

Instituto Tecnológico de Costa Rica

Escuela de Ingeniería Electrónica



**Diseño de un algoritmo para la medición del movimiento de esporas del hongo de la roya ante impactos de gotas de agua en videos de alta velocidad**

Documento de tesis sometido a consideración para optar por el el título de Máster en Ingeniería Electrónica con Énfasis en Procesamiento Digital de Señales con el grado académico de Magister Scientiae

Maestría en Ing. Electrónica con Énfasis en Procesamiento Digital de Señales

Emmanuel Fernando Madrigal Cerdas

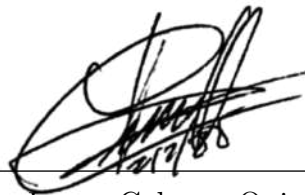
Cartago, 9 de diciembre, 2022

Instituto Tecnológico de Costa Rica  
Escuela de Ingeniería Electrónica  
Maestría Académica en Electrónica  
Trabajo Final de Graduación  
Tribunal Evaluador  
Acta de Aprobación de Tesis

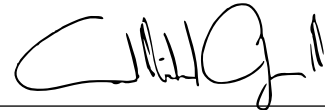
Defensa del Trabajo Final de Graduación  
Requisito para optar por el título de Máster en Ingeniería Electrónica  
Grado Académico de Magister Scientiae

El Tribunal Evaluador aprueba la defensa del Trabajo Final de Graduación denominado Diseño de un algoritmo para la medición del movimiento de esporas del hongo de la roya ante impactos de gotas de agua en videos de alta velocidad, realizado por Emmanuel Fernando Madrigal Cerdas Carné: 2014079803, y hace constar que cumple con las normas establecidas por la Unidad Interna de Posgrados de la Escuela de Ingeniería Electrónica del Instituto Tecnológico de Costa Rica.

Miembros del Tribunal



Dra. Laura Cabrera Quirós  
Profesora Lectora



M.Sc. Michael Grüner Monzón  
Profesor Lector



M.Sc. Marco Madrigal Solano  
Evaluador Independiente



Dr. José Pablo Alvarado Moya  
Profesor Asesor

Cartago, 9 de diciembre, 2022

# Resumen

El café es uno de los principales cultivos de Costa Rica desde que se inició su producción en el siglo XIX. Una de las principales enfermedades que afectan a esta planta es la roya del café, que ha tenido un importante impacto en la producción de café. Es de interés económico y social mitigar los efectos adversos que esta enfermedad tiene sobre el café.

El desarrollo de un método para medir la dispersión de las esporas de la roya bajo el impacto de las gotas de agua es el objeto del presente trabajo. Para ello, se propone un sistema que comprende la captura de datos reales con cámaras de alta velocidad, la generación de datos sintéticos a partir de estas capturas y el entrenamiento de arquitecturas para su posterior capacitación. El fenómeno que se estudia comprende el momento en que una gota de agua entra en el campo de visión de una cámara y finaliza tras impactar en la hoja y dispersar algunas esporas.

En base a esto, se entrenan dos modelos, uno para determinar la velocidad de cada píxel como flujo óptico. Un segundo modelo se utiliza para determinar si el objeto es agua, hoja, óxido de café o cualquier otro objeto.

El mejor modelo para el flujo óptico es la red GMA, que tiene un error por debajo del píxel de 0.2007; sin embargo, tiene problemas con las imágenes reales cuando se presenta el movimiento de las hojas, lo que no está presente en el conjunto de datos. Mientras que el mejor modelo para la segmentación es el que utiliza una arquitectura ConvNext con un mAcc de 0.9094 y un mIoU de 0.9536, esta red tiene problemas con la detección de esporas en imágenes reales y en zonas reflectantes de las hojas.

Todo el código para este trabajo puede ser accesado en los siguientes links: [visualizador](#), [entrenador](#), y [generador de etiquetas](#).

**Palabras clave:** Flujo óptico, Datos sintéticos, Aprendizaje profundo

# Abstract

Coffee is one of Costa Rica's main crops since its production began in the 19th century. One of the main diseases affecting this plant is coffee rust, which has had an important impact on coffee production. It is of economic and social interest to mitigate the adverse effects that this disease has on coffee.

The development of a method to measure the dispersion of rust spores under the impact of water droplets is the subject of the present work. To achieve this, a system is proposed that comprises the capture of real data with high-speed cameras, the generation of synthetic data based on these captures, and the training of architectures for subsequent training. The phenomenon under study comprises when a water droplet enters the FOV of a camera and finishes after it has impacted the leaf and dispersed some spores.

Based on this, two models are trained, one to determine the speed of each pixel as optical flow. A second model is used to determine if the object is water, leaf, coffee rust or any other object.

The best model for optical flow is the GMA network, which has a sub-pixel error of 0.2007; however, it has issues with real images when presented with movement of leaves, which isn't in the dataset. While the best model for segmentation is using a ConvNext architecture with an mAcc of 0.9094 and a mIoU of 0.9536, this network has issues with the detection of spores on real images and in reflective areas of leaves.

All code for this work can be accessed in the following links: [visualizer](#), [trainer](#), y [label generator](#).

**Keywords:** Optical flow, Synthetic data, Deep learning, Semantic Segmentation

*a mis queridos padres*

# Agradecimientos

Quiero agradecer a mis padres Katia Cerdas y Fernando Madrigal por su apoyo incondicional a lo largo de toda mi vida.

Quiero agradecer al Prof. Dr. Pablo Alvarado Moya quien me ha guiado durante el desarrollo de esta tesis. Además agradezco a los estudiantes Lic. Max Hronek y Lic. Samanta Solano, Lic. Jafet Garcia y France Quesada por su apoyo en el desarrollo e implementación de las distintas secciones de este proyecto.

Un agradecimiento especial a la empresa *RidgeRun Embedded Solutions* por financiar gran parte del aprendizaje adquirido para llevar a cabo el desarrollo de este proyecto.

Emmanuel Fernando Madrigal Cerdas

Cartago, 31 de enero de 2023

# Índice general

Índice de figuras	II
Índice de tablas	III
Siglas	V
Nomenclatura	VI
<b>1. Introducción</b>	<b>1</b>
1.1. El hongo de la roya en Costa Rica . . . . .	1
1.2. Estudio del mecanismo de dispersión de la roya . . . . .	2
1.3. Sistema para la determinación de la correlación entre los distintos factores en el mecanismo dispersión del hongo de la roya . . . . .	3
1.3.1. Montaje Experimental . . . . .	3
1.3.2. Generación de imágenes y etiquetas . . . . .	4
1.3.3. Caracterización de los distintos elementos . . . . .	4
1.4. Objetivos y estructura del documento . . . . .	5
<b>2. Marco teórico</b>	<b>6</b>
2.1. Teoría del caos . . . . .	6
2.2. Triangulación de Delauney e Interpolación . . . . .	8
2.2.1. Triangulación de Delauney . . . . .	8
2.2.2. Interpolación . . . . .	9
2.3. Proyección de puntos de tres dimensiones a dos dimensiones . . . . .	11
2.4. Redes neuronales recurrentes . . . . .	11
2.4.1. Unidades recurrentes con compuertas . . . . .	12
2.5. Flujo óptico . . . . .	13
2.5.1. Técnicas Tradicionales . . . . .	14
2.5.2. Técnicas supervisadas . . . . .	15
2.5.3. Técnicas no supervisadas . . . . .	20
2.5.4. Formato de Salida . . . . .	21
2.5.5. Métricas de evaluación . . . . .	21
2.6. Segmentación Semántica . . . . .	22
2.6.1. Backbones . . . . .	22
2.6.2. Swinnet . . . . .	23

2.6.3.	ConvNext . . . . .	23
2.6.4.	Métrica de la intersección sobre la unión . . . . .	24
<b>3.</b>	<b>Sistema de estimación de movimiento de esporas de la roya ante impacto de gotas de agua</b>	<b>26</b>
3.1.	Montaje experimental . . . . .	27
3.2.	Generación del conjunto de datos . . . . .	29
3.2.1.	Generación de imágenes base y simulación . . . . .	30
3.2.2.	Simulación inicial . . . . .	31
3.2.3.	Generación de etiqueta de segmentación . . . . .	32
3.2.4.	Generación de la máscara de flujo óptico para el objeto gaseoso . . . . .	33
3.3.	Selección de la red para la detección del flujo óptico . . . . .	34
3.4.	Métrica FAEE . . . . .	35
3.5.	Selección de la red para la segmentación de los objetos . . . . .	36
3.6.	Entrenamiento y evaluación de las redes neuronales . . . . .	36
3.6.1.	Red para la estimación del flujo óptico . . . . .	36
3.6.2.	Red para estimación de la segmentación semántica . . . . .	37
<b>4.</b>	<b>Resultados</b>	<b>38</b>
4.1.	Generación de datos . . . . .	38
4.1.1.	Datos reales . . . . .	38
4.1.2.	Generación de datos sintéticos . . . . .	39
4.2.	Verificación de las arquitecturas de aprendizaje automático . . . . .	42
4.2.1.	PWC-NET . . . . .	43
4.2.2.	SPyNet . . . . .	43
4.2.3.	LiteFlow Net . . . . .	44
4.2.4.	RAFT . . . . .	45
4.3.	Evaluaciones preliminares . . . . .	45
4.3.1.	Esporas . . . . .	47
4.3.2.	Gotas de Agua . . . . .	49
4.4.	Re-entrenamiento y validación del modelo . . . . .	49
4.4.1.	Esporas . . . . .	49
4.4.2.	Gotas de agua . . . . .	53
4.4.3.	Resultados cualitativos . . . . .	55
4.5.	Entrenamiento de una red integrada para flujo óptico . . . . .	55
4.6.	Entrenamiento de red integrada para segmentación . . . . .	63
4.7.	Visualización . . . . .	70
<b>5.</b>	<b>Conclusiones</b>	<b>71</b>
5.1.	Trabajo futuro . . . . .	73
	<b>Bibliografía</b>	<b>74</b>



# Índice de figuras

1.1.	Diagrama completo de la solución. . . . .	3
2.1.	Trayectoria generada por sistema de ecuaciones de Lorenz. . . . .	7
2.2.	Trayectoria generada por sistema de ecuaciones de Thomas. . . . .	8
2.3.	Comparación de 2 métodos de triangulación (Tomado de [15]). . . . .	9
2.4.	Punto en coordenadas baricéntricas en un triángulo. . . . .	10
2.5.	Sombreado utilizando coordenadas baricéntricas. . . . .	10
2.6.	Arquitectura gráfica de <i>Gated Recurrent Unit</i> [8] (GRU). . . . .	12
2.7.	Arquitectura de <i>Recurrent All-Pairs Field Transform</i> [55] (RAFT) (Adaptado de [55]). . . . .	16
2.8.	Arquitectura de <i>Learning to Estimate Hidden Motions with Global Motion Aggregation</i> [27] (GMA) (Tomado de [27]). . . . .	18
2.9.	Módulo de GMA (Tomado de [27]). . . . .	18
2.10.	Arquitectura de <i>Spatial Pyramid Network</i> [42] (SPyNet) para una red de tres niveles, donde $u$ es un operador de sobremuestreo y $d$ es un operador de submuestreo, $V_k$ es el flujo óptico, $G_k$ es la red convolucional y $w$ es un operador que aplica la transformación dada por el flujo óptico a una imagen. . . . .	19
2.11.	Convención de codificación de colores para la representación del flujo óptico denso (tomado de [13]). . . . .	21
2.12.	Ejemplo de representación del flujo óptico de la base de datos Sintel (tomado de [7]). . . . .	22
2.13.	Diagrama de capas de Swinnet (Tomado de [34]). . . . .	23
2.14.	Intersección (arriba) y unión (abajo) de dos áreas . . . . .	25
3.1.	Diagrama general de la solución. . . . .	27
3.2.	Diagrama del montaje experimental. . . . .	27
3.3.	Ubicación de la cámara en el montaje experimental (Tomado de [41]). . . . .	28
3.4.	Ubicación de la luz en el montaje experimental (Tomado de [41]). . . . .	28
3.5.	Diagrama general del generador de datos. . . . .	29
3.6.	Escena base renderizada. Azul representa las gotas de agua y verde las esporas. . . . .	31
3.7.	Vóxeles reimportados mostrando su area exterior y el punto donde los metadatos se ubican. . . . .	32
3.8.	Proceso de filtrado 2D de la nube y conexión de los nodos. . . . .	33

3.9. Pseudo-renderización para obtener los valores interpolados. . . . .	34
4.1. Secuencia de recuadros de esporas en movimiento capturadas a 600 fps. . .	38
4.2. Muestras del conjunto de datos de entrenamiento con sus etiquetas respec- tivas. (Tomado de [22]) . . . . .	40
4.3. Muestras del conjunto de datos de entrenamiento con sus etiquetas respec- tivas. (Tomado de [50]). . . . .	41
4.4. Muestras del conjunto de datos de entrenamiento con sus etiquetas. Rojo es agua, azul hoja, verde esporas y negro fondo. . . . .	42
4.5. Predicción PWC con cubo en eje x. (Tomado de [50]) . . . . .	43
4.6. Evaluación PWC con conjunto de datos <i>Flying Chairs</i> . A la izquierda se muestra la etiqueta y a la derecha la predicción. (Tomado de [50]) . . . . .	43
4.7. Predicción SPyNet con cubo en eje y. (Tomado de [50]) . . . . .	44
4.8. Evaluación arquitectura SPyNet. A la izquierda entrenado con <i>Karlsruhe Institute of Technology and Toyota Technological Institute dataset</i> (KITTI), al centro con <i>Flying Chairs</i> y a la derecha con <i>Sintel</i> . (Tomado de [50]) . .	44
4.9. Predicción LiteFlow Net con cubo en eje x. (Tomado de [50]) . . . . .	45
4.10. LiteFlow Net. A la izquierda entrenado con KITTI, al centro con <i>Flying Chairs</i> y a la derecha con <i>Sintel</i> . (Tomado de [50]) . . . . .	45
4.11. Predicciones RAFT en el eje y. (Tomado de [50]) . . . . .	45
4.12. Predicciones de la red RAFT para distintos conjuntos de datos. (Tomado de [50]) . . . . .	46
4.13. Estimaciones preliminares de los modelos preentrenados con mejores métri- cas. (Tomado de [22]). . . . .	48
4.14. Estimaciones de los modelos entrenados que poseen el menor <i>Filtered Ave- rage Endpoint Error</i> (FAEE). (Tomado de [22]). . . . .	51
4.15. Imágenes de referencia ( <i>groundtruth</i> ) para evaluación cualitativa. (Tomado de [22]). . . . .	52
4.16. Evaluación cualitativa utilizando el método clásico de Lucas-Kanade. (To- mado de [22]). . . . .	52
4.17. Evaluación cualitativa utilizando los modelos preentrenados. (Tomado de [22])	53
4.18. Predicción del movimiento del agua utilizando imágenes reales a 600 fps, para los modelos prueba 1, prueba 3 y prueba 5. . . . .	55
4.19. <i>Average Endpoint Error</i> (AEE) en función de la cantidad de iteraciones durante el entrenamiento. . . . .	56
4.20. Resultados de flujo óptico para las redes integradas. . . . .	57
4.21. Resultados de flujo óptico utilizando imágenes reales. . . . .	58
4.22. Inferencia de movimiento de esporas. . . . .	58
4.23. Combinaciones de hiperparámetros y AEE resultantes. Esta figura puede ser visualizada interactivamente <a href="#">aquí</a> . . . . .	59
4.24. AEE en función de la cantidad de épocas. $R = -0,393$ . . . . .	60
4.25. AEE en función del porcentaje de épocas del punto máximo de la tasa de aprendizaje. $R = 0,337$ . . . . .	60

---

4.26. AEE en función de la tasa de aprendizaje máxima. $R = 0,424$ . . . . .	61
4.27. AEE en función del tamaño del lote. $R = -0,218$ . . . . .	62
4.28. Valor medio de precisión en todas las clases en función de cantidad de iteraciones durante el entrenamiento. . . . .	63
4.29. Valor medio de IoU en todas las clases en función de cantidad de iteraciones durante el entrenamiento. . . . .	64
4.30. Resultados de segmentación. . . . .	64
4.31. Resultados de segmentación utilizando imágenes reales. . . . .	65
4.32. Resultados de segmentación. . . . .	65
4.33. Combinaciones de hiperparámetros y mIoU resultantes. Esta figura puede ser visualizada interactivamente <a href="#">acá</a> . . . . .	66
4.34. AEE en función de la cantidad del tamaño del lote. Coeficiente de correlacion para la linea de mejor ajuste $R = -0,451$ . . . . .	67
4.35. AEE en función de la cantidad de épocas. Coeficiente de correlacion para la linea de mejor ajuste $R = 0,625$ . . . . .	67
4.36. AEE en función del parámetro de potencia en el calenderizador de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste $R = -0,410$ . . . . .	68
4.37. AEE del valor máximo de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste $R = 0,486$ . . . . .	68
4.38. AEE en función del porcentaje de épocas del valor máximo de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste $R = -0,315$ . . . . .	69
4.39. Visualizacion del movimiento a lo largo de diferentes cuadros . . . . .	70

# Índice de tablas

3.1. Características de la cámara Imaging Source DFK 33UX252. . . . .	29
3.2. Características del objetivo Computar H3Z1014CS. . . . .	29
4.1. Evaluación preliminar por arquitectura para el set de datos de esporas utilizando modelos pre-entrenados. (Tomado de [22]) . . . . .	47
4.2. FAEE de la arquitectura elegida (RAFT) evaluado en el conjunto de esporas. (Tomado de [22]) . . . . .	47
4.3. Evaluación preliminar por arquitectura para el set de datos de gotas utilizando modelos pre-entrenados. (Tomado de [50]) . . . . .	49
4.4. Evaluación preliminar por arquitectura para el set de datos de gotas utilizando modelos pre-entrenados con el FAEE. (Tomado de [50]) . . . . .	49
4.5. Comparación de AEE y FAEE para modelos pre-entrenados en diferentes arquitecturas y modelos entrenados en RAFT. (Tomado de [22]). . . . .	50
4.6. Tiempo de ejecución de la evaluación de cada modelo sobre 1200 imágenes. (Tomado de [22]). . . . .	54
4.7. Valores de AEE para los modelos entrenados. (Tomado de [50]) . . . . .	54
4.8. Valores finales de AEE de la gráfica 4.19. . . . .	56
4.9. Valores máximos de mAcc y mIOU de la gráficas 4.28 y 4.29. . . . .	63



# Siglas

**AEE** *Average Endpoint Error.*

**CATIE** Centro Agronómico Tropical de Investigación y Enseñanza.

**CIRAD** Centro Francés de Investigación Agrícola para el Desarrollo Internacional.

**EPE** *EndPoint Error.*

**FAEE** *Filtered Average Endpoint Error.*

**GRU** *Gated Recurrent Unit* [8].

**KITTI** *Karlsruhe Institute of Technology and Toyota Technological Institute dataset.*

**lr** tasa de aprendizaje.

**LSTM** *Long Short Term Memory.*

**RAFT** *Recurrent All-Pairs Field Transform* [55].

**RNN** *recurrent neural network.*

**SPyNet** *Spatial Pyramid Network* [42].

# Nomenclatura

$\sigma(x)$  Función sigmoide:  $\frac{1}{1+e^{-x}}$ .

$\mathbf{v}$  Vector.

$\mathbf{v}_t$  Vector en un momento dado  $t$ .

$W$  Matriz.

$x^*$   $x$  optimo, aplica tanto para vector como para escalar.

$y$  Escalar.

# Capítulo 1

## Introducción

### 1.1. El hongo de la roya en Costa Rica

El arbusto del café se encuentra nativamente en las tierras altas de África oriental, mientras que su consumo se origina en Yemen a mediados del siglo XV. Su consumo comenzó a expandirse hacia Europa en los siglos XVII y XVIII. En el siglo XVIII el café fue transportado a Martinica con la primera cosecha obtenida en 1726, y para finales del siglo este se cultiva en Latinoamérica [59].

En Costa Rica las plantas del café se comenzaron a introducir en 1808 [19]. Para octubre de 1820 se exporta este grano por primera vez. En 1832 ya se exportaban 500 quintales anuales y para 1844 este número ascendía a 50 000 quintales. Para 1848 este número se había triplicado a 150 000 quintales, convirtiéndose en el principal producto de exportación del país [31]. En 2020 la producción alcanzó un total de 1 915 959 fanegas, beneficiando a cerca de 38 804 familias productoras [37].

El hongo de la roya es documentado por primera vez en Sri Lanka, en ese momento el tercer productor más grande del mundo; cerca de una década después la producción de esta cosecha había sido abandonada. Entre 1870 y 1990 el hongo se expandió a través de todas las regiones cafetaleras del mundo con excepción de Hawaïi [38].

El hongo de la roya fue encontrado en Brasil en 1970, y poco a poco se expandió a lo largo del resto del continente. El hongo llegó a Nicaragua en 1976. Esfuerzos iniciales fueron exitosos en contener el hongo en este país, pero cambios políticos en este país causaron que para 1982 hubiera alcanzado a México. En Costa Rica el hongo se mantuvo afuera del país hasta diciembre de 1983 [38].

Durante este periodo la producción de café no fue devastada en Latinoamérica, esto en parte gracias a zonas de producción más altas. Para aquellas regiones sobre los 1000 m no fue necesario implementar medidas de control. En la década de 2010 una segunda ola impactó nuevamente, esta vez con consecuencias mayores: la producción en Perú se redujo en un 39 %, en México un 46 % y en El Salvador un 57 %; en nuestro país, en el año



2012-2013 se tuvo un pérdida del 5% [57]. El principal causante de esta segunda ola fue el calentamiento global, causando patrones de lluvia irregulares y temperaturas mínimas más cálidas. En regiones donde el hongo previamente no tuvo una mayor afectación, ahora la cosecha fue devastada [38].

El Centro Francés de Investigación Agrícola para el Desarrollo Internacional (CIRAD), en conjunto con el Centro Agronómico Tropical de Investigación y Enseñanza (CATIE), son organizaciones que han ayudado en la lucha y adaptación en contra del hongo de la roya. Como parte de este esfuerzo se han estudiado aquellos factores que causan la dispersión del hongo; estos incluyen la cantidad de lluvia, el tiempo desde la última lluvia, la velocidad del viento y la presencia o no de sombra, lo que les ha permitido generar un modelo para predecir la cantidad de esporas liberadas [4].

## 1.2. Estudio del mecanismo de dispersión de la roya

Se han estudiado mecanismos de dispersión de la roya por medio un muestreador de esporas [4]. Este es un dispositivo que tiene una trampa para tomar las esporas dispersas en el ambiente. Estos resultados se basan en valores estadísticos dada la dispersión de distintas plantas bajo las mismas condiciones, y a pesar de que dan resultados generales del comportamiento de la dispersión de la roya, el mecanismo en particular de cómo se liberan las esporas ante distintas condiciones ambientales no es claro.

Uno de los mecanismos determinantes para esta dispersión es la cantidad de lluvia. Conforme la cantidad de lluvia se incrementa, la dispersión de la espora también; sin embargo, después de cierta cantidad de lluvia, la cantidad de dispersión comienza a reducirse nuevamente. El mecanismo mediante el cual se liberan estas esporas no es conocido, incluyendo la razón para este comportamiento no-lineal ante la cantidad de lluvia que haya caído. Una mejor comprensión de este mecanismo puede ayudar a desarrollar soluciones para mitigar la dispersión de la espora de roya. En concreto, los investigadores del CIRAD están interesados en estudiar los posibles patrones de salpicadura y cómo afectan a la dispersión.

Estudiar esta dispersión correctamente debe tomar en cuenta varios factores: primero, el tamaño de la espora en sí, el cual ronda los  $20\ \mu\text{m}$  [54] lo que dificulta su observación en tránsito. Segundo, el análisis de las gotas de agua; estas son translúcidas por lo que el obtener imágenes claras de estas no es posible, y dado que reflejan el fondo esto causa dificultad en distinguirlas. Tercero es la velocidad de este evento: una gota de agua puede alcanzar aproximadamente los  $8\ \text{m s}^{-1}$  antes de impactar una hoja [6]; parte de esta energía se dispersa en la hoja y otra parte se convierte en energía cinética en las esporas, por que estas podrían alcanzar velocidades comparables. Finalmente es necesario extraer información cuantitativa del evento, lo que requiere obtener cantidades físicas para caracterizarlo, lo cual incluye desde la velocidad de las gotas de agua y las esporas, los patrones en los cuales las esporas se dispersan, hasta la cuantificación de las esporas liberadas y

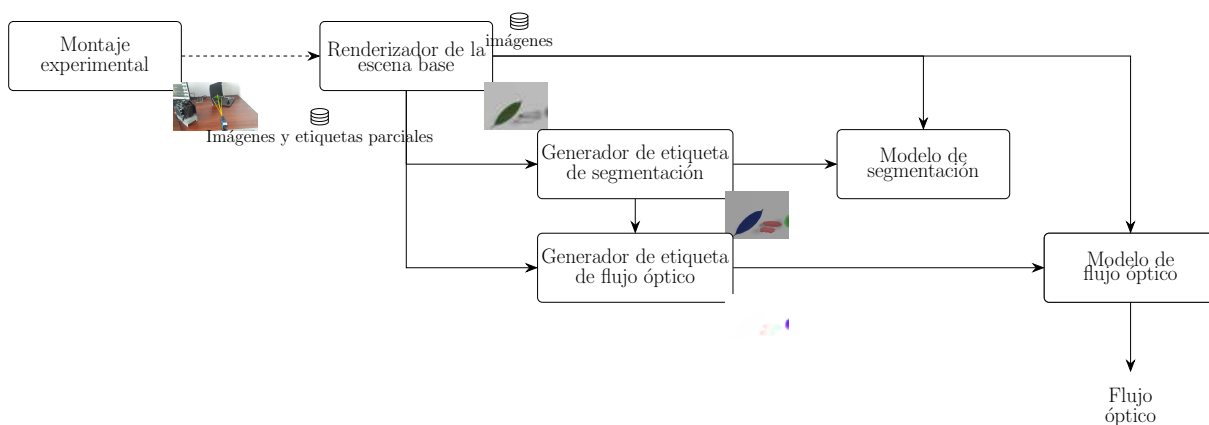
la cantidad que eventualmente alcanza otras hojas y de ser posible, correlacionar estas cantidades físicas para determinar cómo se relacionan entre sí.

