progreso_fig
%% Presento el tiempo de procesamiento
tiempo = toc;
set(handles.salida_proc,'String',sprintf('%d segundos',round(tiempo)));
disp(tiempo)

% --- Executes on button press in help.
function help_Callback(hObject, eventdata, handles)

%% Presenta la ayuda acerca de los pasos a seguir en la Guide
uiwait(helpdlg({'1) Seleccione Mapa: Existe 10 escenarios con distinto nivel de tráfico.'
''...
        '2) Vista aérea: Realiza un zoom en el semáforo escogido.' '',...
        '3) Ruta descongestionada: Presenta la ruta más descongestionada.'}));
```