En síntesis, la pregunta de investigación que motiva este trabajo es cómo construir un sistema que permita realizar mediciones cuantitativas del movimiento de esporas y gotas de agua en una sucesión de imágenes.

En el contexto de este proyecto, se estudiaron algunos subproblemas, que incluyen el flujo óptico tanto de sólo esporas como de sólo agua, la captura de imágenes de alta velocidad del impacto agua-hoja-espora, bajo un modelo de co-asesorado.

### 1.3. Sistema para la determinación de la correlación entre los distintos factores en el mecanismo dispersión del hongo de la roya

En la figura 1.1 se muestra un diagrama general de la solución propuesta a este problema.



**Figura 1.1:** Diagrama completo de la solución.

#### 1.3.1. Montaje Experimental

La primera sección es el sistema de captura. Este se encarga de obtener secuencias de imágenes de referencia donde sea posible observar las gotas de agua, las esporas y las hojas del café. Como se mencionó anteriormente, el tamaño de la espora, el material de la gota de agua, y la velocidad de ambos objetos introducen desafíos para su captura.

Como se ha demostrado en [12], [14], [44], [53] bajo las condiciones correctas es posible capturar todos los objetos correctamente, esto conlleva una serie de condiciones que incluyen:

- Captura de imágenes en alta velocidad.

- Alto contraste contra el fondo de la imagen.
- Iluminación del plano de dispersión de la espora.

### 1.3.2. Generación de imágenes y etiquetas

Para el posterior entrenamiento de un algoritmo que tenga como resultado final la cuantificación del movimiento de las esporas y las gotas de agua, se utilizan imágenes sintéticas. Estas imágenes son generadas inicialmente por un programa de renderización, las que son utilizadas directamente para el entrenamiento de los modelos de segmentación y flujo óptico; sin embargo, las etiquetas generadas por este programa requieren un procesamiento posterior para completar detalles faltantes del proceso de renderizado. Así, un módulo toma los resultados de renderizador y genera la etiqueta completa de segmentación, mientras que el segunda integra estos resultados con los resultados de renderizador para generar las etiquetas de flujo óptico.

### 1.3.3. Caracterización de los distintos elementos

Una vez que se obtienen las secuencias de imágenes con muestras de las esporas dispersándose, es necesario cuantificar todos los valores.

Para las gotas de agua se espera obtener la velocidad a la que éstas se mueven. Para obtener la velocidad de las gotas se utiliza un método de flujo óptico. Los métodos en el estado del arte para este problema [23], [52], [55] obtienen valores de desplazamiento para cada uno de los píxeles. Esto se conoce como un flujo óptico denso, por lo que será necesario conjuntar los valores del flujo óptico para obtener un valor total para la gota y no para sus distintas subregiones.

De igual manera para calcular la velocidad de dispersión de las esporas se pueden utilizar los mismos métodos de flujo óptico. Estos algoritmos son agnósticos a qué objeto experimenta movimiento, por lo que el mismo algoritmo puede ser capaz de obtener la velocidad de las esporas, así como la de las gotas.

Dado que ambos objetos se desean cuantificar de una manera separada, es necesario implementar un algoritmo capaz de distinguir a cuál de los dos objetos pertenece cada uno de los vectores de movimiento. Este problema es solucionado por un algoritmo de segmentación; los algoritmos del estado del arte [34], [35] también son capaces de obtener los valores de movimiento para cada píxel individual.

La principal contribución de esta tesis es el modelado de la nube de esporas como un objeto gaseoso y la generación de etiquetas correctas, lo que hace posible alcanzar los siguientes objetivos.

## 1.4. Objetivos y estructura del documento

El objetivo general de este proyecto es proponer un algoritmo que permita medir cuantitativamente el desplazamiento de gotas de agua y de las esporas del hongo de la roya en sucesiones de imágenes. Para alcanzarlo se identifican los siguientes objetivos específicos. En primer lugar, hay que hacer una revisión del estado del arte para estudiar el progreso actual y la viabilidad de una solución del problema a estudiar; como resultado de esto, se debe determinar un conjunto de métodos candidatos para las siguientes etapas.

En segundo lugar, se debe plantear un montaje experimental que capture el evento de dispersión; lo que debe dar como resultado imágenes en las que el evento sea capturado correctamente. En tercer lugar, hay que preparar los datos tanto para el entrenamiento como para la evaluación; lo que incluye la generación de estos datos y las etiquetas correspondientes. Por último, los métodos propuestos deberán ser evaluados con estos datos; lo que incluye la selección de los modelos, la arquitectura y cualquier optimización realizada sobre los mismos.

En el siguiente capítulo se esbozan los fundamentos teóricos necesarios para explicar en el capítulo 3 la propuesta realizada. Los resultados obtenidos se exponen en el capítulo 4 y finalmente en el capítulo 5 se dan las conclusiones obtenidas.

# Capítulo 2

## Marco teórico

El capítulo explicará los conceptos teóricos para entender la solución. La sección 2.1 introduce la teoría del caos que se utilizará para aumentar la variabilidad de los datos. La sección 2.2 introducirá la triangulación de Delauney y la interpolación utilizada para densificar el mapa de movimiento. La sección 2.3 introducirá las ecuaciones necesarias para proyectar puntos 3D en un plano 2D estas dos secciones utilizadas para generar las etiquetas. La sección 2.4 introducirá las RNN utilizadas en algunas de las redes de flujo óptico, que a su vez se explican en la sección 2.5, finalmente la sección 2.6 introducirá la segmentación semántica.

### 2.1. Teoría del caos

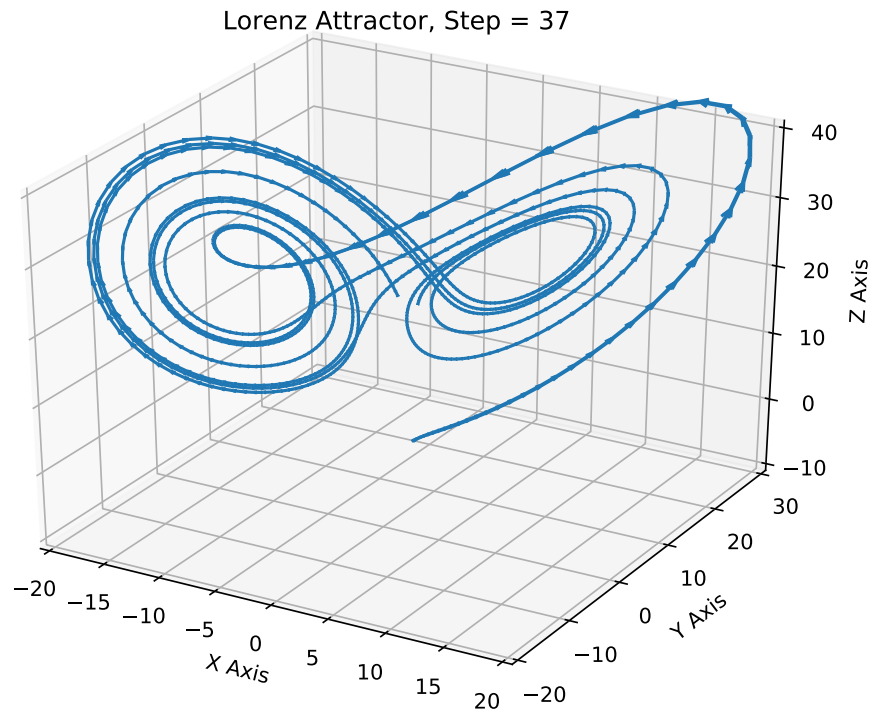
Bajo el contexto de generación de datos, es necesario asegurar que el conjunto tenga una alta variabilidad, dado que de no ser así este podría causar un sesgo en un modelo entrenado utilizando este conjunto. En particular las trayectorias de un objeto en un espacio tridimensional, deben tener alta variabilidad.

La teoría del caos es un área de estudio que se enfoca en modelar eventos caóticos, es decir sistemas dinámicos no lineales por medio de ecuaciones diferenciales. Estas ecuaciones son características porque presentan alta sensibilidad a variaciones en las condiciones iniciales. Estos sistemas presentan la intermitencia, la cual consiste en transiciones espontáneas entre dinámica no caótica y caótica [47].

Un ejemplo de un sistema dinámico caótico es el sistema de Lorenz [51] el cual está definido por:

$$\begin{aligned}
 \frac{dx}{dt} &= \sigma(y - x) \\
 \frac{dy}{dt} &= \rho x - y - xz \\
 \frac{dz}{dt} &= xy - \beta z
 \end{aligned}
 \tag{2.1}$$

donde  $x$ ,  $y$  y  $z$  son las variables correspondientes a la posición en el espacio tridimensional donde se generará la trayectoria caótica y  $\sigma$ ,  $\rho$  y  $\beta$  son los parámetros que denotan el número de Prandtl, el número de Rayleigh y el factor geométrico. Estos parámetros son los que determinan si el sistema es caótico o no. Los valores donde el sistema de Lorenz es caótico son:  $\sigma = 3$ ,  $\rho = 28$  y  $\beta = \frac{8}{3}$  [51]. Al cambiar las condiciones iniciales del sistema de ecuaciones, siempre se generará una trayectoria caótica distinta. En la figura 2.1 se ejemplifica la generación de una trayectoria tridimensional utilizando el sistema de Lorenz con las condiciones iniciales de:  $x = y = z \in [0, 3]$ .

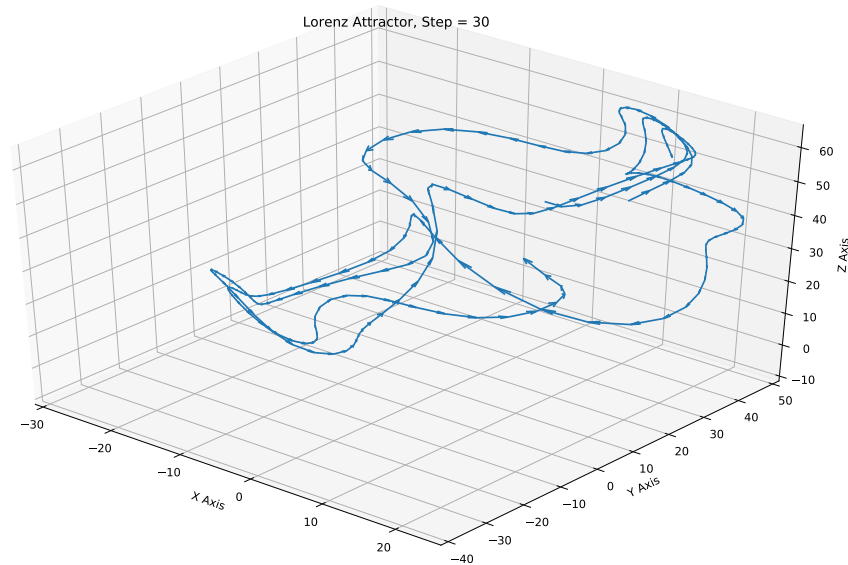


**Figura 2.1:** Trayectoria generada por sistema de ecuaciones de Lorenz.

Adicionalmente existe otro sistema de dinámico conocido como el atractor cíclicamente simétrico de Thomas. Este sistema se define como:

$$\begin{aligned}
 \frac{dx}{dt} &= \text{sen}(y) - bx \\
 \frac{dy}{dt} &= \text{sen}(z) - by \\
 \frac{dz}{dt} &= \text{sen}(x) - bz
 \end{aligned}
 \tag{2.2}$$

donde  $b$  es el parámetro que indica qué tan disipativo es el sistema. Para  $b > 1$  el origen es el estado de equilibrio del sistema. Para  $b \approx 0,208186$  el sistema se vuelve caótico [29]. En la figura 2.2 se ejemplifica la trayectoria generada para las mismas condiciones iniciales para el ejemplo del sistema de Lorenz.



**Figura 2.2:** Trayectoria generada por sistema de ecuaciones de Thomas.

Caracterizar de manera cualitativa y cuantitativa la intermitencia de un sistema dinámico específico con base en el análisis de datos experimentales permite estudiar problemas donde no se conoce con exactitud, o bien, en lo absoluto las ecuaciones que gobiernan el comportamiento de un fenómeno específico. Esto es útil en aplicaciones de ámbito económico, biológico e incluso en el campo de la medicina [47].

## 2.2. Triangulación de Delauney e Interpolación

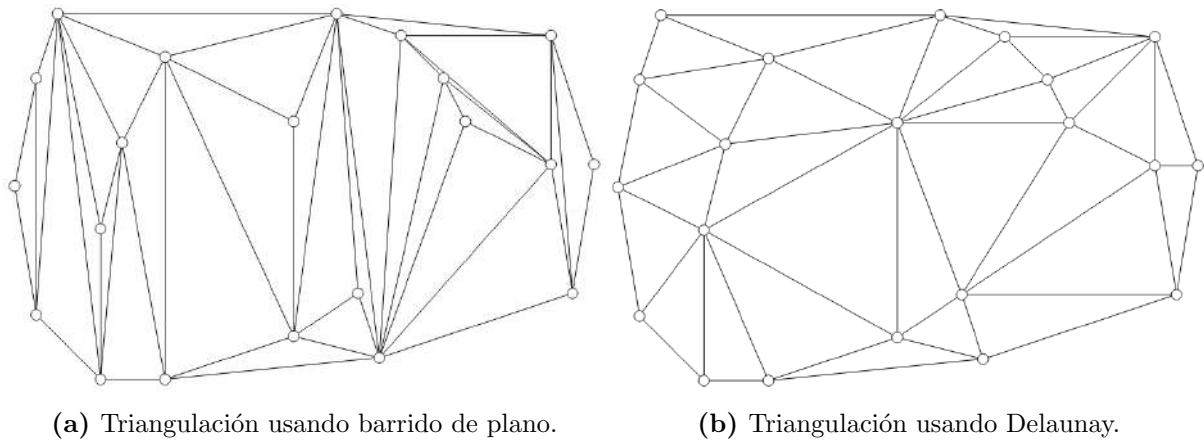
En el contexto de convertir la información discreta de movimiento tridimensional en un mapa de flujo bidimensional, se utilizan las técnicas descritas en esta sección para poder “densificar” el mapa de movimiento.

### 2.2.1. Triangulación de Delauney

Una triangulación, bajo el contexto de gráficos por computador, consiste en encontrar una serie de triángulos que cubren una superficie sin tener aristas que se intersequen las unas a las otras. Para una misma serie de puntos puede existir una gran cantidad de soluciones que cubran una superficie.

La triangulación de Delaunay de un conjunto de puntos es una colección de aristas que satisface una propiedad de “círculo vacío”: para cada arista se busca un círculo que contenga

los puntos finales de la arista pero que no contenga ningún otro punto. Esto maximiza el tamaño de los ángulos de la triangulación [32]. Esta maximización del ángulo se ilustra en la figura 2.3.



**Figura 2.3:** Comparación de 2 métodos de triangulación (Tomado de [15]).

### 2.2.2. Interpolación

**Coordenadas Baricéntricas** Las coordenadas baricéntricas se utilizan para expresar la posición de un punto dentro de un triángulo, utilizando tres escalares, como se muestra en la figura 2.4. Los tres vértices tienen las coordenadas unitarias:

$$\begin{aligned} A &= (1, 0, 0) \\ B &= (0, 1, 0) \\ C &= (0, 0, 1) \end{aligned} \tag{2.3}$$

Para cualquier punto dentro de estas tres coordenadas, su posición se expresa mediante la ecuación:

$$D = uA + vB + wC \tag{2.4}$$

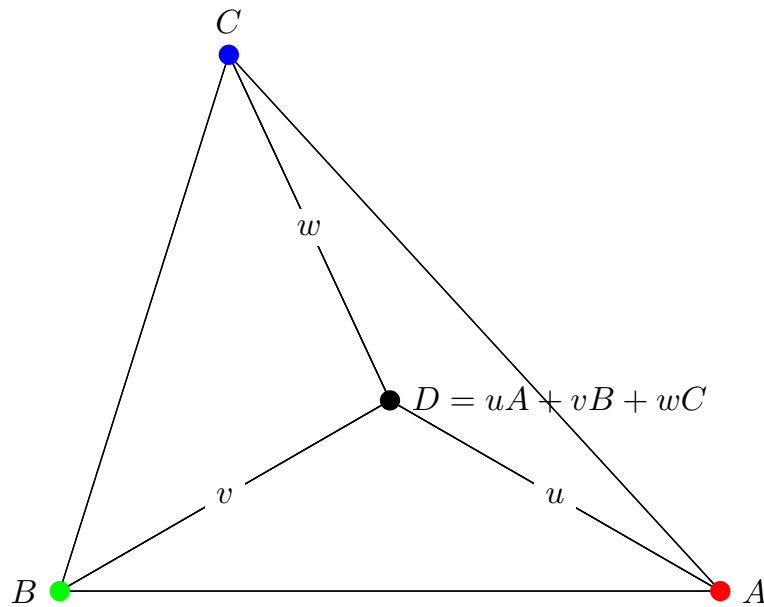
Los valores  $u, v, w$  suman a un valor total de 1:

$$1 = u + v + w \tag{2.5}$$

Una de las tres variables se puede expresar en términos de las otras dos, para así expresar los dos grados de libertad dentro del plano;

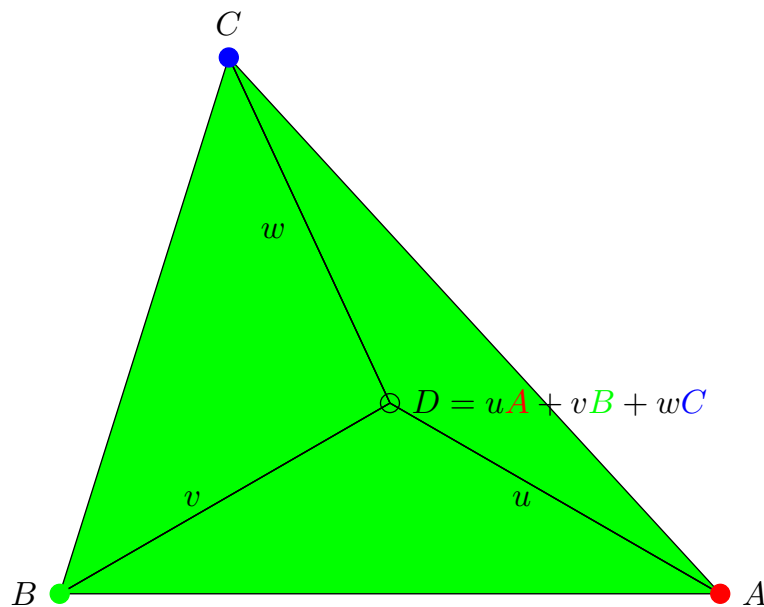
$$w = 1 - u - v \tag{2.6}$$





**Figura 2.4:** Punto en coordenadas baricéntricas en un triángulo.

En gráficos por computador se utilizan las propiedades de estas coordenadas para realizar la operación de sombreado. Si la coordenada del píxel a colorear corresponde con uno de los vértices, el valor del píxel es simplemente el valor conocido del vertex. Si se pueden determinar los valores de todos los píxeles mediante coordenadas baricéntricas, se utiliza la ecuación 2.4, pero en lugar de asignar los valores unitarios de las coordenadas a  $A, B, C$  se asignan los valores de los colores a estas variables para obtener el valor del color en el píxel de destino (ver figura 2.5).



**Figura 2.5:** Sombreado utilizando coordenadas baricéntricas.

Existen algoritmos para la realización eficiente del sombreado que cubren desde el uso de

superficies planas [18], [40] hasta el uso de ray tracing [1] donde el valor de un píxel puede estar controlado por múltiples superficies y texturas.

## 2.3. Proyección de puntos de tres dimensiones a dos dimensiones

El renderizador de objetos representa las escenas utilizando vertices y superficies en coordenadas 3D. En esta sección se detallan las operaciones necesarias para proyectar esto en un sistema de coordenadas 2D para que coincida con las imágenes generadas.

Puntos tridimensionales se proyectan a un espacio de dos dimensiones utilizando geometría proyectiva:

$$\begin{bmatrix} x' \\ y' \\ z \end{bmatrix} = \begin{bmatrix} p_1 & p_2 & p_3 & p_4 \\ p_5 & p_6 & p_7 & p_8 \\ p_9 & p_{10} & p_{11} & p_{12} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \quad (2.7)$$

Utilizando esta ecuación se proyecta el punto en coordenadas homogéneas  $[X \ Y \ Z \ 1]^T$  a sus coordenadas proyectivas  $[x' \ y' \ z]^T$ , donde  $z$  es la distancia del punto focal de la cámara al píxel. Para obtener las coordenadas en el plano bidimensional  $(x,y)$  se utiliza la ecuación:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x'/z \\ y'/z \end{bmatrix} \quad (2.8)$$

## 2.4. Redes neuronales recurrentes

En general, las redes neuronales artificiales se utilizan para estimar modelos de clasificación y regresión estáticos entre una entrada y una salida y se describen en detalle en [17]. Por otro lado, para lidiar con información temporal se usa la *recurrent neural network* (RNN), la cual se utiliza como parte de las redes de flujo óptico *Recurrent All-Pairs Field Transform* [55] (RAFT) y *Learning to Estimate Hidden Motions with Global Motion Aggregation* [27] (GMA) detalladas en la sección 2.5.2.

Estas redes son similares a las redes neuronales alimentadas hacia adelante, pero capaces de manejar secuencias de entrada de longitud variable. Esto se logra al tener un estado oculto cuya activación con respecto a una entrada depende de la entrada anterior [9]. Formalmente dada una secuencia de entrada  $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_r)$ , la RNN actualiza su estado oculto  $\mathbf{h}_t$  de la siguiente manera:

$$\mathbf{h}_t = \begin{cases} 0, & t = 0 \\ \Phi(\mathbf{h}_{t-1}, \mathbf{x}_t), & t > 0 \end{cases} \quad (2.9)$$

donde  $\phi$  es una función no-lineal.

Tradicionalmente la actualización del estado oculto en (2.9) se implementa mediante:

$$\Phi(\mathbf{h}_{t-1}, \mathbf{x}_t) = g(W\mathbf{x}_t + U\mathbf{h}_{t-1}) \quad (2.10)$$

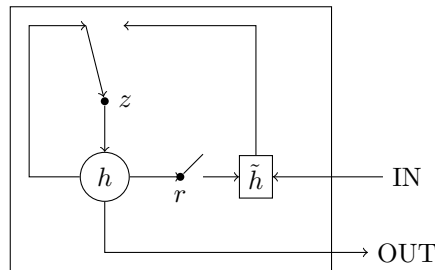
donde  $W$  y  $U$  son los pesos aplicados a la entrada y al estado oculto respectivamente, y  $g$  es la función de activación.

Existe un problema con aplicar este enfoque directamente, dado que es difícil entrenar a las RNN a capturar dependencias temporales a largo plazo, dado que los gradientes tienden a desvanecerse [3].

Una manera de evitar esto es mediante de una función de activación más compleja, la cual consiste en una transformación afín seguida por uno o más elementos no lineales implementados mediante el uso de unidades de activación. Esto en conjunto con el estado interno, permite dar distintos valores de salida a la misma entrada, según el estado interno, o incluso ignorar completamente uno de estos. Un ejemplo de esto son las unidades de memoria larga de corto plazo (*Long Short Term Memory* (LSTM)) [20] o de mayor interés para este trabajo es el uso de una unidad recurrente con compuerta (*Gated Recurrent Unit* [8] (GRU)).

### 2.4.1. Unidades recurrentes con compuertas

Las GRU fueron propuestas para hacer que cada unidad recurrente capture de forma adaptativa dependencias de diferentes escalas de tiempo. El GRU tiene unidades de enlace que modulan el flujo de información dentro de la unidad. Su arquitectura general se muestra en la figura 2.6.



**Figura 2.6:** Arquitectura gráfica de GRU.

El estado interno  $\mathbf{h}_t$  de una GRU al instante  $t$  con  $j$ -ésima componente  $h_t^j$  es una interpolación lineal entre el estado interno anterior  $\mathbf{h}_{t-1}$  y el candidato a estado interno  $\tilde{\mathbf{h}}_t$ :

$$h_t^j = (1 - z_t^j) h_{t-1}^j + z_t^j \tilde{h}_t^j \quad (2.11)$$

donde cada componente  $z_t^j$  de la *compuerta de activación* decide cuánto va a actualizar la unidad su activación o su contenido. La  $j$ -ésima compuerta de activación viene dada por:

$$z_t^j = \sigma (W_z \mathbf{x}_t + U_z \mathbf{h}_{t-1})^j \quad (2.12)$$

La  $j$ -ésima componente  $\tilde{h}_t^j$  del candidato de activación está compuesta de manera similar a la de una unidad recurrente tradicional:

$$\tilde{h}_t^j = \tanh (W \mathbf{x}_t + \mathbf{r}_t \odot (U \mathbf{h}_{t-1}))^j \quad (2.13)$$

donde  $\mathbf{r}_t$  es un conjunto de compuertas de reinicio y  $\odot$  es el producto de Hadamard. Cuando está cerrada ( $r_t^j$  cercano a 0), la compuerta de reinicio hace a la unidad actuar como si estuviera leyendo el primer símbolo de la secuencia de entrada, permitiéndole “olvidar” los estados anteriores.

La compuerta de reinicio se calcula de una manera similar a la compuerta de activación:

$$r_t^j = \sigma (W_r \mathbf{x}_t + U_r \mathbf{h}_{t-1})^j \quad (2.14)$$

Las GRU, debido a su capacidad de utilizar secuencias temporales, son un componente de las redes neuronales descritas en la sección 2.5.2.

## 2.5. Flujo óptico

El campo de movimiento en una imagen es el movimiento real del objeto en una escena 3D proyectada sobre el plano de la imagen. El flujo óptico se define como el “flujo” de los valores de gris en el plano de la imagen, es decir, lo que se observa. El flujo óptico y el campo de movimiento son iguales cuando los objetos no cambian su iridiscencia conforme los objetos se mueven en la escena [25].

Históricamente un primer enfoque al cálculo del flujo óptico consistió en el rastreo de características para estimar movimiento. Dado que estas características se obtienen solo para ciertos puntos en la imagen, se obtiene un flujo óptico que se conoce como disperso [46]. Estos métodos han evolucionado a no utilizar rastreo para su cálculo pero, la cantidad de puntos sigue siendo un subconjunto de la imagen completa. En contraste, un método en el cual todos los puntos tienen un vector de movimiento correspondiente tiene como salida un flujo óptico denso.

## 2.5.1. Técnicas Tradicionales

### Lucas-Kanade

El método de Lucas-Kanade es un método ampliamente utilizado para el cálculo del flujo óptico [36]. El principio de este algoritmo se basa en alinear una imagen  $T(\mathbf{x})$  a otra imagen  $I(\mathbf{x})$ . El cálculo del flujo óptico se realiza sobre subregiones pequeñas (por ejemplo  $5 \times 5$ ) donde  $I(\mathbf{x})$  es la imagen en  $t = 1$  y  $T(\mathbf{x})$  es la imagen en  $t = 2$  [2]. Estas ecuaciones asumen una coherencia local, lo cual quiere decir que píxeles en regiones lo suficientemente pequeñas se van a mover en conjunto.

Para que la estimación del flujo óptico vaya más allá de estas regiones, se utiliza un enfoque piramidal. Este enfoque consiste en lo siguiente: primero se submuestran ambas imágenes a una resolución baja equivalente al tamaño de la subregion de operación, y se aplica el cálculo de flujo óptico sobre esta región. Luego se obtiene la imagen un grado de submuestreo antes, es decir con una mayor resolución, y se aplica el submuestreo sobre estas subregiones de esta imagen. Para que no se obvие el resultado obtenido por el paso anterior, la imagen se transforma de manera que se compense el flujo óptico calculado. Este proceso se itera hasta que finalmente en el último submuestreo se tienen las regiones pequeñas. Finalmente se acumula el resultado para cada paso intermedio para obtener el valor del flujo óptico [5].

Con este algoritmo, para las representaciones a los niveles más altos (aquellas con el menor grado de submuestreo) se tiene un resultado muy ruidoso, por lo que por lo general se detiene antes de este punto. Dado que esta capa contiene la información para movimiento de píxeles individuales, el resultado es disperso, es decir no se tiene un valor de flujo óptico para cada píxel, sino solo para un subconjunto submuestreado de este.

### Horn-Schunk

Una alternativa al método de Lucas-Kanade es el método de Horn-Schunk [21]. Este es un método global. En lugar de calcular el flujo óptico para regiones pequeñas, se calcula utilizando una restricción de suavidad sobre el flujo óptico computado. Este método asume que si el desplazamiento entre dos cuadros es lo suficientemente pequeño el valor de gris del píxel en movimiento no cambiará drásticamente. Esta suposición de suavidad lleva a la conclusión de que puntos cercanos tienen una velocidad similar entre ellos.

De igual manera que para el Lucas-Kanade, se utiliza un método piramidal para calcular un flujo óptico refinado.

### Limitaciones

Ambos algoritmos calculan el flujo óptico a través de gradientes de la intensidad de las imágenes. Dado que el gradiente representa diferencias locales, en casos donde existan

movimientos de gran amplitud, la precisión de estos algoritmos disminuye. Esto se reduce con el uso de pirámides para calcularlo o utilizando una tasa alta de cuadros por segundo [24].

### 2.5.2. Técnicas supervisadas

Los métodos tradicionales formulan el cálculo del flujo óptico como un problema de optimización cuyo objetivo es encontrar las correspondencias entre dos recuadros consecutivos. Este enfoque es utilizado en conjunto con las redes neuronales profundas para aprender la estimación del flujo óptico, donde las redes aprenden a producir directamente un flujo óptico denso [58]. En las siguientes secciones se detallan algunos de los métodos más exitosos para el flujo óptico.

#### RAFT

El método de *RAFT* busca atacar el problema de refinamiento al tener un espacio de búsqueda donde todos los píxeles están relacionados con todos los otros píxeles. Esta arquitectura se divide en tres componentes:

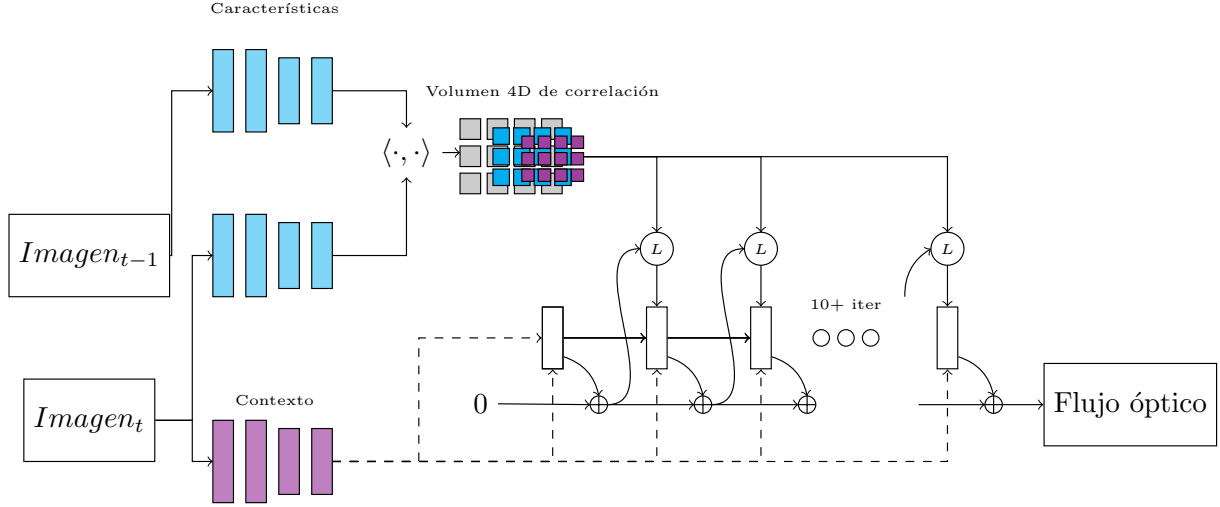
1. Un codificador de características, el cual extrae un vector de características para cada pixel.
2. Una capa de correlación que produce un hipervolumen 4D para todos los pares de píxeles, con *pooling* subsecuente para producir volúmenes en resoluciones menores.
3. Un operador de actualización basado en una red recurrente GRU, que obtiene valores de los volúmenes en la capa de correlación e iterativamente actualiza un campo de flujo inicializado en cero.

En el último paso se usa como alternativa de inicialización el resultado del flujo del recuadro anterior (en el caso de videos), esto tomando en cuenta que flujos consecutivos va a tener una alta correlación el uno con el otro.

Esta arquitectura se muestra en la figura 2.7.

Este método se basa en enfoques tradicionales que utilizan optimizaciones iterativas sobre el resultado del flujo óptico. El codificador de características extrae características por pixel. La capa de correlación calcula similitud visual entre píxeles, y el operador de actualización cumple la función de los pasos iterativos de un algoritmo optimizador. Pero a diferencia de los enfoques tradicionales, las características y los movimientos resultantes no son determinados “manualmente” sino calculados por el codificador y el operador de actualización respectivamente.

A diferencia de otros métodos que utilizan redes neuronales, RAFT mantiene y actualiza un único flujo de campo a una resolución alta. Esto evita errores comunes en estas redes



**Figura 2.7:** Arquitectura de RAFT (Adaptado de [55]).

tal como lo es la dificultad de recuperarse de errores a resoluciones gruesas, la tendencia de ignorar objetos pequeños que se mueven rápido, y una alta cantidad de iteraciones requeridas para una cascada multi-etapa. La otra característica a tomar en cuenta, es que el operador de actualización es liviano y recurrente por lo que no incrementa significativamente el tiempo requerido para obtener el resultado.

### Extractor de características

La extracción se realiza mediante una red convolucional. Esta misma red se utiliza dos veces para obtener los mapas de las imágenes  $I_t$  e  $I_{t+1}$  y mapea las imágenes a mapas de características densos a un menor tamaño; la salida es a un  $1/64$  del tamaño:  $g_{\Theta} : \mathbb{R}^{H \times W \times 3} \rightarrow \mathbb{R}^{H/8 \times W/8 \times D}$ , donde  $D = 256$  es determinado como el valor óptimo por los autores. Adicionalmente se utiliza una red de contexto separada, la cual extrae características únicamente de la primera imagen  $I_t$ .

### Cálculo de similitud visual

Un volumen de correlación se obtiene para todos los pares de píxeles, dados los mapas de características  $g_{\Theta}(I_t) \in \mathbb{R}^{H \times W \times D}$  y  $g_{\Theta}(I_{t+1}) \in \mathbb{R}^{H \times W \times D}$ . El volumen de correlación se calcula al tomar el producto punto entre todos los pares de vectores de características. Para todos los vectores esto se expresa mediante una multiplicación de matrices:

$$\mathbf{C}(g_{\Theta}(I_t), g_{\Theta}(I_{t+1})) \in \mathbb{R}^{H \times W \times H \times W} \quad (2.15)$$

Estos volúmenes de correlación son adicionalmente muestreados mediante un *polling* en sus últimas dos dimensiones con kernel de tamaño 1, 2, 4 y 8, y *strides* equivalentes, para obtener un set  $\mathbf{C}^1, \mathbf{C}^2, \mathbf{C}^4, \mathbf{C}^8$ , de manera que cada volumen  $\mathbf{C}^k$  tenga un tamaño  $H \times W \times H/2^k \times W/2^k$ . Esto permite tener información de desplazamientos grandes para los  $k$  altos donde la imagen está altamente submuestreada. Además como cada  $\mathbf{C}$  mantiene en sus primeras dos dimensiones el tamaño de la imagen, esto da información

en alta resolución, lo que permite visualizar objetos pequeños con un rápido movimiento.

Estos mapas de correlación adicionalmente se utilizan en conjunto con un operador de búsqueda  $L_C$  el cual genera un mapa de características indexando de la pirámide de correlación. Dado el estimado actual del flujo óptico se mapea cada pixel  $\mathbf{x} \in I_t$  a su correspondencia en  $I_{t+1} : \mathbf{x}' = (u + f^1(u, v), v + f^2(u, v))$ , y luego se define una región local alrededor de  $\mathbf{x}'$ :

$$\mathcal{N}(\mathbf{x}')_r = \{\mathbf{x}' + \mathbf{dx} | \mathbf{dx} \in \mathbb{Z}^2, \|\mathbf{dx}\|_1 \leq r\} \quad (2.16)$$

Esto se realiza para todos los niveles de la pirámide lo que permite obtener información de un radio más grande a los niveles más bajos. Los valores de cada nivel son luego concatenados en un solo mapa de características.

### Actualización iterativa

El operador de actualización estima una secuencia de flujos ópticos  $\mathbf{f}_1, \dots, \mathbf{f}_N$  desde un punto inicial  $\mathbf{f}_0 = \mathbf{0}$ , donde cada iteración lo que estima es una actualización a la dirección  $\Delta \mathbf{f}$ . El operador de actualización se entrena de manera que realice actualizaciones sobre la secuencia, de manera que la secuencia converja a un punto fijo  $\mathbf{f}_k \rightarrow \mathbf{f}^*$ .

Esta actualización se realiza mediante el uso de una *gated activation unit* basada en una celda GRU. Esta recibe como entrada además de su estado anterior, la concatenación de las características de correlación, y el flujo y contexto definidas en las dos etapas anteriores.

La salida del estado oculto del GRU es pasado a través de dos capas convolucionales para predecir la actualización del flujo óptico  $\Delta \mathbf{f}$ . Este tiene un tamaño de  $1/64$  de la imagen de entrada. Este resultado se sobremuestra a la resolución de entrada para tener un mapa de flujo óptico denso.

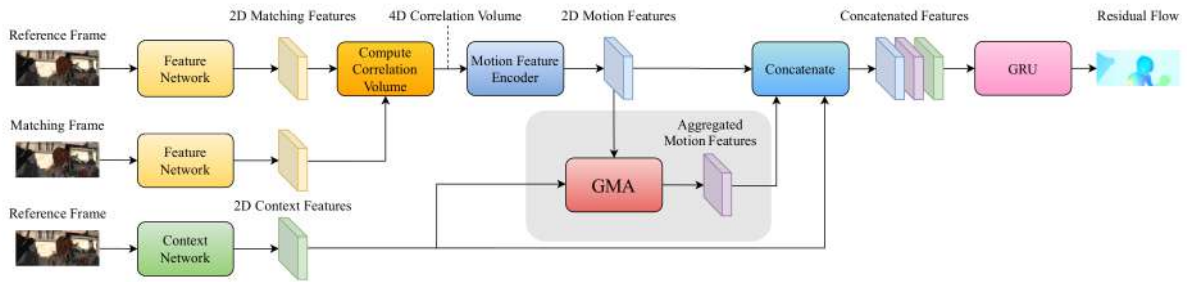
Alternativamente se utiliza  $\mathbf{f}_{0_t} = \mathbf{f}_{t-1}^*$  como punto inicial. Esto asume que el movimiento es suave entre recuadros consecutivos.

### GMA

El método de GMA se basa en RAFT para atacar el problema de oclusión, donde un punto ocluido es aquel que se visualiza en el fotograma de referencia pero no en el siguiente, lo que supone una ligera sobrecarga de la definición estándar, ya que también incluye los puntos que se mueven fuera del fotograma. La arquitectura de esta red se observa en la figura 2.8.

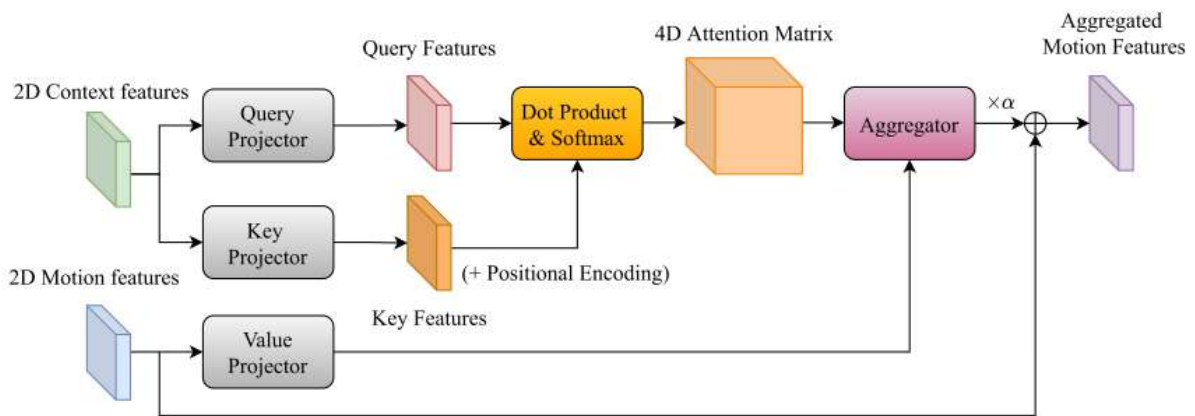
Este método toma inspiración en los transformadores [56]. A diferencia del mecanismo de autoatención de los transformadores, donde la consulta, la clave y el valor proceden de los mismos vectores de características, este método utiliza una variante generalizada de la atención. Sus características de consulta y clave son proyecciones del mapa de características del contexto, que se utilizan para modelar las autosimilitudes de la apariencia





**Figura 2.8:** Arquitectura de GMA (Tomado de [27]).

en el fotograma 1. Las características de valor son proyecciones de características de movimiento, que en sí mismas son una codificación del volumen de correlación 4D. La matriz de atención calculada a partir de las características de consulta y de las características clave se utiliza para agregar las características de valor, que son representaciones ocultas de los movimientos. Esto se denomina módulo de agregación global de movimiento (GMA por sus siglas en inglés). Las características de movimiento agregadas se concatenan con las características de movimiento locales, así como con las características de contexto, que serán decodificadas por el GRU. Todo este módulo se muestra en la figura 2.9. Este módulo a su vez se integra dentro de la arquitectura de RAFT para mejorar su rendimiento.



**Figura 2.9:** Módulo de GMA (Tomado de [27]).

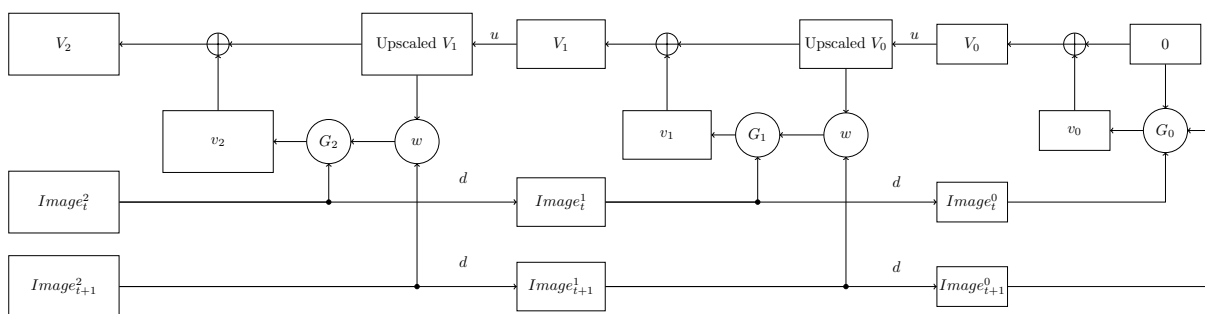
## SPyNet

La red *Spatial Pyramid Network* [42] (SPyNet) sigue un enfoque de fino a grueso utilizando pirámides espaciales. En cada nivel de la pirámide la red trata de aprender a través de sus filtros convolucionales aquellos desplazamientos de píxeles que sean menores al tamaño del filtro. Para los niveles más bajos esto representa píxeles, mientras que conforme se incrementan los niveles de la pirámide, el submuestreo de los píxeles en estos niveles representará movimientos de una mayor magnitud. La imagen “objetivo” es transformada hacia la imagen “origen” utilizando el flujo actual antes de cambiar niveles de pirámide.

Esta red utiliza una red convolucional para predecir el incremento del flujo a un nivel dado, en lugar de una función objetivo como sería en el caso de un método tradicional. La red es entrenada para predecir el flujo a cada uno de los niveles, y este flujo es “añadido” a la red en el nivel superior. Esto asume que el desplazamiento es menor al tamaño del filtro en cada nivel de la pirámide.

Al obviar la optimización iterativa de las funciones tradicionales, lo cual requeriría ejecutar la inferencia de la red múltiples veces, esta red es computacionalmente eficiente de operar. Sin embargo, el uso del enfoque piramidal presenta problemas para predecir movimientos de una alta magnitud.

La red está compuesta por cinco “subredes”. Cada una de estas tiene como tarea predecir el movimiento a un nivel dado. Dado que estas redes tienen una tarea relativamente simple (predecir la actualización de movimientos de baja magnitud en oposición al cálculo del flujo óptico completo), estas son pequeñas. Los autores de este artículo solo utilizan cinco capas convolucionales en cada una de estas subredes. La arquitectura general de esta red se ilustra en la figura 2.10.



**Figura 2.10:** Arquitectura de SPyNet para una red de tres niveles, donde  $u$  es un operador de sobremuestreo y  $d$  es un operador de submuestreo,  $V_k$  es el flujo óptico,  $G_k$  es la red convolucional y  $w$  es un operador que aplica la transformación dada por el flujo óptico a una imagen.

## FlowNet y FlowNet 2.0

FlowNet [13] es una red que calcula el flujo óptico de extremo a extremo, es decir, una misma red convolucional recibe como entrada el par de imágenes y a la salida se obtiene directamente el flujo óptico.

FlowNet es de los primeros métodos en aplicar exitosamente redes neuronales profundas al problema del flujo óptico. El problema de aplicar estas redes, especialmente aquellas con características convolucionales, proviene de que no solo es necesario extraer características de la imagen, sino que la red también debe ser capaz de correlacionar estas características en distintas regiones de la imagen.

Para procesar el par de imágenes los autores proponen dos planteamientos para las redes. En el primero de ellos se concatenan ambas imágenes de manera que si en la entrada se

tienen dos imágenes  $I_t, I_{t+1} \in \mathbb{R}^{W \times H \times C}$ , la entrada de la red será  $I_{t,t+1} \in \mathbb{R}^{W \times H \times 2C}$ . La segunda opción consiste en utilizar dos redes individuales que obtienen características para cada una de las imágenes, para finalmente pasar cada uno de estos resultados a través de una capa de “correlación” la cual realiza comparaciones de parches multiplicativos entre los dos mapas de características, para luego pasar a través de más capas convolutivas hasta llegar a la salida.

Finalmente ambas capas son alimentadas a una etapa de expansión, donde el componente principal de la parte de expansión son las capas *upconvolutional*, que consisten en un sobremuestreo y una convolución. Para realizar el refinamiento, se aplica esta capa a los mapas de características y el resultado se concatena con los mapas de características correspondientes de la parte “contractiva” de la red.

FlowNet 2.0 [23] expande la idea de FlowNet, pero utiliza un enfoque recursivo para estimar movimientos grandes. Una vez que se procesa un par de imágenes una primera vez por una red, una de las imágenes se transforma para compensar el movimiento ya estimado y se utiliza una segunda red para reestimar el flujo óptico. Este proceso se repite una vez más antes de sumar todos los flujos intermedios y obtener el total.

### 2.5.3. Técnicas no supervisadas

Dado que la mayoría de las redes neuronales profundas están construidas para predecir el flujo utilizando dos recuadros consecutivos y utilizando un entrenamiento supervisado, esto requiere una gran cantidad de datos de entrenamiento para obtener un rendimiento razonable. Esto es fácil de obtener para sets de datos basados en sucesiones sintéticas (por ejemplo el set de datos de *Sintel* [7] o *Chairs* [13]); sin embargo los videos del mundo real generalmente son difíciles de anotar para generar las etiquetas [26].

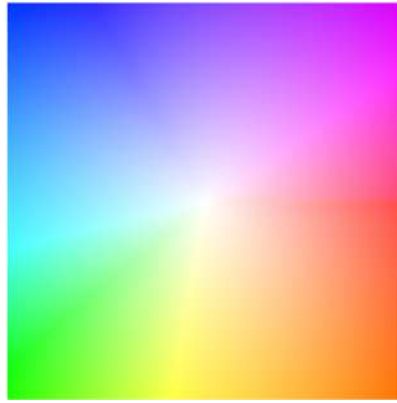
El aprendizaje no supervisado es utilizado para utilizar los recursos de videos sin etiquetar de manera que el aprendizaje se pueda realizar directamente en videos de movimiento real. La estrategia general detrás de esos métodos no supervisados es que en lugar de entrenar directamente las redes neuronales con el flujo de referencia, utilizan una medida de la diferencia entre la imagen objetivo y la imagen posterior deformada (inversamente) según el campo de flujo denso predicho de las redes plenamente convolucionales. Esto permite entrenar las redes con una gran cantidad de pares de imágenes sin etiquetar, superando la limitación de la falta de anotaciones de flujo de verdad del terreno [58].

Sin embargo, el desempeño de los métodos no supervisados todavía es menor en comparación con sus los métodos supervisados. Los principales problemas que los algoritmos no supervisados encuentran es la oclusión y movimientos de gran amplitud [43].

### 2.5.4. Formato de Salida

El formato utilizado por todas estas redes para codificar el flujo óptico entre un par de imágenes  $I_t, I_{t+1} \in \mathbb{R}^{W \times H \times C}$  viene especificado por una matriz  $F \in \mathbb{R}^{W \times H \times 2}$ , donde las dos dimensiones de profundidad representan las coordenadas  $u, v$  del vector de flujo óptico.

En casos donde no todos los pixeles contengan magnitud en las componentes  $u, v$  se pueden representar estos vectores de desplazamiento como flechas sobre la imagen. En casos densos, donde es de interés el desplazamiento de todos los pixeles, se opta por el código de colores ilustrado en la figura 2.11 donde la saturación del color representa la magnitud y el matiz el ángulo del flujo. El centro del cuadro es el origen y si el vector no tiene magnitud entonces el pixel correspondiente se representa con el color blanco. Si la magnitud normalizada del vector es de 1 y ángulo es de  $135^\circ$  el pixel correspondiente se representa con el color azul oscuro y así sucesivamente. Con esta representación es posible visualizar el movimiento de todos los pixeles de una imagen de alta resolución de forma simultánea.



**Figura 2.11:** Convención de codificación de colores para la representación del flujo óptico denso (tomado de [13]).

En la figura 2.12 se presenta un ejemplo de la base de datos de Sintel [7], donde las primeras dos figuras son la secuencia de movimiento y la tercer figura es la matriz de salida/vectores de desplazamiento mapeado a la codificación de colores.

### 2.5.5. Métricas de evaluación

Una vez implementado el método de estimación de flujo óptico, es necesario poder validarlo. La métrica más utilizada para validar el flujo óptico es el *EndPoint Error* (EPE). Esta métrica consiste en calcular la magnitud en pixeles de la diferencia de la estimación y la etiqueta. El EPE para un pixel  $i, j$  dado se expresa como:

$$EPE_{ij} = \sqrt{(u_{ij} - u_{ijGT})^2 + (v_{ij} - v_{ijGT})^2} \quad (2.17)$$



**Figura 2.12:** Ejemplo de representación del flujo óptico de la base de datos Sintel (tomado de [7]).

donde  $(u_{ij}, v_{ij})$  es la posición estimada y  $(u_{ij_{GT}}, v_{ij_{GT}})$  la posición de referencia. Este cálculo se realiza para cada píxel de la imagen, resultando en una matriz de EPE de dimensiones  $W \times H$ . Para calcular una métrica que represente el comportamiento del EPE en toda la imagen se debe calcular el promedio de EPE. Esta métrica se conoce como *Average Endpoint Error* (AEE), calculado como

$$AEE = \frac{\sum_{i=0}^{W-1} \sum_{j=0}^{H-1} EPE_{ij}}{WH} \quad (2.18)$$

## 2.6. Segmentación Semántica

En el procesamiento de imágenes, el proceso de estimar si un píxel dado corresponde o no a un objeto dado es conocido como segmentación. Para un objeto dado el resultado es una máscara binaria; en esta un valor de 1 indica que ese píxel pertenece al objeto y un 0 indica que no pertenece a este.

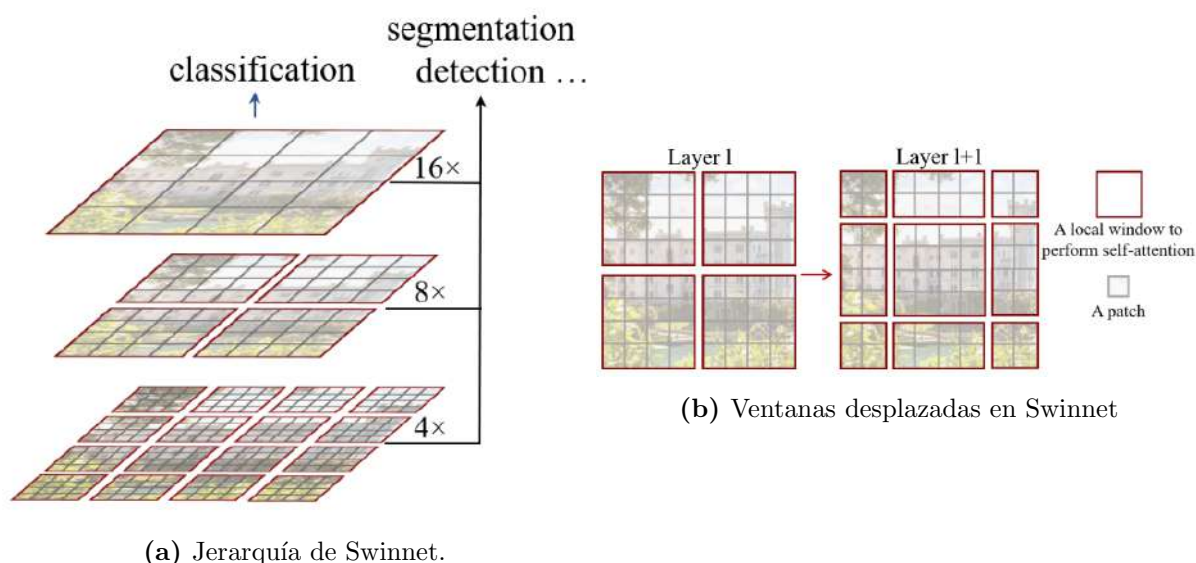
Es segmentación semántica cuando dos objetos del mismo tipo, a pesar de no ser el mismo objeto, por ejemplo 2 personas en la imagen, son marcadas con la misma etiqueta [25].

### 2.6.1. Backbones

Los segmentadores modernos se separan modularmente en dos secciones: un una columna vertebral (o *backbone*) que opera como un extractor de características y un clasificador por píxel o conjuntos de píxeles (ventanas que opera sobre las características extraídas en la primera sección).

### 2.6.2. Swinnet

Un transformador Swin [34] es un transformador de visión que sirve como columna vertebral de propósito general para la visión por computador. Propone un transformador [56] jerárquico cuya representación se calcula con ventanas desplazadas. El esquema de ventanas desplazadas aporta una mayor eficiencia al limitar el cómputo de la autoatención a las ventanas locales que no se traslapan, al tiempo que permite la conexión entre ventanas, mostrado en la figura 2.13b. Esta arquitectura jerárquica tiene la flexibilidad de modelar a varias escalas y tiene una complejidad computacional lineal con respecto al tamaño de la imagen, mostrado en la figura 2.13a.



**Figura 2.13:** Diagrama de capas de Swinnet (Tomado de [34]).

### 2.6.3. ConvNext

Los transformadores jerárquicos (por ejemplo, los transformadores Swin), hicieron que los transformadores fueran prácticamente viables como columna vertebral de la visión genérica y demostraron un rendimiento notable en una amplia variedad de tareas de visión [35]. Sin embargo, la eficacia de estos enfoques híbridos se atribuye en gran medida a la superioridad intrínseca de los transformadores, más que a los sesgos inductivos inherentes a las convoluciones. En [35] los autores reexaminan los espacios de diseño y ponen a prueba los límites de lo que puede lograr una red convolucional pura. Gradualmente “modernizan” una ResNet estándar hacia el diseño de un transformador de visión. Estos cambios incluyen:

- Cambios en la relación entre etapas.
- Estrategia de “parcheado” más agresiva: Convolución  $4 \times 4$  no traslapada.

- Convolución de profundidad.
- Cuello de botella invertido.
- Cambios en los tamaños de kernel.
- Usa una función de activación GELU en vez de RELU.
- Menos activaciones.
- Menos normalizaciones.
- Utiliza normalización lineal en lugar de normalización por lotes.
- Capas separadas de submuestreo.

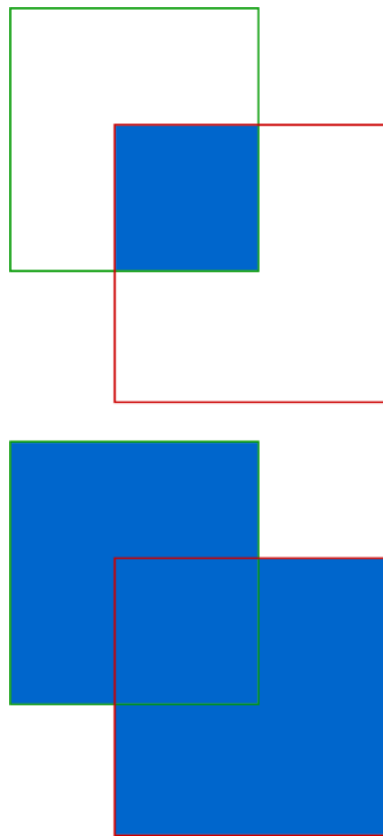
Todos estos cambios le permiten subir su rendimiento en ImageNet top-1 del 78.8 % en ResNet a un 82.0 % en la version final. Esto sobre el 81.3 % de SwinNet.

#### 2.6.4. Métrica de la intersección sobre la unión

Intersección sobre la unión *Intersection over Union* (IoU) es una medida basada en el índice de Jaccard[33] que evalúa la superposición entre dos áreas delimitadoras. Requiere un área delimitadora de verdad  $B_{gt}$  y un cuadro área predicha  $B_p$ . Al aplicar el IOU se puede determinar si una detección es válida (verdadero positivo) o no (falso positivo) por medio de un umbral. IOU está dado por el área de intersección entre el área delimitadora predicha y el área delimitadora de los datos reales, dividida por el área de unión entre ellas:

$$IOU = \frac{B_p \cap B_{gt}}{B_p \cup B_{gt}} \quad (2.19)$$

En la figura 2.14 se observan las dos áreas representadas, donde el IOU sería la razón entre estas áreas.



**Figura 2.14:** Intersección (arriba) y unión (abajo) de dos áreas



## Capítulo 3

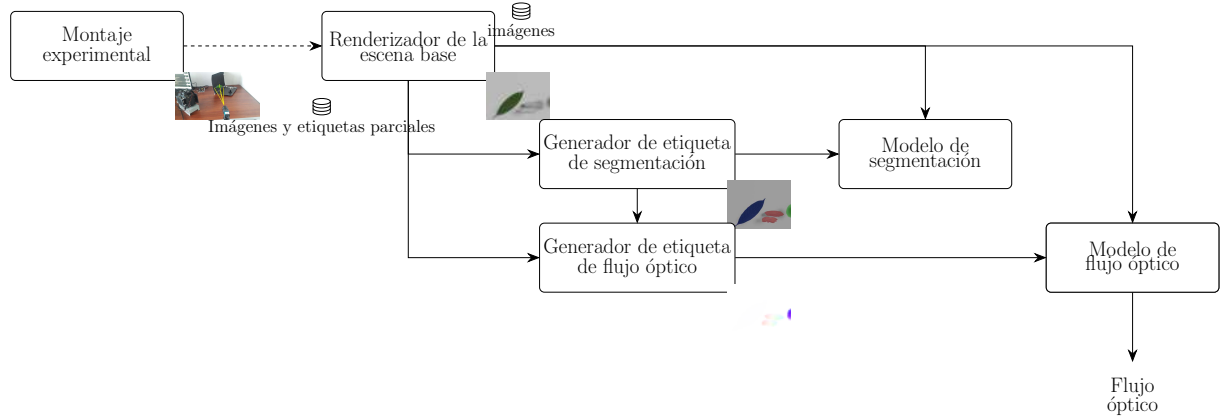
# Sistema de estimación de movimiento de esporas de la roya ante impacto de gotas de agua

En este capítulo se detalla la solución propuesta para el desarrollo del algoritmo de estimación del movimiento de esporas del hongo de la roya. La solución consiste en los siguientes elementos:

- Una configuración experimental utilizada como referencia para la generación posterior de datos.
- Un renderizador para la escena base, que generará las imágenes similares a la configuración experimental, que se utilizarán para el entrenamiento y la evaluación de los modelos. También generará algunos datos intermedios que se utilizarán para la generación de etiquetas de segmentación y de flujo óptico.
- Un generador de datos para las etiquetas de segmentación.
- Un generador de etiquetas de flujo óptico.
- Un modelo de segmentación utilizado para la inferencia de la etiqueta de los objetos.
- Un modelo de flujo óptico para la inferencia del movimiento de los objetos.

En la figura 3.1 se esboza un diagrama de bloques de la solución propuesta.

En las siguientes secciones se detalla el diseño de cada etapa y los detalles relevantes para su implementación.



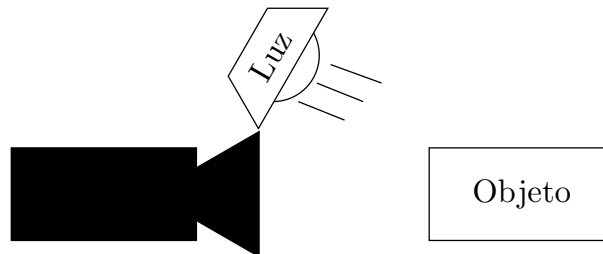
**Figura 3.1:** Diagrama general de la solución.

### 3.1. Montaje experimental

El fenómeno de la dispersión de esporas tiene varias características. Primero, el tamaño de las esporas es aproximadamente  $20\ \mu\text{m}$ . El principal efecto que esta característica tiene es que debido a su reducido tamaño, estas son difíciles de ser iluminadas y por lo tanto la captura de estas en una imagen es una tarea compleja. Lo segundo es que, debido a la velocidad a la que este fenómeno de dispersión ocurre, la captura a una tasa baja de 20 o 30 cuadros por segundo causa que las esporas se observen como trazas en lugar de puntos.

El sistema de captura propuesto utiliza una cámara de alta velocidad para capturar los recuadros del fenómeno, en conjunto con un sistema de iluminación con una alta luminancia, de manera que el fenómeno pueda ser observado claramente.

Un diagrama de esta solución se ilustra en la figura 3.2. El montaje consiste en tres objetos: la cámara de alta velocidad, el objeto bajo estudio y la iluminación. Nótese que la iluminación es una luz frontal, es decir en la dirección de captura de la cámara.



**Figura 3.2:** Diagrama del montaje experimental.

La cámara se coloca con la hoja centrada en su campo de visión, y con la hoja y la cámara colocadas a la misma altura en la mesa. La distancia entre la cámara y la hoja es de aproximadamente 50 cm. Se utiliza un fondo negro uniforme para proporcionar un mayor contraste del fenómeno. Esta configuración se muestra en la figura 3.3.

La ubicación de la iluminación está en un ángulo de aproximadamente 45 grados con



**Figura 3.3:** Ubicación de la cámara en el montaje experimental (Tomado de [41]).

respecto al eje óptico de la cámara, como se muestra en la figura 3.4. Casi paralelo a los rayos de iluminación se encuentra el fondo. La ligera diferencia con la normal permite iluminar la hoja y el evento, pero el fondo y sus imperfecciones no se iluminan.



**Figura 3.4:** Ubicación de la luz en el montaje experimental (Tomado de [41]).

Las gotas se dejan caer a una altura aproximada de 60 cm, dado que para alturas menores a esta las gotas no generan ningún tipo de salpiqueo.

Las características de la cámara utilizada se resumen en la tabla 3.1 mientras que las características del objetivo se encuentran en la tabla 3.2.

**Tabla 3.1:** Características de la cámara Imaging Source DFK 33UX252.

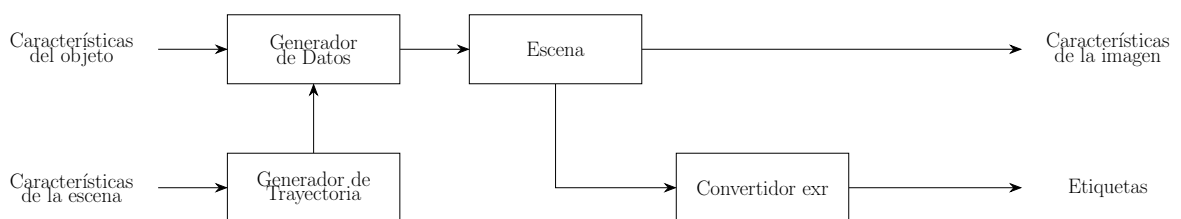
Característica	Valor
Tamaño del Chip	1/1.8"
Resolución Máxima	2048 × 1536@120fps
Resolución Mínima	640 × 480@608fps
Pixel Size	3.45 μm
Eficiencia Cuántica	63 % (530 nm)

**Tabla 3.2:** Características del objetivo Computar H3Z1014CS.

Característica	Valor
Mínima Distancia de Operación	60 cm
FOV horizontal	44.8°–15.6°
Distancia Focal	10 mm–30 mm
Apertura	16–1.4

## 3.2. Generación del conjunto de datos

El diseño del generador de datos sintéticos se muestra en la figura 3.5. Se utiliza *Blender* [10] para la generación de los datos sintéticos y de las correspondientes etiquetas. Para generar la escena correspondiente se toman como referencia videos del fenómeno correspondiente.



**Figura 3.5:** Diagrama general del generador de datos.

Para la escena se debe considerar el posicionamiento de la cámara y las características como: la distancia focal, el tamaño del sensor e incluso aspectos del objetivo como la apertura. Para estos valores se toman como referencia aquellos de la cámara utilizada para el montaje experimental.

La escena dentro de la animación es generada dinámicamente basada en dos factores. El primero lo constituyen las características de los objetos a ser simulados; por ejemplo la refracción del agua o el color de las esporas. La segunda entrada al sistema es cómo

se comportan los objetos, lo que implica la trayectoria de estos objetos para simular el movimiento deseado. La trayectoria de los objetos no se controla directamente, sino que se utiliza el sistema de simulación de física de *Blender*.

Cuando todos los objetos están integrados en la escena, la animación está lista para ser renderizada. Para generar las etiquetas a partir del renderizado, en *Blender* se debe utilizar el motor de renderizado “cycles” y habilitar pases de renderizado en las capas que requieren información de los vectores de movimiento. Luego se utiliza el editor de nodos para conectar las capas de renderizado a un bloque de salida, donde se guardará tanto la imagen, el vector y la máscara de segmentación. La imagen y la máscara de segmentación se guardan en formato **png** y el vector en formato **exr** el cual es un formato para imágenes de punto flotante [28] soportado por *Blender*. El tamaño de la imagen se establece en la configuración de renderizado y se define en  $640 \times 480$ .

Finalmente para convertir los archivos de salida de *Blender* a la etiqueta se requiere de un convertidor. Este convertidor tiene la funcionalidad de extraer la información como el tamaño de la imagen, leer los tres canales RGB de la imagen **exr** y utiliza únicamente los canales R y G para representar las componentes  $u$  y  $v$  del vector de movimiento. Estos datos se guardan de nuevo en un archivo binario con estos dos canales.

### 3.2.1. Generación de imágenes base y simulación

El escenario de la simulación incluye tres tipos diferentes de objeto:

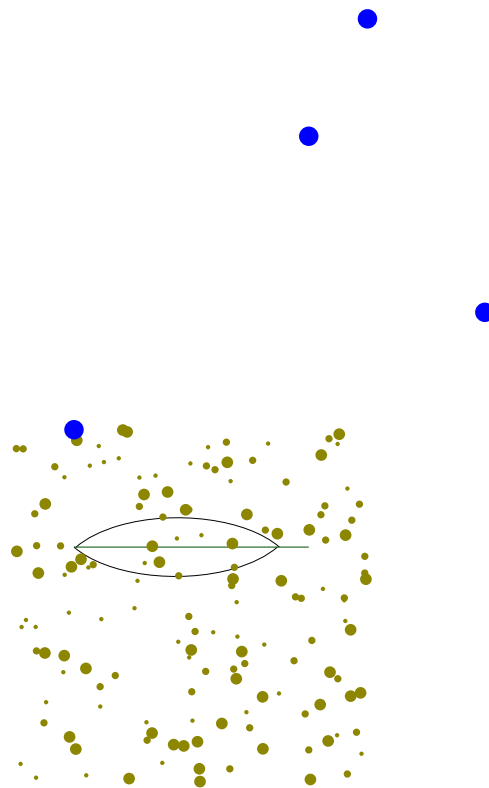
1. Agua: Este objeto está integrado por una cantidad variable de gotas entre las distintas escenas simuladas. El comportamiento de estas gotas de agua está gobernado por una simulación de líquido. Durante sus primeras etapas de caída libre este comportamiento tendrá una aceleración similar a la dada por la gravedad. Una vez que estas gotas impacten, ya sea con la base de la simulación o con las hojas, estas tendrán un efecto de salpicadura y se depositarán sobre el objeto donde hayan impactado, donde interactuará posteriormente con cualquier otra gota de agua que impacte. Su material está configurado de manera que tenga un efecto de transparencia. Además, debido a la geometría de las gotas de agua en conjunto con su efecto de reflexión interna, aquellos objetos que se encuentren detrás de la gota de agua se verán reflejados verticalmente.
2. Hojas: Este objeto está colocado fijo en la simulación e interactuará con las gotas de agua que caigan sobre él.
3. Esporas: Estas se comportan según una simulación de gas. Esta nube gaseosa es ligeramente traslúcida y flotará alrededor de la escena. Esta nube de esporas no se genera luego de un impacto de las gotas de agua con la hoja, sino que esta existe desde el inicio de la simulación.

- 3.1. Esporas puntuales: Además de las esporas generadas por la simulación de gas, se tienen esporas puntuales. Estas son representadas por objetos sólidos, configurados de manera que se visualicen de una manera reflectiva. El movimiento de estas será controlado de igual manera por la simulación de gas pero provee mayor variabilidad en la simulación.

El conjunto de datos está conformado por una serie de escenas independientes. Para incrementar la variabilidad del conjunto de datos, la cantidad de gotas de agua, la posición inicial de todos los objetos, incluyendo la cámara y la velocidad inicial de la nube de esporas varía entre las escenas utilizando un número aleatorio que varía un valor base.

### 3.2.2. Simulación inicial

El primer paso en la generación de la etiqueta es la simulación inicial. Esta simulación genera una renderización de la escena realista, además de las etiquetas de flujo óptico y de segmentación para toda el área en la imagen que no corresponda a la “nube” de esporas. Si la escena no contuviera objetos gaseosos, esta configuración inicial sería suficiente para obtener etiquetas de todos los objetos. Una representación de la escena en esta etapa se ilustra en la figura 3.6.



**Figura 3.6:** Escena base renderizada. Azul representa las gotas de agua y verde las esporas.

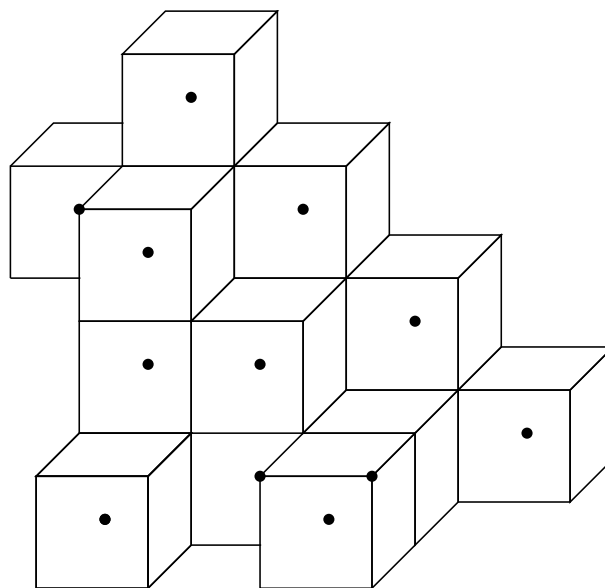
Las imágenes generadas son utilizadas directamente como insumo para el entrenamiento de tanto del modelo de flujo óptico como del modelo de segmentación, mientras que las

etiquetas serán integradas posteriormente con las etiquetas de segmentación. Durante esta etapa además se configura la simulación gaseosa para que utilice como formato de su caché (en este contexto el almacenamiento interno de la simulación) a `OpenVDB`. `OpenVDB` es una biblioteca C++ que incluye una estructura de datos jerárquica y un amplio conjunto de herramientas para el almacenamiento y la manipulación eficientes de datos volumétricos dispersos discretizados en cuadrículas tridimensionales [39].

Además se almacenan los datos de la matriz de cámara para la escena particular de manera que se utilice para la proyección de puntos tridimensionales sobre la imagen generada.

### 3.2.3. Generación de etiqueta de segmentación

Los datos de los vóxeles de la escena se reimportan hacia Blender utilizando un modificador de *Volume as Mesh* (una herramienta interna del programa). Esto genera un polígono alrededor del área de la “nube” de esporas. El límite exacto de cada polígono es determinado por un umbral de densidad. Al tener este objeto reimportado una malla asociada, el renderizador interno es capaz de generar etiquetas de profundidad y de segmentación. Esta máscara de segmentación se utilizará directamente como insumo del modelo de segmentación, además de que, en conjunto con la máscara de profundidad, se utiliza para poder generar la etiqueta de segmentación. En la figura 3.7 se muestra una representación de este proceso. La representación de `OpenVDB` tiene metadatos, como la velocidad, ubicados en una posición específica representado por el punto en la imagen.



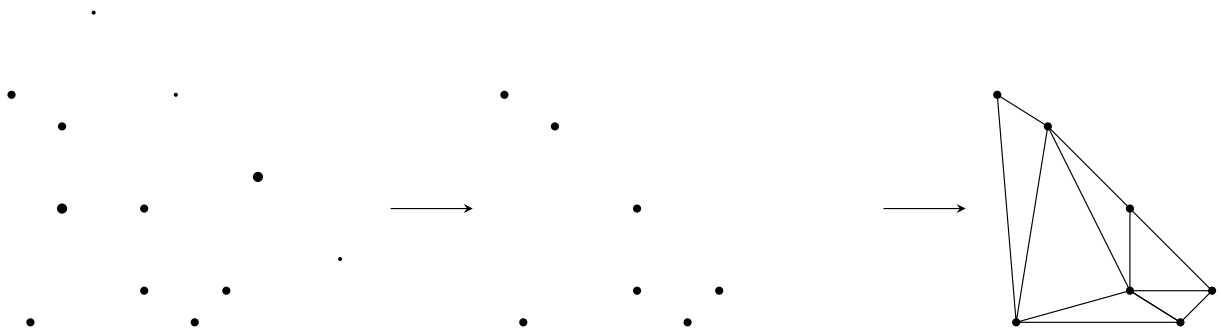
**Figura 3.7:** Vóxeles reimportados mostrando su area exterior y el punto donde los metadatos se ubican.

Actualmente debido a las limitaciones del programa Blender, la herramienta de reimportación de la información volumétrica no toma en cuenta los datos de velocidad incluidos en los vóxeles, por lo que una etiqueta de flujo óptico generado en esta etapa no tendría información para la “nube”.

### 3.2.4. Generación de la máscara de flujo óptico para el objeto gaseoso

Esta etapa toma como insumo los datos generados en las etapas anteriores para generar una sola etiqueta de flujo óptico. Los pasos necesarios para esto son:

1. Importar la información volumétrica de los vóxeles: posición, densidad y volumen.
2. Proyectar tanto la velocidad como la posición al eje de coordenadas 2D de la imagen. Este paso utiliza la ecuación (2.7), de manera que no solo se tenga la información de la proyección sino también la distancia en píxeles. El resultado de esto es la primera imagen en la figura 3.8.



**Figura 3.8:** Proceso de filtrado 2D de la nube y conexión de los nodos.

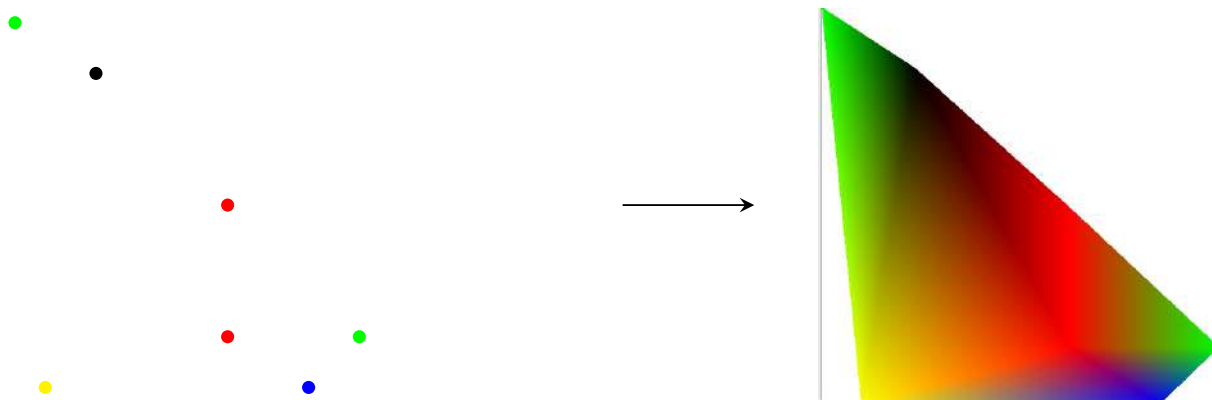
3. Para cada uno de los vóxeles proyectados, se filtra según dos factores:
  - La densidad tiene que ser mayor a un cierto umbral, que coincide con el utilizado en la sección 3.2.3.
  - La distancia obtenida en el paso anterior tiene que no diferir más de cierto umbral de la distancia dada en la etiqueta de profundidad.

Estos requisitos son utilizados de manera que la información en la etiqueta a ser generada corresponda a la superficie de la “nube” de cara a la cámara, y a su vez que el movimiento sea el representado por esta y no por las partículas de menor densidad frente a esta que no son visualmente representativas del movimiento o a las de mayor densidad que son ofuscadas por el movimiento de partículas delante de ellas. El resultado de este paso se representa en la imagen intermedia de la figura 3.8.

4. Para cada vóxel se tiene un valor puntual de posición, a pesar de que el vóxel proyectado cubre un área mayor a un solo píxel.
  - Estos píxeles se conectan en una sola red utilizando el algoritmo de Delauney [32]. El resultado de esto se presenta en la imagen final de la figura 3.8.
  - Utilizando esta red, conformada por una serie de nodos y sus vecinos, y los valores de velocidad para cada nodo, se interpolan los valores para cada uno



de los píxeles intermedios utilizando un algoritmo gráficos por computadora implementado a través de OpenGL [45]. El resultado de esto se muestra en la figura 3.9. En esta los puntos con colores representan aquellas posiciones conocidas donde se tienen valores de velocidad, representado por los distintos colores; en esta imagen se utilizan los tres canales RGB, pero dentro de OpenGL solo se utilizan los primeros dos para las dos componentes de los vectores de movimiento. OpenGL interpola los valores en el área comprendida entre los puntos.



**Figura 3.9:** Pseudo-renderización para obtener los valores interpolados.

5. La matriz resultante de la interpolación se filtra con la etiqueta de segmentación para solo mantener los píxeles que efectivamente se muestran en la cámara. Hasta ahora, incluso si la “nube” está ocultada por un objeto delante de ella, se ha seguido trabajando con esos puntos. De igual manera la etiqueta de flujo óptico generada por *Blender* se filtra con todos aquellos píxeles en la etiqueta que no sean parte de la “nube”.
6. Finalmente se combinan ambas matrices de etiquetas para obtener la etiqueta final. Esta se guarda en un formato estándar para su uso posterior en el entrenamiento del modelo de flujo óptico.

### 3.3. Selección de la red para la detección del flujo óptico

El resultado del flujo óptico es un mapeo del movimiento de píxeles entre cuadros consecutivos. Si se utiliza una red entrenada en base a un set de datos de referencia, los resultados para un set de datos distinto no serán óptimos pero seguirán siendo válidos como valores de referencia.

En el caso de las esporas, se toman 1200 pares de imágenes y para las gotas de agua se toman 93 pares de imágenes. Para el caso de las gotas de agua se evalúa AEE en

LiteFlowNet, SPyNet, PWC-Net y RAFT; mientras que para las esporas se evalúa esta misma métrica en SPyNet, HD3 y RAFT. Para la red conjunta solo se utiliza las redes de GMA.

La mayoría de estas redes tienen disponibles múltiples modelos pre-entrenados con sets de datos de referencia, los cuales van a afectar el desempeño final, por lo que no solo se evalúan las redes sino que para cada una de estas se evalúan los modelos preentrenados para conjuntos de entrenamiento públicamente disponibles. Esta evaluación permite tomar como punto de partida no solo la mejor red sino también el mejor conjunto de entrenamiento usado en el preentrenamiento.

La métrica que todas estas redes utilizan es el AEE; sin embargo, dado que la naturaleza del problema lleva a que un porcentaje del área de imagen de gotas esporas pixeles no tengan movimiento, el utilizar todas el área lleva a un promedio mucho más bajo que distorsiona el valor para los pixeles en movimiento de interés. Por lo tanto se propone una nueva métrica denominada *Filtered Average Endpoint Error* (FAEE).

### 3.4. Métrica FAEE

Para realizar el cálculo del FAEE es necesario comparar cada pixel de la predicción con el pixel correspondiente de la etiqueta, con el fin de determinar si en ese pixel existe movimiento que no sea despreciable, es decir, si las componentes  $u$  y  $v$  tanto en la predicción como en la etiqueta son mayores a un umbral  $\epsilon$ . En caso de que las componentes sean menor al umbral definido se debe excluir del cálculo y del promedio del error de la imagen en cuestión.

De esta matriz resultante se extrae una comparación entre las componentes de la predicción y la etiqueta ya que si el valor de la suma es menor a  $\epsilon$  indica que tanto las componentes de la predicción como las componentes de la etiqueta poseen una magnitud despreciable, es decir, se excluyen estos pixeles en el cálculo del error.

Luego, se toma la matriz de EPE para una predicción y se suman. Finalmente se cuentan todos los pixeles que no poseen movimiento despreciable  $F_{filt} > \epsilon$  y se divide el EPE entre la cantidad de elementos filtrados. El FAEE se expresa como

$$FAEE_{ij} = \frac{\sum_{i=0}^{W-1} \sum_{j=0}^{H-1} E_{ij}}{N} \quad (3.1)$$

donde  $N$  es la cantidad de elementos filtrados.

Esto es una aproximación al método exacto, pero con la ventaja de poder calcular el error utilizando tensores del GPU, lo cual reduce el tiempo de ejecución del proceso de entrenamiento y evaluación.

### 3.5. Selección de la red para la segmentación de los objetos

Las redes de detección del flujo óptico denso son capaces de estimar el movimiento para cada uno de los píxeles en la imagen. Sin embargo dado que la imagen está compuesta por más de un objeto, si se deseara, por ejemplo, estimar el movimiento promedio de las esporas, la red no es capaz de discriminar este del movimiento de las gotas de agua.

Para solucionar esto se entrenó una segunda red, en este caso para poder distinguir entre los objetos de la imagen. Para seleccionar estas se toman en cuenta las mejores redes en el estado del arte actual, según [11] y se realiza un entrenamiento contra el conjunto de datos actual.

### 3.6. Entrenamiento y evaluación de las redes neuronales

Esta etapa consiste en el entrenamiento y evaluación de modelos de estimación del flujo óptico y segmentación semántica de los objetos.

#### 3.6.1. Red para la estimación del flujo óptico

Se utiliza la técnica de aprendizaje transferido sobre el modelo con menor FAEE de la arquitectura elegida.

Inicialmente se entrenan dos redes separadas, una para la estimación del movimiento de las gotas de agua y una segunda red para la estimación del flujo óptico con los sets de datos iniciales. Finalmente se entrena una red “conjunta” que estima el movimiento tanto de las esporas como de las gotas de agua.

Todas las arquitecturas utilizan una función de costo diferente, sin embargo todas optimizan la diferencia entre la etiqueta del flujo óptico y la predicción. Para enfatizar que se requieren optimizar principalmente los píxeles que poseen movimiento, la función de costo debe ser modificada. Esta modificación permite que no solo se optimice el AEE, sino que también se considere el FAEE. Para ello se diseña una función de costo dada por la siguiente expresión:

$$L = loss_{AEE} + \lambda loss_{FAEE} \quad (3.2)$$

donde  $\lambda$  es un parámetro que regula el peso que tiene la función de costo calculada utilizando el FAEE en la función de costo total.

El entrenamiento y validación del modelo se realizan utilizando el conjunto de datos

sintéticos generado. Los resultados permiten determinar si la arquitectura y modelo pueden mejorar la estimación respecto al modelo base utilizando el conjunto de datos sintéticos generado.

El proceso de evaluación consiste en variar hiperparámetros como la tasa de aprendizaje, tamaño de lote, número de iteraciones, entre otros. Además, se ajustan los métodos de regularización como *dropout* y *weight decay* con el fin de caracterizar los efectos en el modelo que se está entrenando por medio de la optimización de la función de costo. Los datos se dividen en un conjunto de entrenamiento, uno de validación y uno de prueba.

Para mitigar los efectos del sobreajuste, se implementa un aumento de datos sobre el conjunto de datos de entrenamiento.

### **3.6.2. Red para estimación de la segmentación semántica**

Para esta red de igual manera se utiliza la técnica de aprendizaje transferido sobre los modelos con el mejor rendimiento en el estado del arte, SwinNet y ConvNext. Para esta red no se realizó una etapa previa de evaluación sobre el conjunto de datos generado, sino que se realizaron entrenamientos directamente sobre el conjunto de datos.

El entrenamiento y la evaluación se realizan sobre el conjunto de datos sintéticos generados. Se utiliza la métrica de IoU para evaluar qué tan capaces son las redes entrenadas de estimar correctamente qué objeto se está observando.

# Capítulo 4

## Resultados

Como se mencionó en la introducción, algunos de los resultados mostrados aquí fueron estudiados a través de un modelo de co-asesoramiento y, por lo tanto, presentados por primera vez en los trabajos citados de Hronek [22], Solano [50], Garcia [16] y Quesada [41].

### 4.1. Generación de datos

Como se menciona en el capítulo anterior, la generación de datos incluye tanto la generación de un conjunto de datos reales de referencia y para resultados cualitativos, y un conjunto de datos sintéticos para el entrenamiento y validación cuantitativo. Esta sección presenta los resultados de esta generación.

#### 4.1.1. Datos reales

En la figura 4.1 se muestra una secuencia de recuadros de una captura de esporas reales. Las esporas fueron sopladas para simular el movimiento de estas en el evento real. Se pueden notar esporas individuales en movimiento, similar a lo simulado con el generador de datos de esporas discretas. Sin embargo la mayor parte en esta imagen está conformada por una “nube” de esporas que no es considerada en la versión discreta de los datos simulados.



**Figura 4.1:** Secuencia de recuadros de esporas en movimiento capturadas a 600 fps.

### 4.1.2. Generación de datos sintéticos

Esta subsección muestra las imágenes renderizadas y las etiquetas correspondientes tanto para las esporas [22] como para las gotas de agua [50].

#### Esporas discretas

Para las esporas, el conjunto de datos generado consiste de 15 videos de 1000 cuadros cada uno, para un total de 15 000, imágenes con sus etiquetas respectivas de flujo óptico. Estas imágenes fueron renderizadas con tamaño de  $1280 \times 720$  px en un formato RGB. Por cada video renderizado se variaron los siguientes parámetros: cantidad de esporas, tamaño de la espora, tipo de atractor, iluminación y tipo de material. Además, se variaron las condiciones iniciales de los atractores de manera que se garantice variabilidad en el conjunto de datos. En la figura 4.2 se muestran dos ejemplos del conjunto de datos de entrenamiento donde se tienen los dos cuadros consecutivos y la etiqueta de flujo óptico correspondiente.

Los tamaños de las figuras geométricas que conforman el objeto que simula la espora, son generados con una distribución gaussiana con la media en el tamaño de la espora indicado en el parámetro de entrada y una varianza de un tercio del tamaño de la espora.

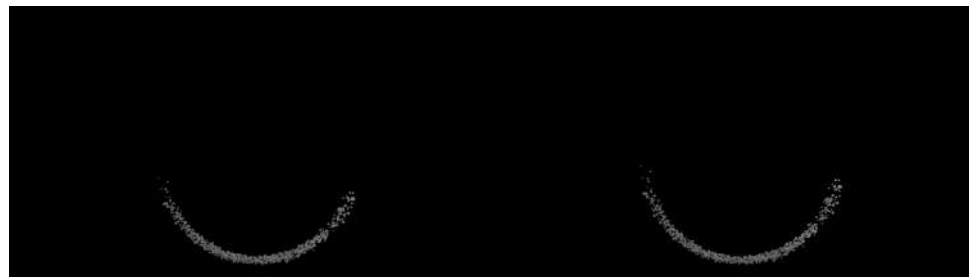
Se utilizan dos materiales distintos para las esporas: un material metálico y un material “espora”. El material metálico tiene una mayor reflectividad, una superficie lisa (sin difusión) y el color es plateado. El material “espora” posee una superficie más áspera lo cual genera diferentes sombras en la misma cara expuesta a una misma fuente de luz, su color es anaranjado pálido como es mostrado en la figura 4.2b.

#### Gotas de Agua

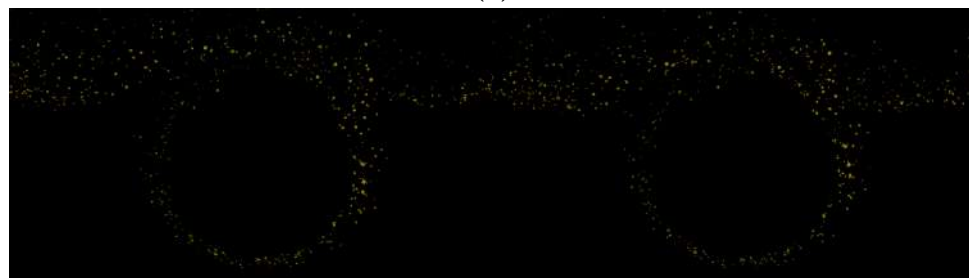
En el caso de las gotas de agua se generaron 4000 imágenes con sus correspondientes etiquetas de flujo óptico. En este caso el tamaño de las imágenes es de  $1024 \times 436$  px para el entrenamiento y  $480 \times 360$  px para las pruebas de verificación de las redes. Estas 4000 imágenes se separan en 4 escenas con diferente configuración en lo que corresponde a velocidad inicial, masa, tamaño, cantidad de gotas de agua y el fondo utilizado para las gotas. Las partículas son “emitidas” en una serie de planos, los cuales se encuentran perpendiculares al plano de la cámara y al fondo.

Las gotas de agua de la animación están hechas con un material que permite la refracción. Se configuró mediante la opción de *refraction BSDF*. Además, se determinó empíricamente una aspereza de 0.83 y un índice de refracción de 1.45 para tener una visualización de refracción en las gotas. La animación se renderiza con una tasa de 100 fps en *Blender*.

En las figuras 4.3a y 4.3b se presenta una muestra de las gotas de agua generadas en las animaciones y sus etiquetas respectivas. Las etiquetas presentan el flujo óptico del recuadro en el momento  $I_n$  al  $I_{n+1}$  y se almacenan en  $Gt_n$ .

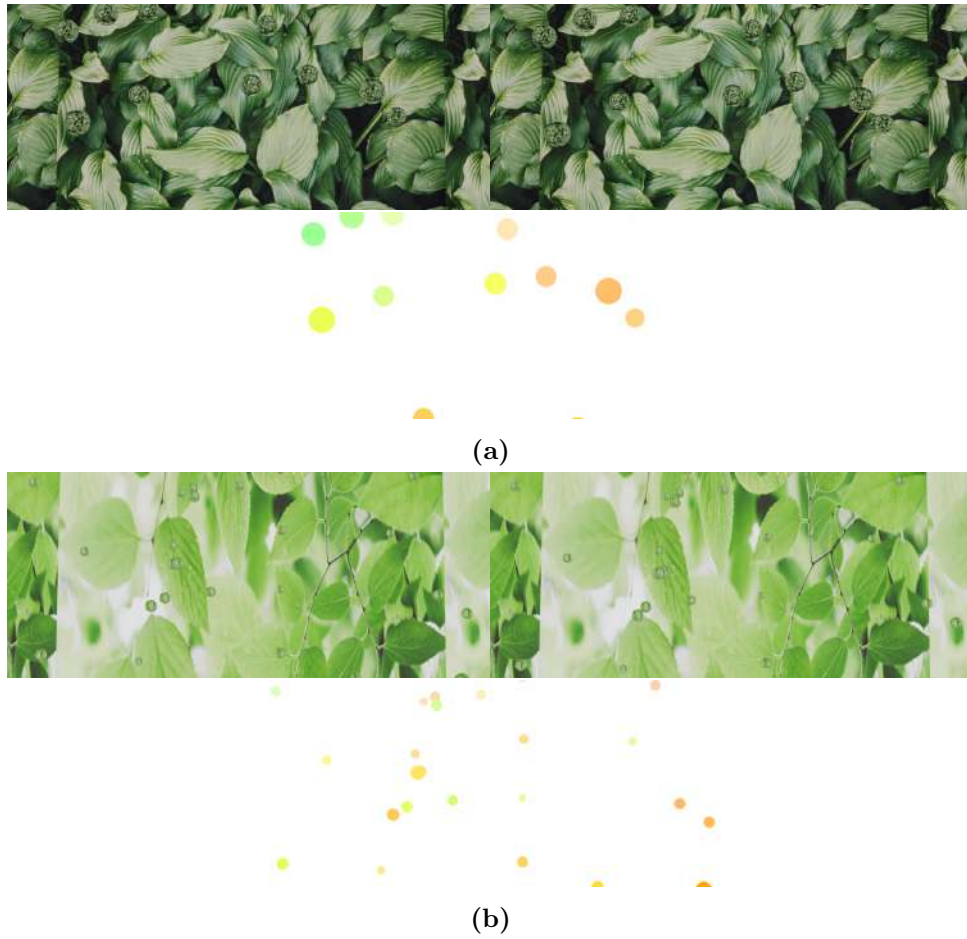


(a)



(b)

**Figura 4.2:** Muestras del conjunto de datos de entrenamiento con sus etiquetas respectivas.  
(Tomado de [22])



**Figura 4.3:** Muestras del conjunto de datos de entrenamiento con sus etiquetas respectivas. (Tomado de [50]).

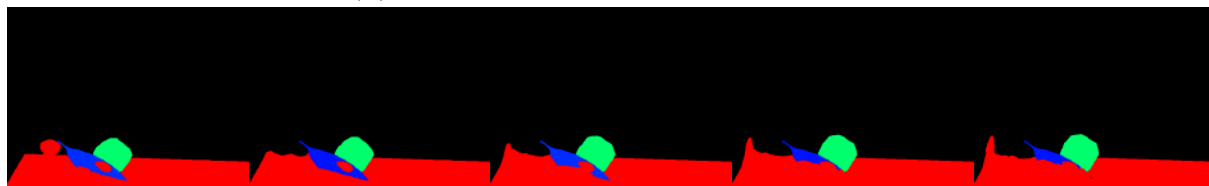




(a) Imágenes renderizadas.



(b) Etiqueta de flujo óptico correspondiente.



(c) Etiqueta de segmentación correspondiente.

**Figura 4.4:** Muestras del conjunto de datos de entrenamiento con sus etiquetas. Rojo es agua, azul hoja, verde esporas y negro fondo.

### Gotas de Agua y Esporas

El conjunto de datos integrado consta de 10 escenas, cada una de ellas con 250 fotogramas consecutivos, para 2500 imágenes. Estas imágenes se renderizaron con una resolución de  $640 \times 480$  para que coincidieran con las imágenes producidas por la cámara real. En cada una de las escenas se varía el número de gotas de agua y su posición y velocidad iniciales, la posición y la velocidad inicial de la nube de esporas, la posición de la cámara y la posición del objeto hoja. El fondo varía entre un fondo completamente negro y un fondo completamente blanco. La variación de todos estos parámetros se realiza en torno a un valor por defecto, cambiando según un valor aleatorio.

La escena también incluye la configuración para la refracción de las gotas de agua, y la interacción de los objetos físicamente con la escena, excepto la hoja, que no se mueve.

## 4.2. Verificación de las arquitecturas de aprendizaje automático

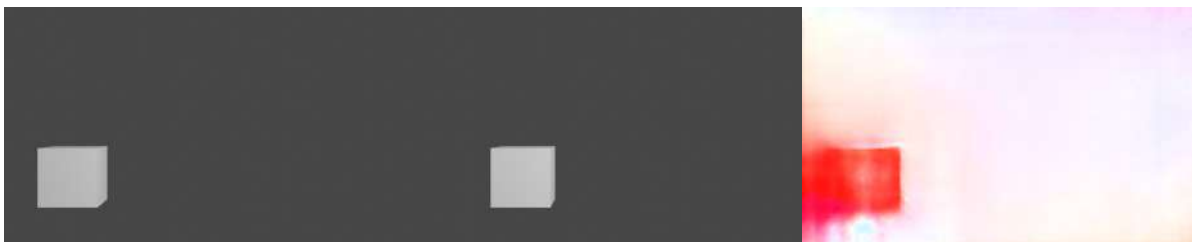
En esta sección se evalúan cuatro arquitecturas de aprendizaje automático con sus modelos preentrenados con los conjuntos de datos *Sintel*, *Flying chairs*, *Falling Things* y *Karlsruhe Institute of Technology and Toyota Technological Institute dataset* (KITTI). El principal objetivo de esta verificación es determinar cuál modelo tiene un mayor potencial para realizar una predicción correcta con las esporas. Entre las pruebas realizadas para verificar

el funcionamiento de las arquitecturas se encuentran pruebas con el conjunto de datos sintético de gotas [50].

Se observa que cada red sigue convenciones particulares de almacenamiento del vector de flujo: puede almacenar respecto a la imagen en el tiempo  $n$  ó  $n + 1$  y en dirección hacia adelante o hacia atrás, es decir almacenado el flujo de  $n \rightarrow n + 1$  o  $n - 1 \rightarrow n$ . Por esta razón se realiza un experimento cualitativo para determinar visualmente las características de las arquitecturas para asegurar que coincidan con las etiquetas.

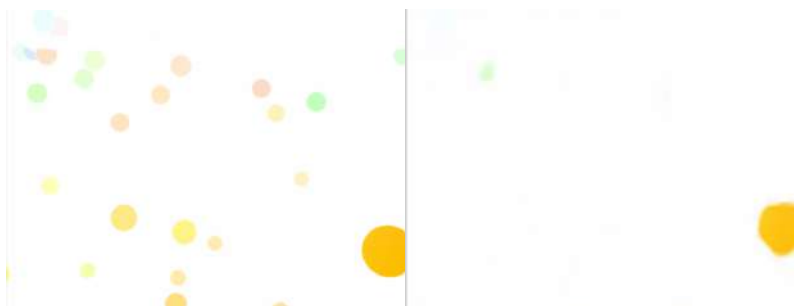
#### 4.2.1. PWC-NET

En la figura 4.5 se observan las imágenes al desplazar un cubo en el eje  $x$ , esto con el fin de determinar de manera empírica el tipo de flujo óptico que produce la red. La primera imagen presenta el cubo en el momento  $I_n$ , la segunda en el momento  $I_{n+1}$  y la tercera la predicción de PWC-Net. Se observa que esta red realiza el cálculo de flujo óptico hacia adelante y almacena la etiqueta en el momento  $n$ , al igual que el set de datos de gotas sintético.



**Figura 4.5:** Predicción PWC con cubo en eje  $x$ . (Tomado de [50])

La figura 4.6 muestra el flujo óptico de la etiqueta y la predicción, de izquierda a derecha respectivamente. La arquitectura no identifica las gotas más pequeñas.



**Figura 4.6:** Evaluación PWC con conjunto de datos *Flying Chairs*. A la izquierda se muestra la etiqueta y a la derecha la predicción. (Tomado de [50])

#### 4.2.2. SPyNet

En la figura 4.7 se ilustran las imágenes al desplazar un cubo en el eje  $y$ . La primera imagen presenta el cubo en el momento  $I_n$ , la segunda en el momento  $I_{n+1}$  y la tercera

la predicción de la red. Se observa que realiza el cálculo de flujo óptico hacia adelante y almacena la etiqueta en el momento  $n$ .



**Figura 4.7:** Predicción SPyNet con cubo en eje  $y$ . (Tomado de [50])

La figura 4.8 muestra el flujo óptico para KITTI, *Flying Chairs* y *Sintel*, de izquierda a derecha respectivamente. Tomando en cuenta que el color amarillo representa movimientos hacia abajo y rojo hacia arriba, se nota en esta imagen como la red realiza una inversión del flujo óptico. Esto es probablemente causado por el efecto de refracción que se da en las gotas de agua, el cual invierte el movimiento por un efecto lente. Se observa que esta arquitectura reconoce más objetos que PWC-Net.



**Figura 4.8:** Evaluación arquitectura SPyNet. A la izquierda entrenado con KITTI, al centro con *Flying Chairs* y a la derecha con *Sintel*. (Tomado de [50])

### 4.2.3. LiteFlow Net

En la figura 4.9 se observan las imágenes al desplazar un cubo en el eje  $x$ . La primera imagen presenta el cubo en el momento  $I_n$ , la segunda en el momento  $I_{n+1}$  y la tercera la predicción de la red. Se observa que también esta red realiza el cálculo del flujo óptico hacia adelante y almacena la etiqueta en el momento  $n$ .

La arquitectura LiteFlow Net fue evaluada con los pesos de los conjuntos de datos KITTI, *Flying Chairs*, *Sintel*. La figura 4.10 muestra el flujo óptico para estos conjuntos, de izquierda a derecha respectivamente. La arquitectura realiza una inversión del flujo óptico usando los pesos del conjunto KITTI. En el caso de *Flying Chairs* y *Sintel* se observa que reconoce más objetos que PWC-Net, pero menos que SPyNet.



**Figura 4.9:** Predicción LiteFlow Net con cubo en eje x. (Tomado de [50])



**Figura 4.10:** LiteFlow Net. A la izquierda entrenado con KITTI, al centro con *Flying Chairs* y a la derecha con *Sintel*. (Tomado de [50])

#### 4.2.4. RAFT

En la figura 4.11 se observan las imágenes al desplazar un cubo en el eje  $y$ . RAFT hace las predicciones de la imagen  $I_n$  a  $I_{n+1}$  y guardando las etiquetas en el momento  $n$ .

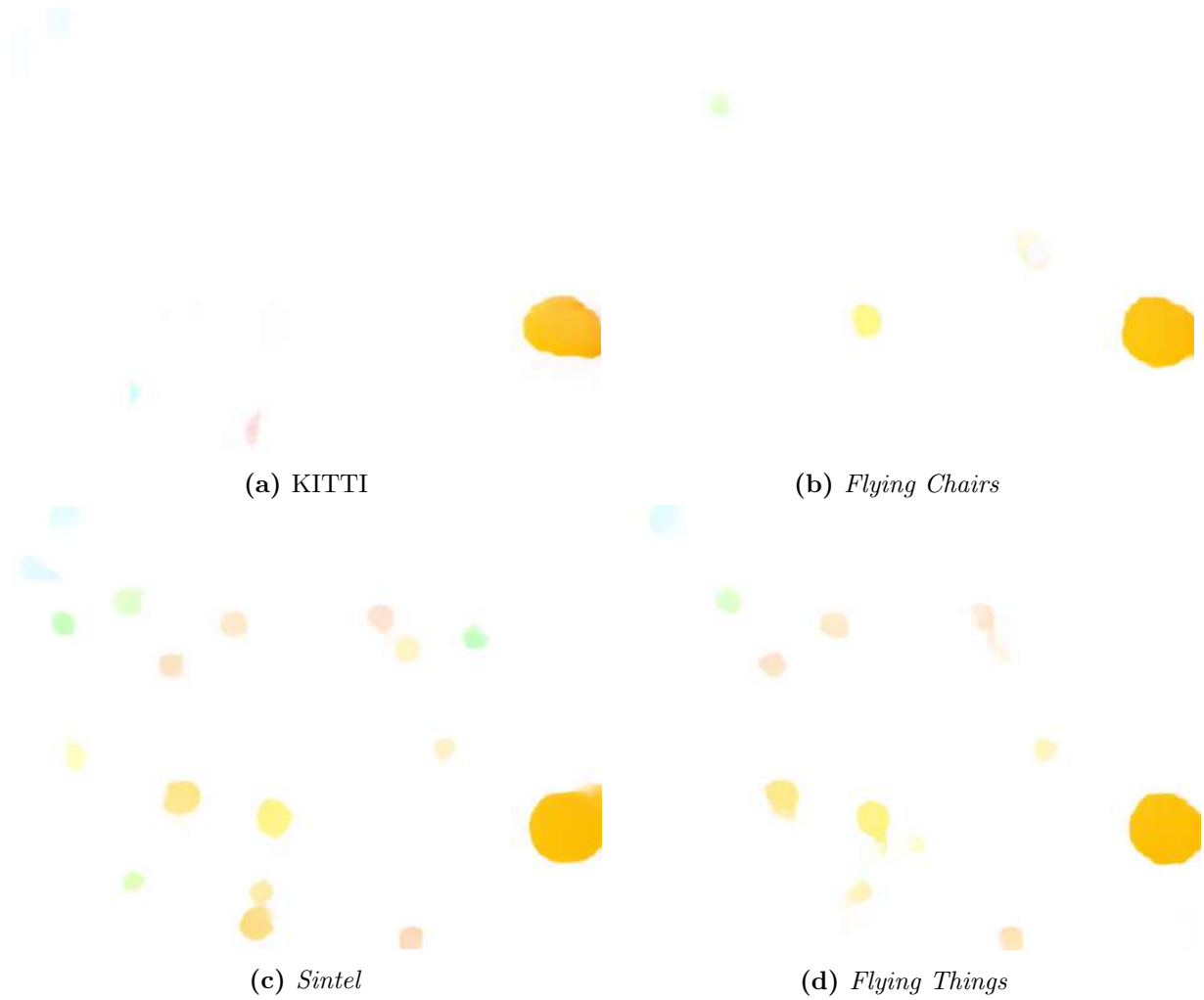


**Figura 4.11:** Predicciones RAFT en el eje  $y$ . (Tomado de [50])

La figura 4.12 muestra el flujo óptico para los distintos modelos. La arquitectura RAFT elimina varias gotas usando los pesos del conjunto KITTI y *Flying Chairs*. En el caso de *Sintel* y *Flying Things* se observa que reconoce más objetos que PWC-Net.

### 4.3. Evaluaciones preliminares

En esta sección se presentan los resultados de las evaluaciones preliminares para elegir la arquitectura y los pesos que se utilizarán en el proceso de entrenamiento. Esta evaluación se realiza tanto para las esporas [22] como para las gotas [50].



**Figura 4.12:** Predicciones de la red RAFT para distintos conjuntos de datos. (Tomado de [50])

### 4.3.1. Esporas

Para la evaluación de las redes para el caso de las esporas se utilizan 1200 imágenes de tres videos del conjunto de datos generado. Se realizan cuatro evaluaciones por arquitectura (exceptuando SPyNet que no posee un modelo pre-entrenado con el conjunto de datos KITTI).

Los resultados del AEE de las evaluaciones se resumen en la tabla 4.1. El modelo con menor AEE de RAFT es *Chairs* ( $AEE = 0.464$  px), de HD3 es KITTI ( $AEE = 29.62$  px) y de SPyNet es *Chairs* ( $AEE = 5.516$  px). Si se comparan los tres modelos con menor AEE de cada arquitectura, RAFT posee el menor error de los tres. Por lo tanto, la arquitectura RAFT será utilizada con el modelo *Chairs* como punto de partida para el proceso de entrenamiento del nuevo modelo.

**Tabla 4.1:** Evaluación preliminar por arquitectura para el set de datos de esporas utilizando modelos pre-entrenados. (Tomado de [22])

Modelo	AEE [pixel]		
	RAFT	HD3	SPyNet
<b>Chairs</b>	<b>0.464</b>	37.88	32.35
<b>Things</b>	0.599	58.14	-
<b>KITTI</b>	6.600	29.62	35.868
<b>Sintel</b>	0.847	32.35	11.536

Para cuantificar el rendimiento de la estimación de flujo óptico en únicamente los pixeles de interés se utiliza la métrica desarrollada FAEE. Los resultados del FAEE para todos los modelos disponibles en la arquitectura RAFT se resumen en la tabla 4.2.

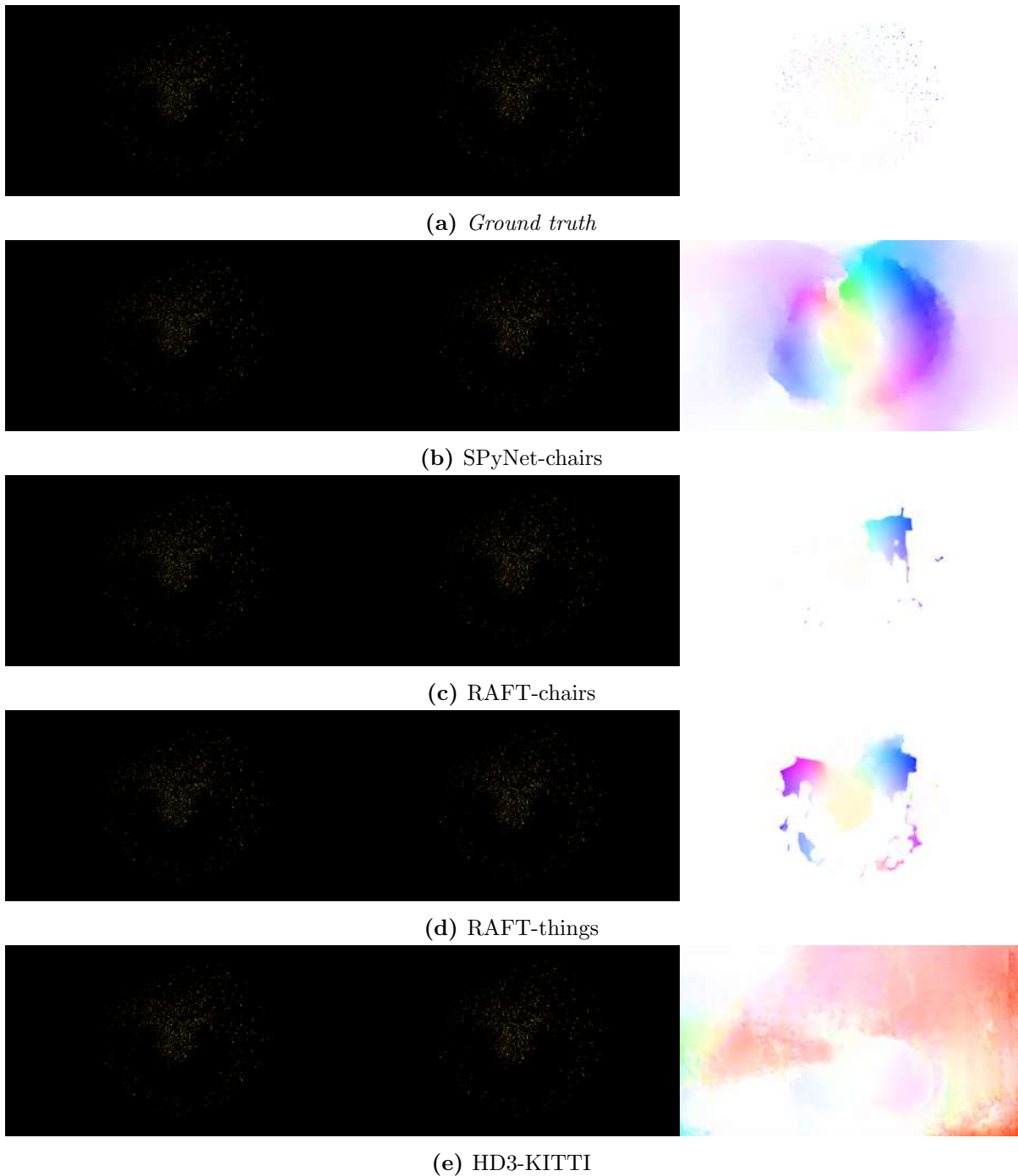
A partir de la tabla 4.1 se determina que el modelo *Chairs* utilizando la arquitectura RAFT posee un AEE 63.75 veces menor que el mejor modelo evaluado de HD3 (KITTI) y 11.87 veces menor que SPyNet (*Chairs*).

**Tabla 4.2:** FAEE de la arquitectura elegida (RAFT) evaluado en el conjunto de esporas. (Tomado de [22])

Modelo	FAEE [pixel]
<b>Chairs</b>	3.901
<b>Things</b>	<b>2.246</b>
<b>KITTI</b>	12.365
<b>Sintel</b>	5.192

Según los resultados de la tabla 4.2 los modelos con menor FAEE son *Chairs* y *Things*, sin embargo, el modelo *Chairs* tiene un FAEE 1.74 mayor respecto al modelo *Things*. Según las figuras 4.13c y 4.13d, se observa que el modelo de *Chairs* posee un AEE menor ya que la estimación es cero en algunas regiones donde sí existe movimiento. Esto se ve

reflejado en la métrica de FAEE, donde se enfatizan específicamente los píxeles donde existe movimiento. El modelo de *Things* sí predice en la mayoría de las regiones por lo que el AEE es mayor pero el FAEE es menor.



**Figura 4.13:** Estimaciones preliminares de los modelos preentrenados con mejores métricas. (Tomado de [22]).

### 4.3.2. Gotas de Agua

El caso de las gotas de agua tiene la ventaja de que son objetos más grandes que las esporas, lo cual es más representativo de los conjuntos de datos existentes. Al implementar las cuatro arquitecturas, se realizaron pruebas que evalúan el AEE y el FAEE con un umbral de 0.5 píxeles.

**Tabla 4.3:** Evaluación preliminar por arquitectura para el set de datos de gotas utilizando modelos pre-entrenados. (Tomado de [50])

Modelo	AEE [pixel]			
	RAFT	HD3	SPyNet	PWC-Net
<b>Chairs</b>	<b>0.396</b>	0.922	0.995	0.700
<b>KITTI</b>	0.878	0.974	0.984	-
<b>Sintel</b>	0.866	0.870	0.894	-

**Tabla 4.4:** Evaluación preliminar por arquitectura para el set de datos de gotas utilizando modelos pre-entrenados con el FAEE. (Tomado de [50])

Modelo	FAEE [pixel]			
	RAFT	LiteFlowNet	SPyNet	PWC-Net
<b>Chairs</b>	<b>0.880</b>	2.931	2.131	0.973
<b>KITTI</b>	3.784	2.145	3.978	-
<b>Sintel</b>	3.660	2.818	1.276	-

En las tablas 4.3 y 4.4 se observan los resultados al evaluar las arquitecturas con 93 pares de imágenes. Los valores más bajos de error los presenta la red RAFT preentrenada con el conjunto de datos *Flying Things* con un AEE de 0.396 y un FAEE de 0.880. Con base en esto se selecciona la red RAFT y los pesos de *Flying Chairs* para realizar el entrenamiento con el conjunto de datos sintético de gotas de agua.

## 4.4. Re-entrenamiento y validación del modelo

Esta sección presenta los resultados de la validación de los modelos entrenados para las esporas [22] y para las gotas de agua [50] por separado, partiendo de los modelos seleccionados en la sección anterior.

### 4.4.1. Esporas

Para las esporas se considera el reentrenamiento y la evaluación de los modelos de *Chairs* y *Things* utilizando la arquitectura de RAFT. Para todos los entrenamientos se utiliza un valor de  $\gamma = 0,8$  para la función de costo dada por la siguiente expresión:



$$\mathcal{L} = \sum_{i=1}^N \gamma^{N-i} \|f_{gt} - f_i\|_1 \quad (4.1)$$

El factor  $\gamma$  reduce el impacto de las primeras predicciones del lote de tamaño  $N$  en el cálculo de la función de costo, dando mayor peso a las últimas predicciones refinadas. Esto de manera que el impacto de la diferencia entre el modelo preentrenado y el modelo final no sea drástico.

Para integrar el FAEE a la función de costo con el objetivo de poder reducir no solo el AEE, sino que también el FAEE, la función de costo agregando el factor del FAEE se expresa como:

$$\mathcal{L} = \sum_{i=1}^N \gamma^{N-i} \|f_{gt} - f_i\|_1 + \lambda \sum_{i=1}^N \gamma^{N-i} \|f'_{gt} - f'_i\|_1 \quad (4.2)$$

donde el primer término consiste en la pérdida del AEE que ya era considerada, y el segundo término representa el FAEE. Los factores  $f'_{gt}$  y  $f'_i$  corresponden a la etiqueta y la estimación filtrada respectivamente y  $\lambda$  es un factor que permite ponderar el efecto del FAEE en la optimización.

El FAEE y el AEE se evalúan contra un conjunto de 1200 imágenes. Además se utiliza como referencia el método clásico de Lucas-Kanade. Estas métricas resultantes se resumen en la tabla 4.5.

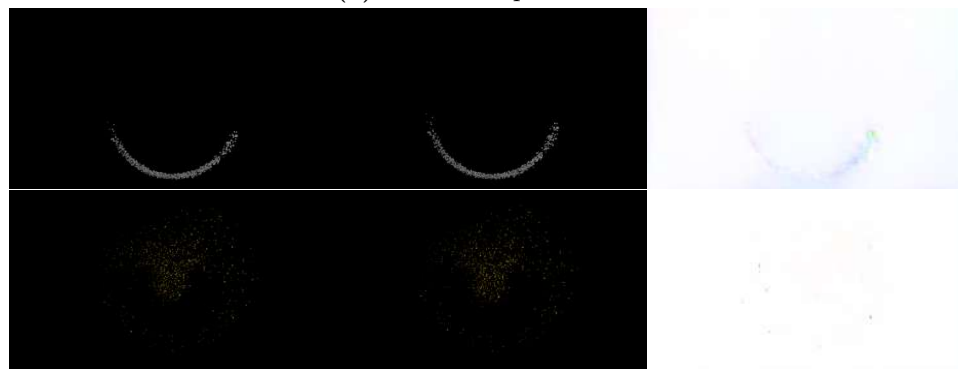
**Tabla 4.5:** Comparación de AEE y FAEE para modelos pre-entrenados en diferentes arquitecturas y modelos entrenados en RAFT. (Tomado de [22]).

Modelo	AEE	FAEE
Lucas Kanade	1.431	7.524
SPyNet-chairs	5.516	-
HD3-KITTI	29.62	-
RAFT-Chairs	0.695	5.85
RAFT-Things	0.882	3.326
unique-shadow	0.205	8.195
apricot-hill	0.287	<b>1.095</b>
neat-rain	0.207	4.004
dandy-tree	<b>0.124</b>	4.625

Para realizar la comparación cualitativa entre modelos entrenados y los pre-entrenados se toman dos muestras del conjunto de datos generado. En la figura 4.15 se muestran las imágenes utilizadas para realizar la evaluación así como las etiquetas correspondientes, asimismo, en la figura 4.14 se muestra el resultado de la estimación de cada modelo entrenado utilizando las mismas imágenes de la figura 4.15.



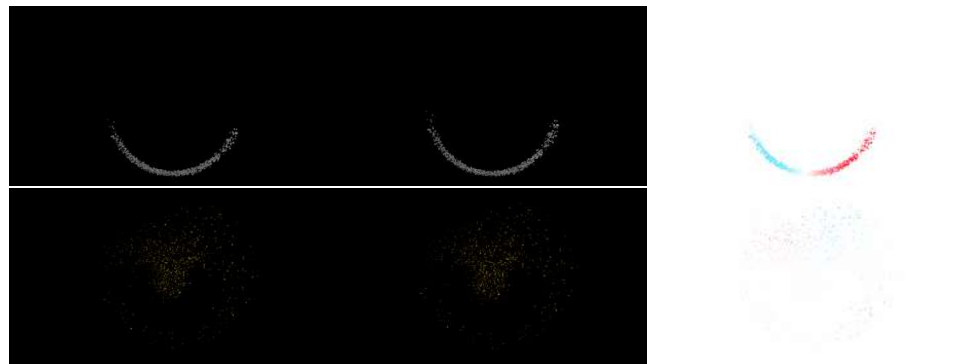
(a) RAFT-unique-shadow



(b) RAFT-apricot-hill

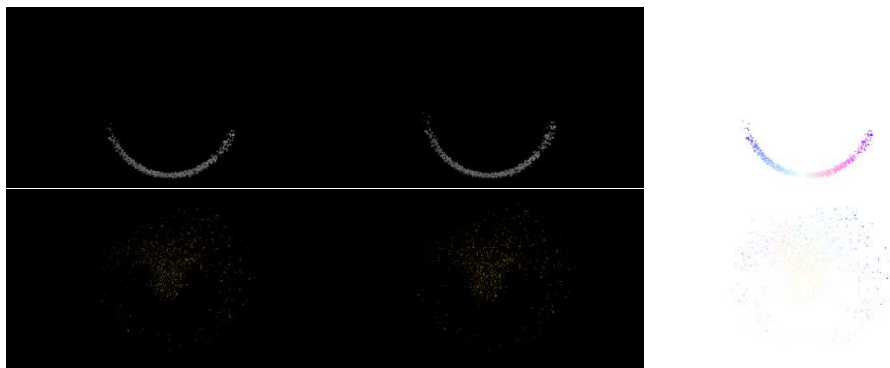


(c) RAFT-neat-rain



(d) RAFT-dandy-tree

**Figura 4.14:** Estimaciones de los modelos entrenados que poseen el menor FAEE. (Tomado de [22]).



**Figura 4.15:** Imágenes de referencia (*groundtruth*) para evaluación cualitativa. (Tomado de [22]).

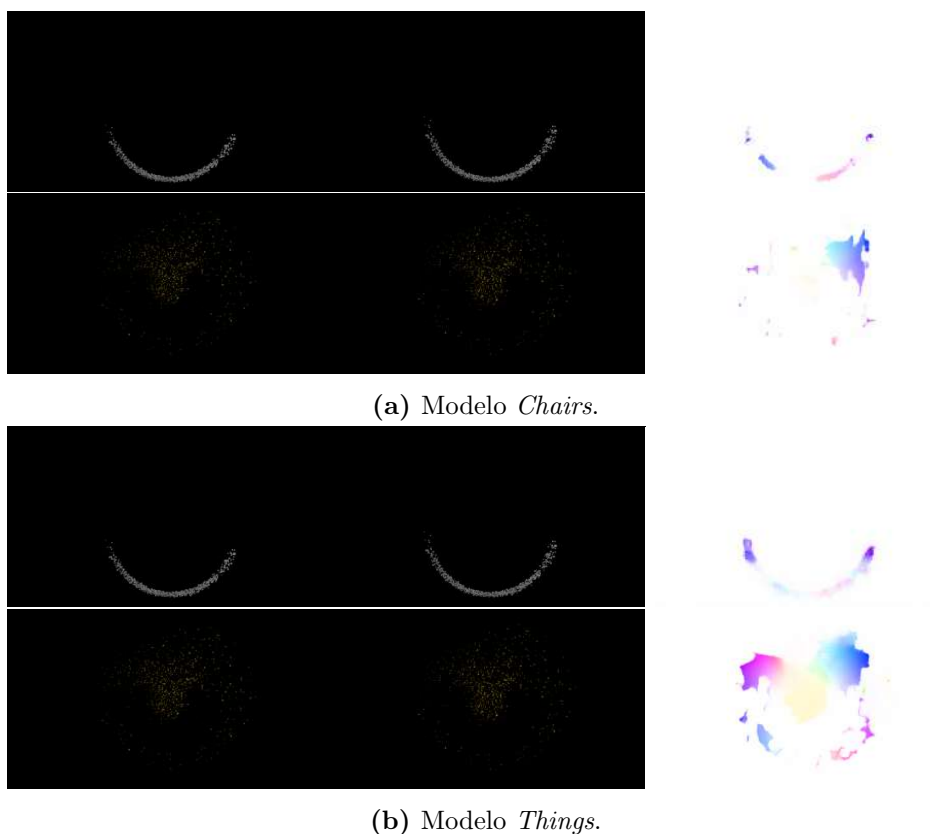
El resultado de la comparación contra el método de Lucas-Kanade se observa en la figura 4.16. Este método clásico es capaz de identificar correctamente la ausencia de movimiento, que se observa como una región blanca en la etiqueta. Esto coincide con el AEE de la evaluación realizada ya que el error es de aproximadamente 1.43 px. Como también se observa en la misma figura, el método estima efectivamente tanto la dirección del movimiento como la posición donde se ubica cada espora; sin embargo el área de estimación es mucho mayor al del área real de la espora. Consecuentemente el FAEE es mayor que el AEE con un valor de 7.52 px.



**Figura 4.16:** Evaluación cualitativa utilizando el método clásico de Lucas-Kanade. (Tomado de [22]).

De los modelos entrenados, el que posee el menor FAEE es *apricot-hill*, y el AEE es menor que los modelos pre-entrenados *Chairs* y *Things*; sin embargo, se observa en la figura 4.14b que la estimación en comparación a las de *Chairs* y *Things* de la figura 4.17 es prácticamente toda blanca, es decir, el modelo aprendió a estimar 0 en toda la imagen para reducir la función de costo. Por esto se consideró relevante incorporar la componente del FAEE a la función de costo, con el fin de evitar este tipo de problemas. A partir de la evolución del error en el entrenamiento de los modelos incorporando este factor, se determina que esto empeora la estimación.

Si se compara el AEE y el FAEE de los modelos *neat-rain* y *dandy-tree* con el modelo *apricot-hill* de la tabla 4.5 parece ser que ambos modelos poseen una peor estimación puesto que el error filtrado es mayor. Sin embargo, a partir de los resultados de las estimaciones de la figura 4.14b y 4.14d se puede observar que *neat-rain* a diferencia de *apricot-hill* (en la figura 4.14b) sí realiza una estimación donde hay movimiento. De igual forma aplica para el modelo *dandy-tree* de la figura 4.14d pero ahora se puede observar



**Figura 4.17:** Evaluación cualitativa utilizando los modelos preentrenados. (Tomado de [22])

de forma exacta que la estimación se realiza únicamente sobre el área de la espora; sin embargo, en ambos casos existe un alto nivel de sobreajuste, ya que las estimaciones se realizan únicamente en dos direcciones (representadas en las figuras por los colores rojo y azul).

El sobreajuste observado se puede atribuir a la arquitectura de la red. Esta realiza un submuestreo de las imágenes de entrada, por lo que para objetos muy pequeños, como es el caso de la espora, es posible que la información de estos píxeles se pierda y no sea tomada en cuenta en la inferencia del modelo.

En el RAFT se utiliza un GRU para realizar un refinamiento iterativo sobre la estimación de flujo óptico. Este refinamiento se realiza únicamente para el lote de la iteración de entrenamiento.

Finalmente se evaluó el tiempo de inferencia en promedio para un par de imágenes. Estos valores se compilan en la tabla 4.6.

#### 4.4.2. Gotas de agua

En el caso de las gotas de agua, se realiza el reentrenamiento y la evaluación utilizando como base el modelo preentrenado de *Flying Things*.

**Tabla 4.6:** Tiempo de ejecución de la evaluación de cada modelo sobre 1200 imágenes. (Tomado de [22]).

Modelo	Tiempo [ms]
<b>raft-chairs</b>	306
<b>raft-things</b>	320
<b>unique-shadow-17</b>	320
<b>apricot-hill-13</b>	322
<b>neat-rain-14</b>	325
<b>dandy-tree-10</b>	327
<b>radiant-sunset-21</b>	328

Se realizaron múltiples pruebas de entrenamiento variando hiperparámetros con el fin de ajustar la red para la identificación de gotas de agua. Entre estos se encuentra la presencia de ruido, tamaño del lote, *dropout*, la tasa de aprendizaje y número de iteraciones de la red. La función de costo utilizada en el entrenamiento y validación tiene la misma modificación que fue discutida en la sección 4.4.1 para incluir el error filtrado con un valor de  $\lambda = 0,05$ . La cantidad de iteraciones máxima fue de 10 000 y se utilizó *Adam* [30] para la optimización.

En la tabla 4.7 se encuentran los resultados de AEE de todos los entrenamientos y validación de los modelos entrenados para calcular el flujo óptico de gotas de agua. La prueba 1 obtuvo el valor más bajo de error en entrenamiento AEE=0.024, mientras que la prueba 3 obtuvo el AEE de validación más bajo con un valor de 0.154. La prueba 8 obtuvo los valores de error más altos en entrenamiento AEE=0.226 y validación AEE=0.960.

**Tabla 4.7:** Valores de AEE para los modelos entrenados. (Tomado de [50])

Modelo	AEE Entrenamiento	AEE validación
<b>Prueba 1</b>	<b>0.024</b>	0.155
<b>Prueba 2</b>	0.027	0.346
<b>Prueba 3</b>	0.023	<b>0.152</b>
<b>Prueba 4</b>	0.056	0.401
<b>Prueba 5</b>	0.101	0.310
<b>Prueba 6</b>	0.217	0.612
<b>Prueba 7</b>	0.216	0.612
<b>Prueba 8</b>	0.227	0.960

El valor del AEE para la arquitectura RAFT inicial con el conjunto de datos *Flying Things* es de 0.396 mientras que el valor de entrenamiento de la red posterior al entrenamiento con el conjunto sintético gotas haciendo la validación es de 0.152 en el entrenamiento Prueba 3, mostrando una disminución del valor en un factor de 2.6.

### 4.4.3. Resultados cualitativos

En la figura 4.18 se muestra la predicción de tres redes neuronales utilizando una imagen de una gota real en movimiento. En todos los casos se detecta correctamente el movimiento hacia abajo de la gota; sin embargo el movimiento del agua que ha impactado no se detecta correctamente. En los modelos prueba 1 y prueba 3 se muestra parcialmente el movimiento hacia arriba a la derecha de estos movimientos. Nótese que la sombra de la gota no muestra una falsa detección del flujo óptico.



**Figura 4.18:** Predicción del movimiento del agua utilizando imágenes reales a 600 fps, para los modelos prueba 1, prueba 3 y prueba 5.

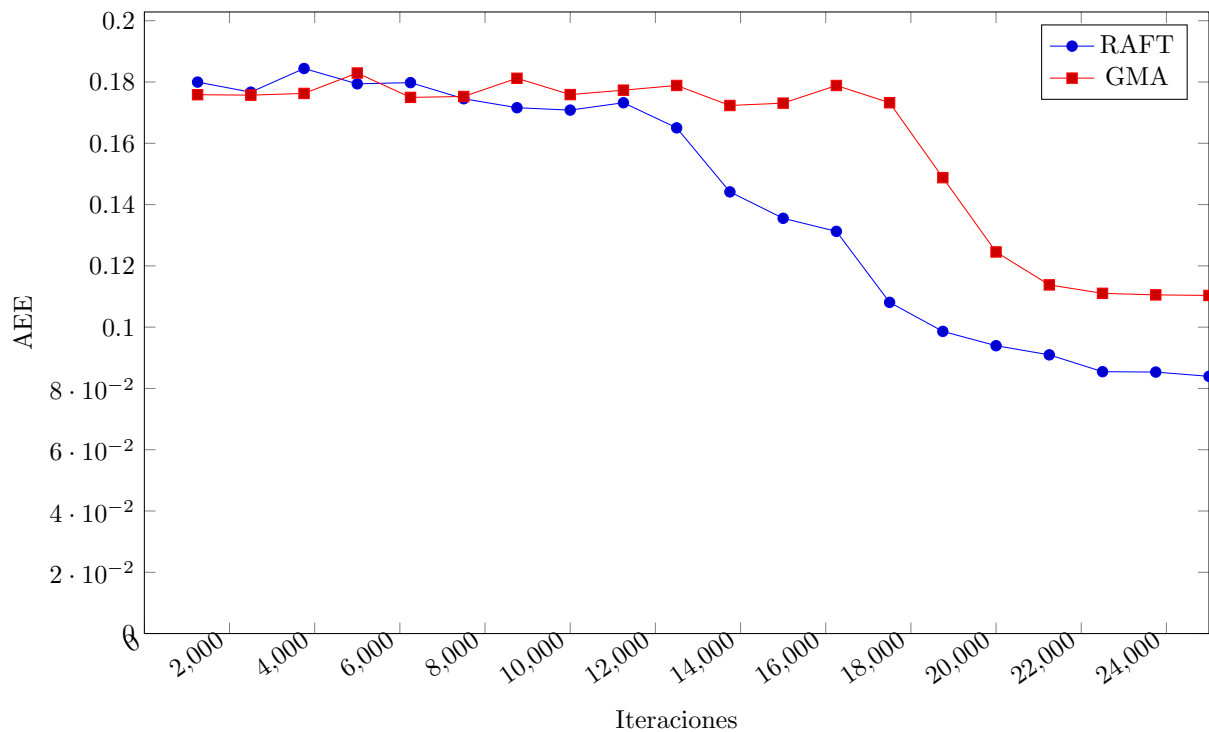
## 4.5. Entrenamiento de una red integrada para flujo óptico

A manera de conclusión de los resultados de agua y esporas por separado, en donde siempre fue RAFT el mejor método, lo que conduce a seleccionar para el método integrado para la estimación del movimiento de ambos tipos de objetos a usar RAFT y el método más reciente GMA.

En la figura 4.19 se muestra el AEE promedio en función de la cantidad de iteraciones, para las dos redes entrenadas RAFT y GMA para la detección del movimiento en conjunto de las gotas de agua y de las esporas. En ambos casos estos entrenamientos se muestra el resultado para la red con el AEE entrenadas con el conjunto de datos de gotas de agua y esporas.

Como se detalla en la tabla 4.8, ambos alcanzaron un AEE muy inferior a 1 (siendo 1 una diferencia de un solo píxel). Dado que, en la inferencia final, los valores se determinan para cada uno de los píxeles, esto significa que el valor de cada uno de ellos debería ser correcto. Entre estos dos, la red RAFT alcanzó un valor final más bajo.

Otra punto que hay que tener en cuenta aquí es que, en lugar de entrenar las redes hasta que no se obtuviera ninguna mejora significativa, éstas se detuvieron en un valor fijo de 25 000 iteraciones. Esto significa que sería posible mejorar aún más estos valores finales de rendimiento entrenando durante más tiempo, como referencia para los entrenamientos mostrados en la figura 4.19, la red RAFT tomo un tiempo de 11 h y GMA 17 h para llegar a los valores finales mostrados.



**Figura 4.19:** AEE en función de la cantidad de iteraciones durante el entrenamiento.

	RAFT	GMA
AEE	0.2172	0.2007

**Tabla 4.8:** Valores finales de AEE de la gráfica 4.19.

En la figura 4.20 se muestran los resultados cualitativos de la estimación de flujo óptico.



(a) Resultado de red integrada RAFT para flujo óptico. (b) Resultado de red integrada GMA para flujo óptico.

**Figura 4.20:** Resultados de flujo óptico para las redes integradas.

Las figuras 4.20a y 4.20b muestran los resultados de ambas redes entrenadas. En la figura 4.20a, donde se ve el resultado de RAFT, se puede ver correctamente tanto el movimiento en la nube como en el agua. La dirección y la magnitud correctas del movimiento son visualmente muy cercanas.

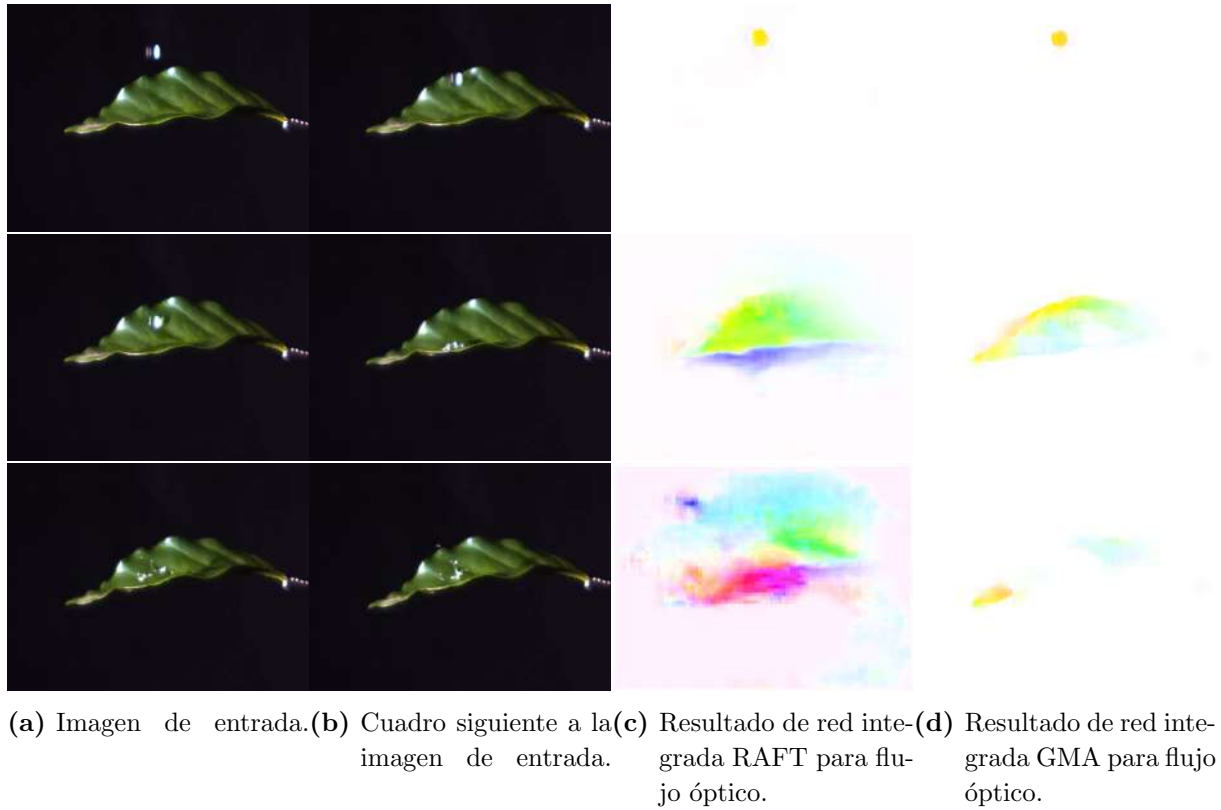
En la figura 4.20b, se muestra el resultado de la Red GMA, en este caso la polaridad detectada para la salpicadura de agua es correcta, mostrándose aquí como el tono correcto, mientras que el movimiento de la nube no se detecta en absoluto. La diferencia en la saturación es principalmente debido a un cambio en la normalización debido a la detección de la nube de esporas en una de las imágenes.

En ambos casos, la visualización muestra una limitación de la red GMA, y es que la salida de la red está submuestreada en relación con la entrada, y esto significa que bloques de píxeles cercanos tendrán la misma salida. Y aunque no se muestra en el conjunto de datos actual, el movimiento de los píxeles individuales no se mostrará correctamente.

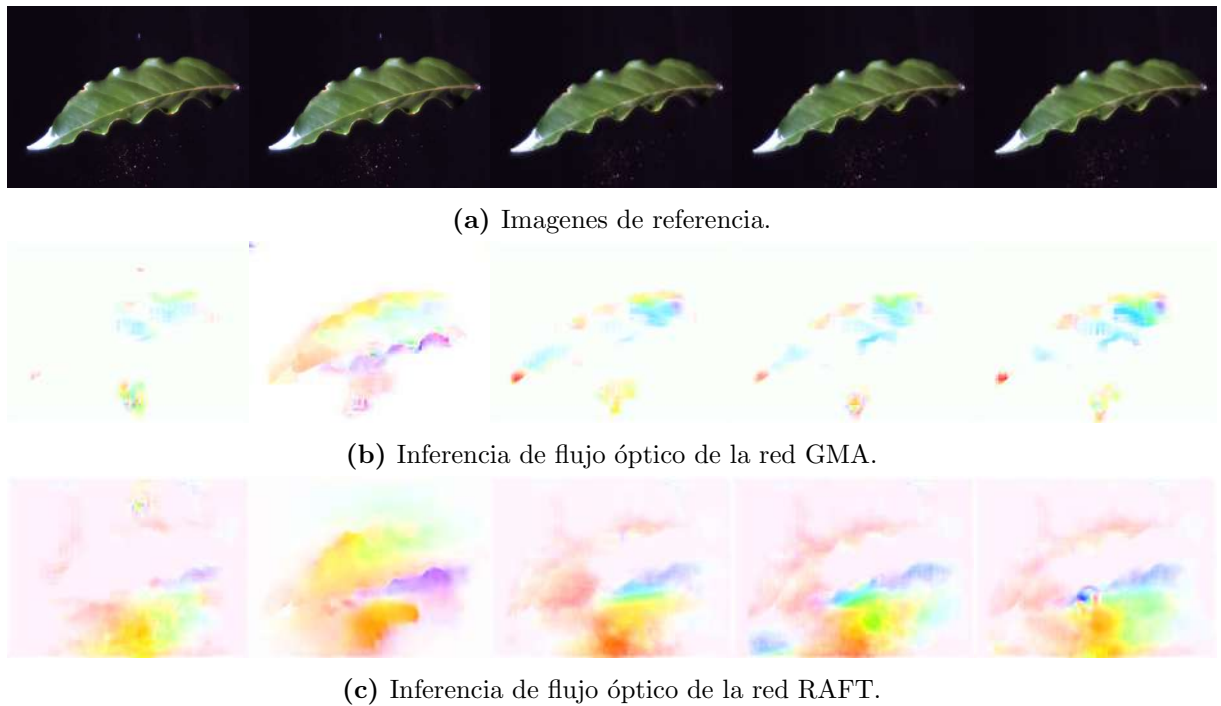
Las figuras 4.21a y 4.21b muestran fotogramas consecutivos de una captura real. Esto muestra el movimiento antes de que una gota de agua golpee una hoja. En este caso, ambas redes son capaces de detectar correctamente el movimiento en los fotogramas. El resultado para la RAFT, mostrado en la figura 4.21c, es ligeramente más ruidoso, con detecciones incorrectas alrededor de la imagen, mientras que la salida de la GMA, mostrado en la figura 4.21d contiene su salida más estrechamente con el área donde la gota de agua está realmente. Se debe tomar en cuenta, que para la visualización, se normaliza la saturación en las imágenes por lo que la comparación de la magnitud del movimiento entre dos imágenes no se puede realizar solo con respecto a la saturación de sus colores.

La figura 4.22 muestra una serie de imágenes reales en las que está presente el movimiento de las esporas, y las correspondientes inferencias para los pares de imágenes. La inferencia de GMA se muestra en la figura 4.22b. Su inferencia muestra parte del movimiento correctamente, sin embargo, el movimiento completo no se detecta correctamente y tiene





**Figura 4.21:** Resultados de flujo óptico utilizando imágenes reales.



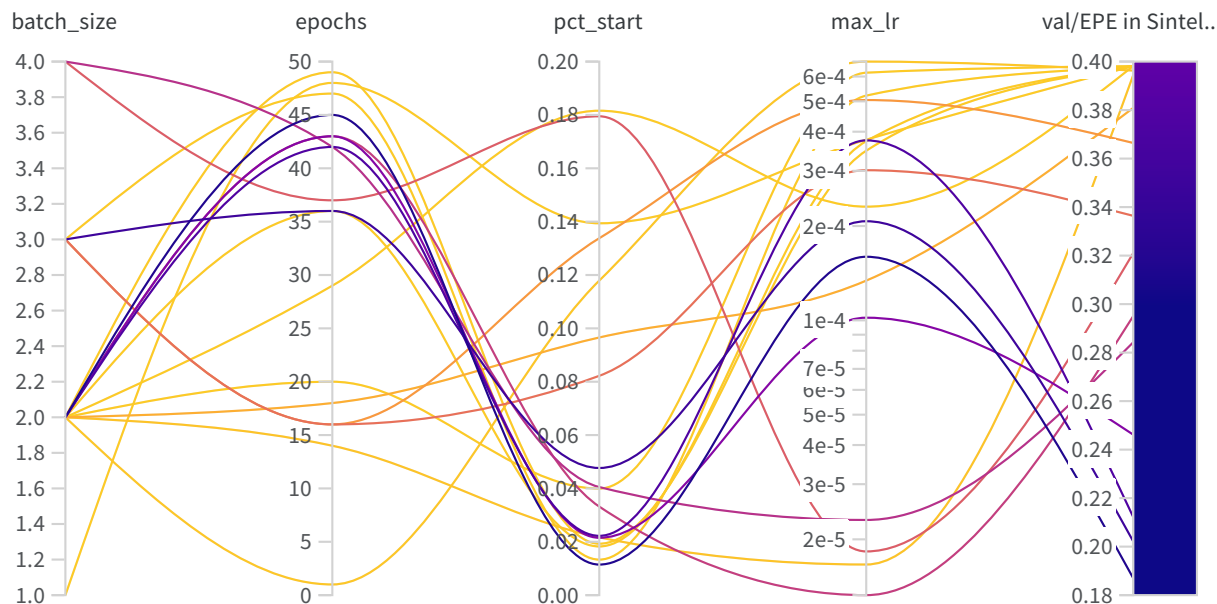
**Figura 4.22:** Inferencia de movimiento de esporas.

algo de *bleeding* cuando el movimiento está cerca de la hoja. La propia hoja también tiene detecciones incorrectas, pero como se ha mencionado antes, esto se debe probablemente a la falta de hojas en movimiento en el conjunto de datos.

La red RAFT produce una salida ruidosa, mostrada en la figura 4.22c, aunque el movimiento de las esporas genera la inferencia de movimiento, pero éste es mucho mayor que en la zona donde se encuentran las esporas, y también tiene una detección incorrecta en esta zona.

La figura 4.33 muestra una visualización de los diferentes hiperparámetros y sus combinaciones para optimizarlos. La distribución de estos parámetros se obtiene a partir de un modelo bayesiano controlado por el sistema WandDB [49], que está configurado para minimizar el AEE. Nótese que varias de las combinaciones del modelo tienen el mismo resultado de aproximadamente 0.40, que es el valor inicial de los pesos preentrenados. Esto significa que para estas combinaciones, el entrenamiento no pudo mejorar el valor inicial. Tras esta optimización de los hiperparámetros, el AEE resultante alcanzó un valor optimizado de 0.187.

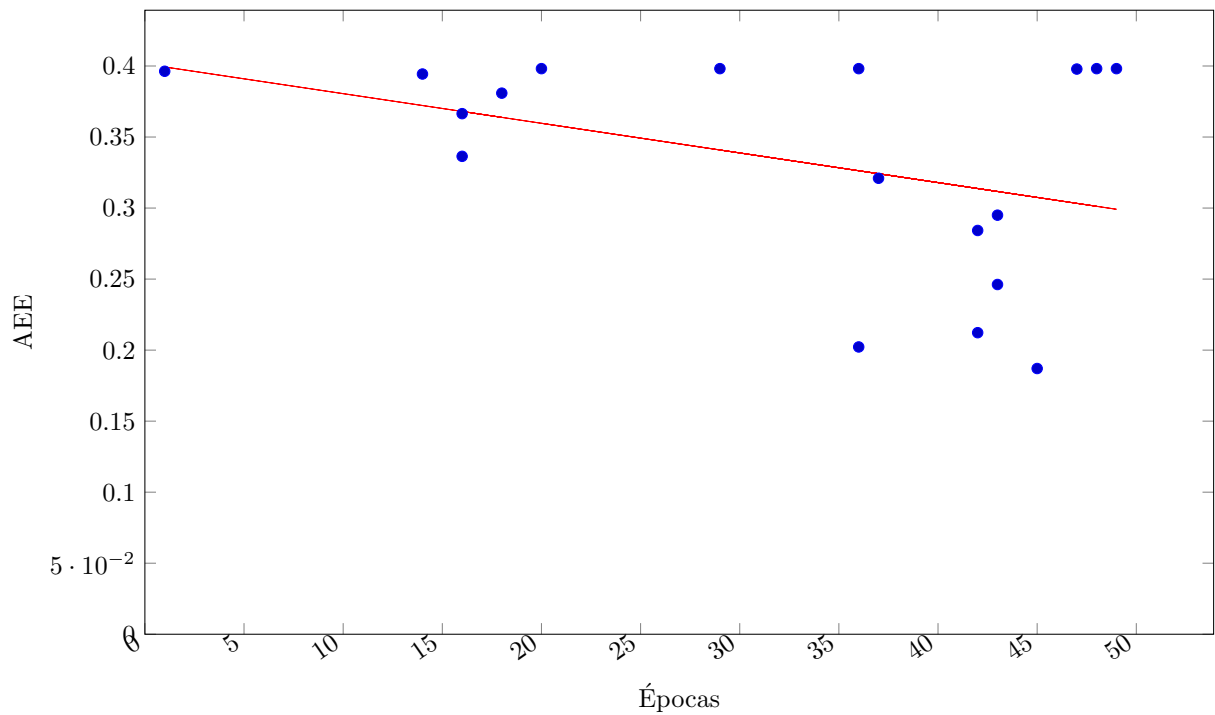
Se utiliza una política de tasa de aprendizaje de OneCycle [48], esta política va de un valor inicial a un valor máximo de tasa de aprendizaje en un porcentaje dado de la cantidad total de épocas, volviendo luego a un valor menor que la tasa de aprendizaje inicial al final del programa de aprendizaje.



**Figura 4.23:** Combinaciones de hiperparámetros y AEE resultantes. Esta figura puede ser visualizada interactivamente [aquí](#)

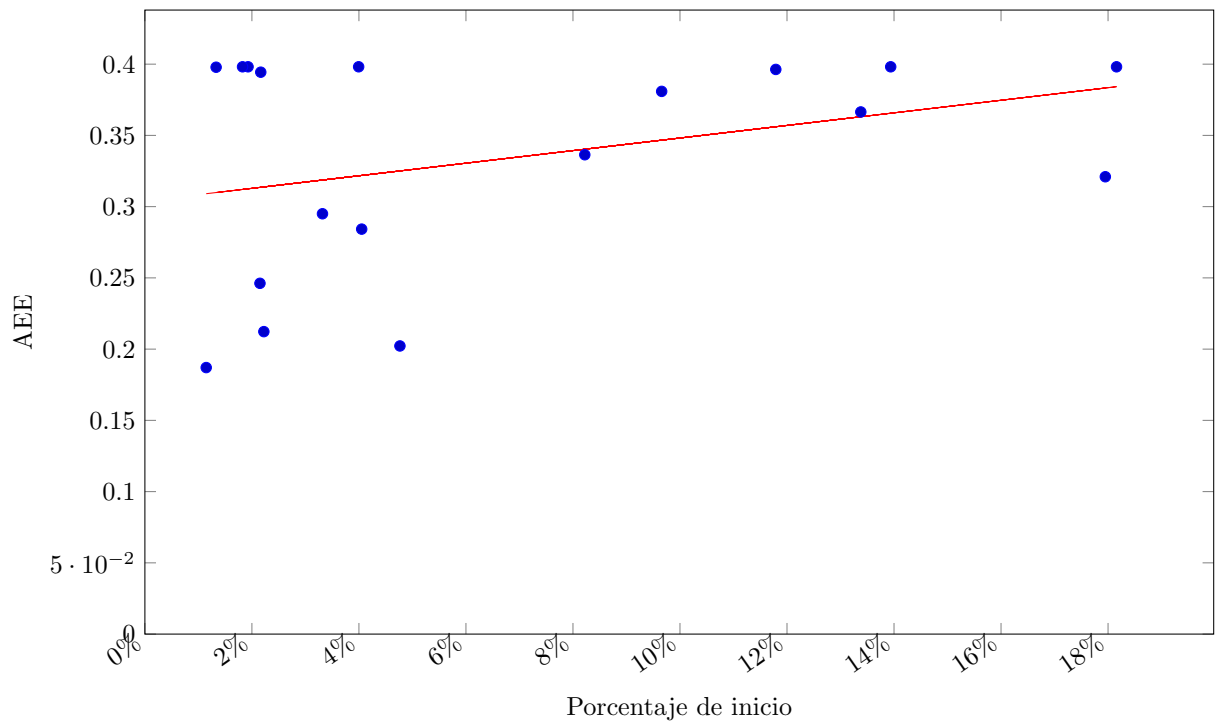
La figura 4.24 muestra la distribución de la AEE resultante en función del número de épocas. Excepto en el caso de las combinaciones de hiperparámetros que no tuvieron ninguna mejora, una mayor cantidad de épocas conduce a una menor AEE.

La figura 4.25 muestra la distribución de la AEE en función del porcentaje de la cantidad



**Figura 4.24:** AEE en función de la cantidad de épocas.  $R = -0,393$

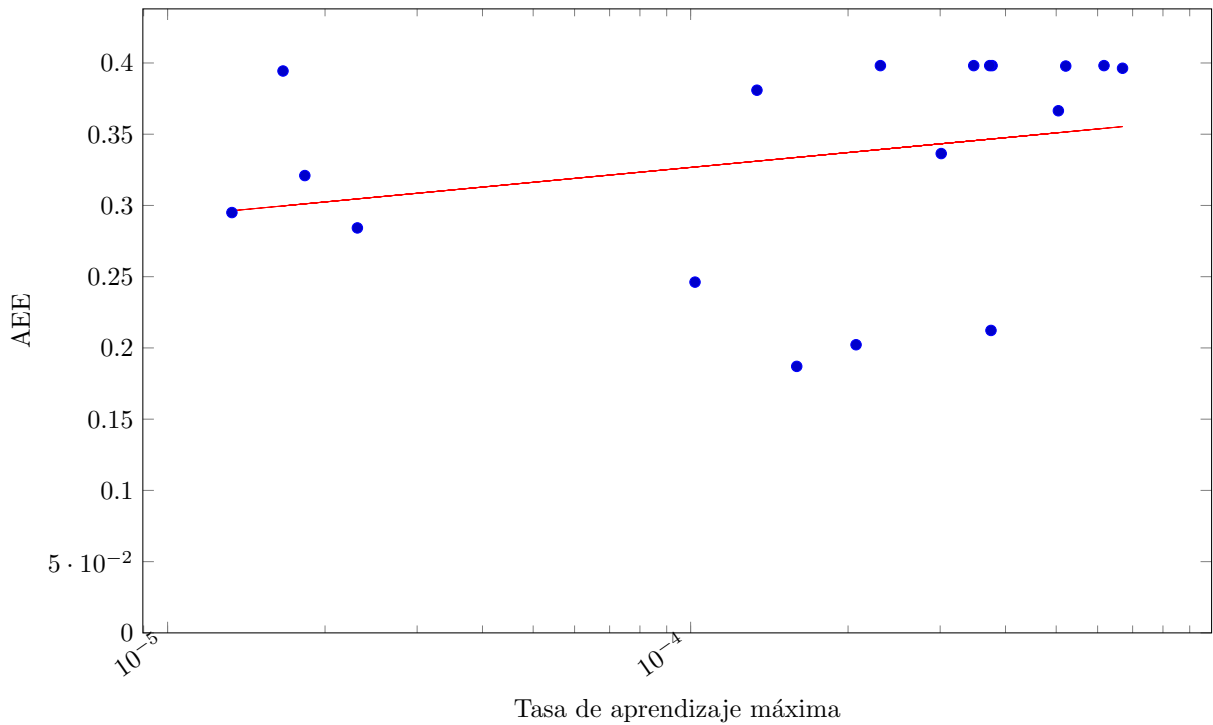
total de épocas. En este caso, un valor mayor de este hiperparámetro conduce a un aumento de la AEE.



**Figura 4.25:** AEE en función del porcentaje de épocas del punto máximo de la tasa de aprendizaje.  $R = 0,337$

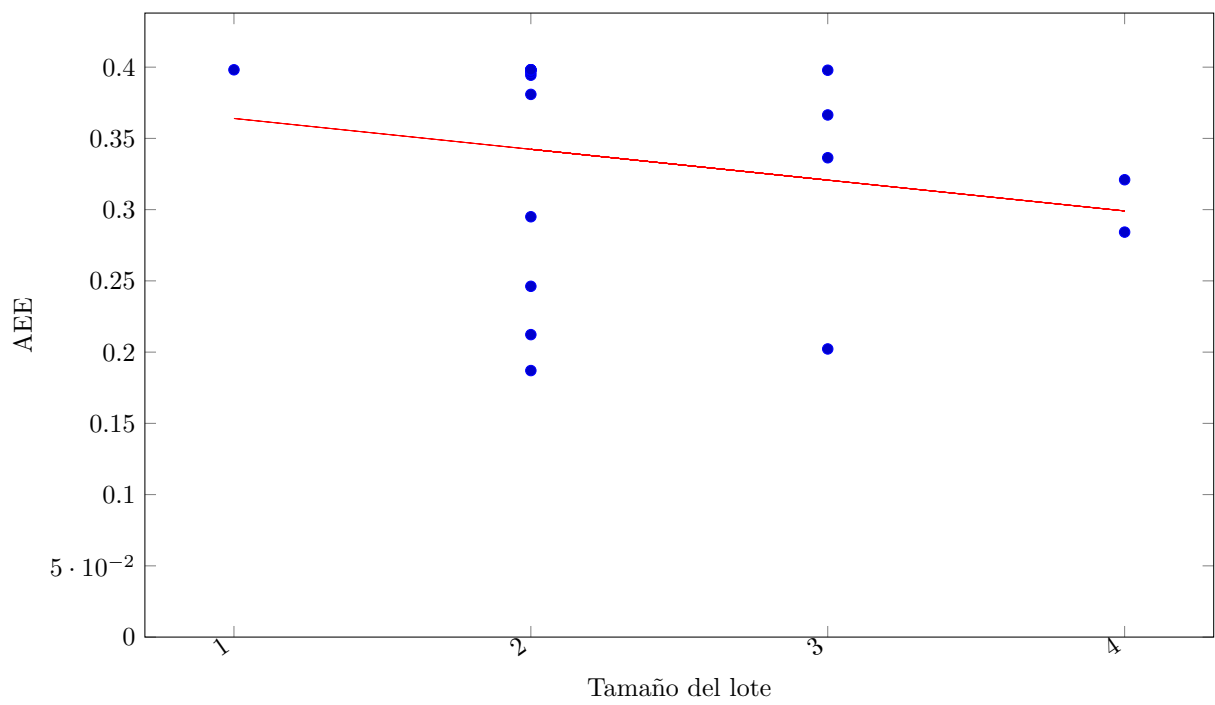
La Figura 4.26 muestra las distribuciones de la AEE en función de la tasa máxima de

aprendizaje. La tasa de aprendizaje sigue una distribución logarítmica entre  $1e-5$  y  $1e-3$ . Existe una correlación positiva entre la AEE y la tasa máxima de aprendizaje.



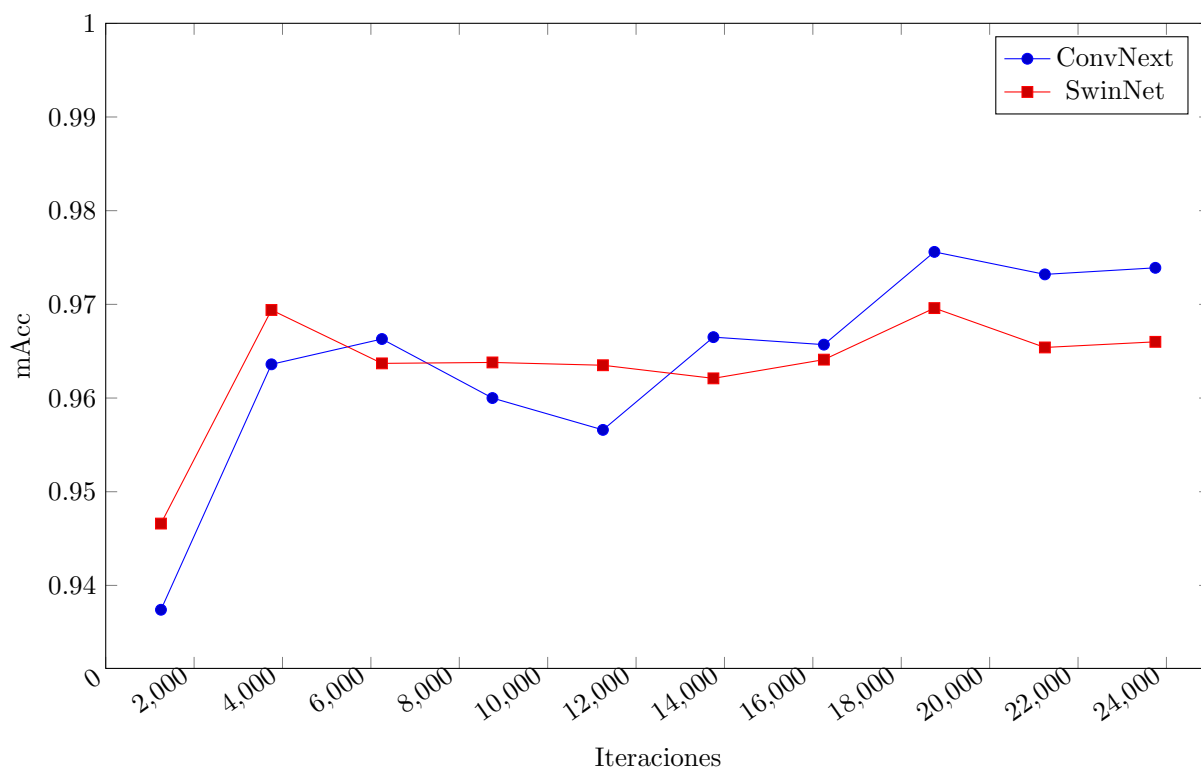
**Figura 4.26:** AEE en función de la tasa de aprendizaje máxima.  $R = 0,424$

La figura 4.27 muestra la distribución del AEE en función del tamaño del lote. La correlación en este caso muestra que un tamaño de lote mayor conduce a una AEE menor. Nótese, sin embargo, que ambos casos de borde sólo tienen un par de pruebas en ellos. Además, el límite superior, que podría conducir a una AEE más baja, está limitado por el hardware.



**Figura 4.27:** AEE en función del tamaño del lote.  $R = -0,218$

## 4.6. Entrenamiento de red integrada para segmentación



**Figura 4.28:** Valor medio de precisión en todas las clases en función de cantidad de iteraciones durante el entrenamiento.

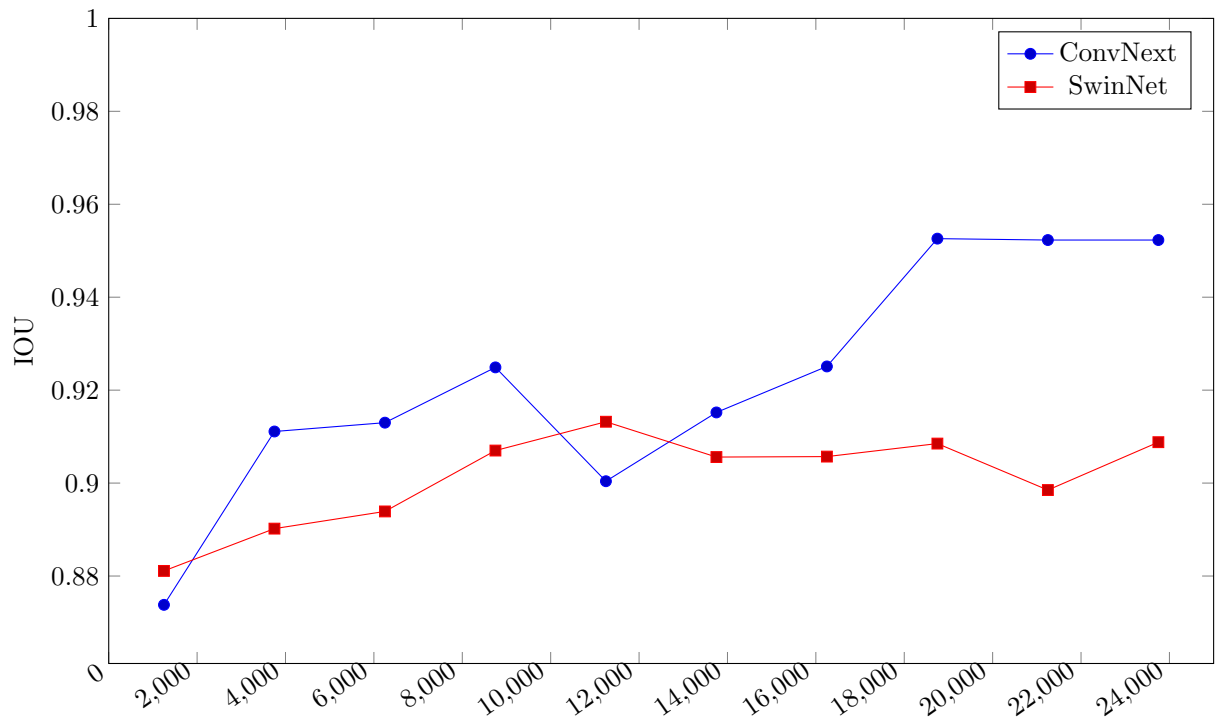
	ConvNext	SwinNet
<b>mAcc</b>	0.9094	0.9052
<b>mIOU</b>	0.9536	0.9515

**Tabla 4.9:** Valores máximos de mAcc y mIOU de la gráficas 4.28 y 4.29.

La figura 4.28 muestra la exactitud promedio mAcc durante el entrenamiento de las redes ConvNext y SwinNet. En ambos casos, ambas alcanzan casi un valor ideal de 1.

Para estas redes, el entrenamiento también se ejecutó durante un valor establecido de 25 000 iteraciones, por lo que el resultado final encontrado aquí podría ser mejorado. Sin embargo, como puede verse en el gráfico, la mayor parte de la mejora se alcanzó ya en las primeras 4000 iteraciones.

La figura 4.29 muestra la intersección sobre la unión (mIOU) para ambas redes entrenadas. Aquí, la red ConvNext muestra un valor más alto para su medida. Mientras que la red SwinNet no sólo muestra un valor más bajo, sino que su valor máximo se alcanzó en torno a las 8000 iteraciones con una reducción del mIOU después de eso.



**Figura 4.29:** Valor medio de IoU en todas las clases en función de cantidad de iteraciones durante el entrenamiento.

Al igual que con los otros entrenamientos, el valor podría mejorarse ejecutando este durante más iteraciones, ya que este entrenamiento se detuvo en un valor fijo de 25 000 iteraciones.

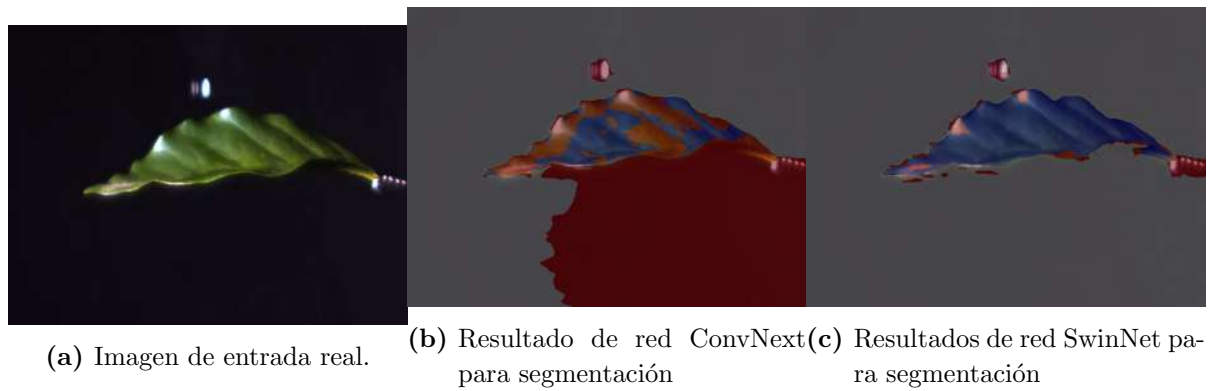


(a) Datos de referencia de etiqueta (b) Resultado de red ConvNext (c) Resultados de red SwinNet para segmentación.

**Figura 4.30:** Resultados de segmentación.

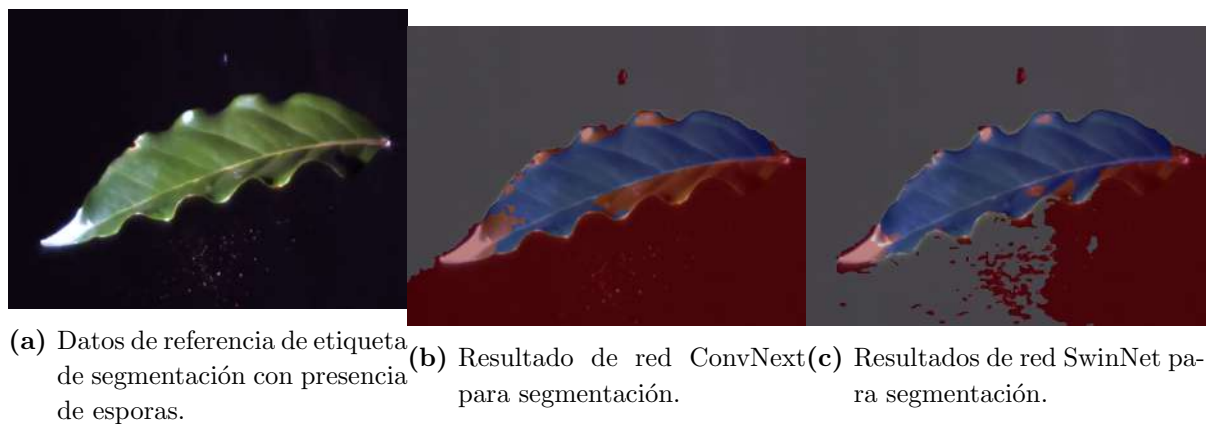
La figura 4.30 muestra la verdad de referencia y las inferencias resultantes de ambas redes para una de las imágenes de entrenamiento. Las inferencias resultantes en ambos casos coinciden estrechamente con los valores esperados, incluso cuando hay reflejos y sombras que se muestran en la imagen.

A diferencia de las redes de flujo óptico, estas redes no sufren un problema de muestreo, ya que los resultados de los píxeles no se dan en pequeños bloques, sino píxel a píxel.



**Figura 4.31:** Resultados de segmentación utilizando imágenes reales.

La figura 4.31 muestra los resultados cualitativos de una imagen real utilizando las redes de segmentación entrenadas para la inferencia. Ambas redes son capaces de segmentar correctamente la gota de agua, sin embargo con la hoja tienen problemas. La red SwinNet puede, en su mayoría, segmentar correctamente la imagen, pero cuando ésta es demasiado reflectante, comienza a clasificarla incorrectamente como una gota de agua. La red ConvNext, aunque tiene detecciones correctas para la hoja, una extensa sección de la misma está siendo clasificada incorrectamente como agua, mientras que también tiene un gran *bleeding* del fondo siendo clasificado como una gota de agua.



**Figura 4.32:** Resultados de segmentación.

Los resultados para una imagen real con esporas presentes se muestran en la figura 4.32. Ninguna de las dos redes es capaz de inferir correctamente las esporas con la etiqueta correcta. En ambos casos, las esporas se etiquetan como agua. La red SwinNet, aunque no es capaz de etiquetar las esporas, puede separarlas del fondo claramente.

La figura 4.33 muestra una visualización de los diferentes hiperparámetros y sus combinaciones. Esta distribución también es controlada por el modelo bayesiano de WandDB. En este caso, está configurado para maximizar el mIoU que tiene un valor máximo de 1. Se realizaron un total de 30 entrenamientos.

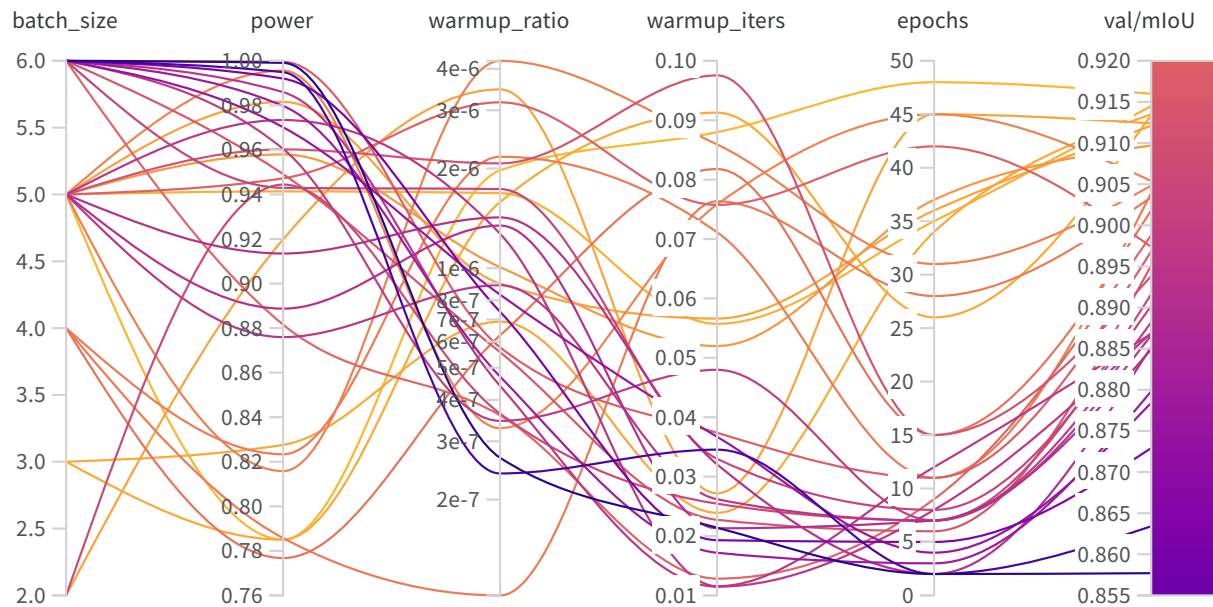
Todos estos entrenamientos utilizan una tasa de aprendizaje “Poly” [60] que es similar a



el OneCycle. Inicialmente utiliza un calentamiento lineal hasta que alcanza una tasa de aprendizaje máxima, y luego decae de acuerdo con la siguiente fórmula:

$$lr = max\_lr \left( 1 - \frac{iter}{max\_iter} \right)^{power} \quad (4.3)$$

Tras esta optimización, el mIoU alcanzó un valor máximo de 0.916.

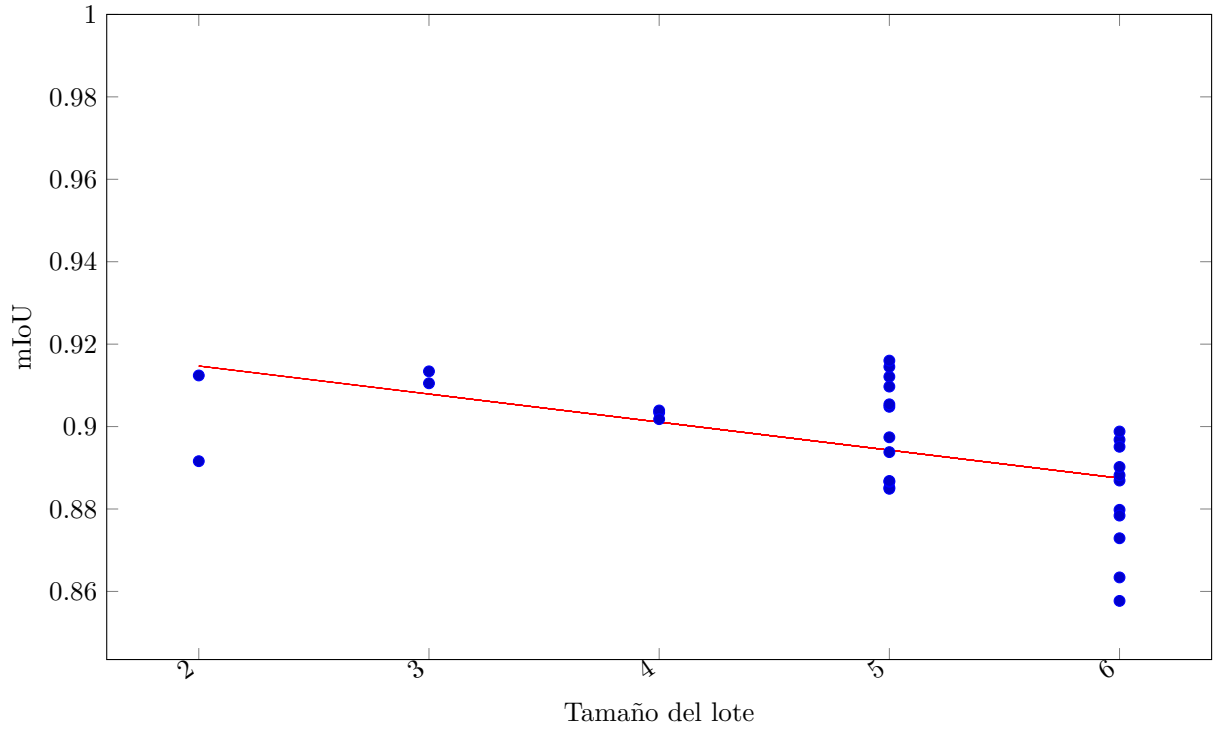


**Figura 4.33:** Combinaciones de hiperparámetros y mIoU resultantes. Esta figura puede ser visualizada interactivamente [aquí](#).

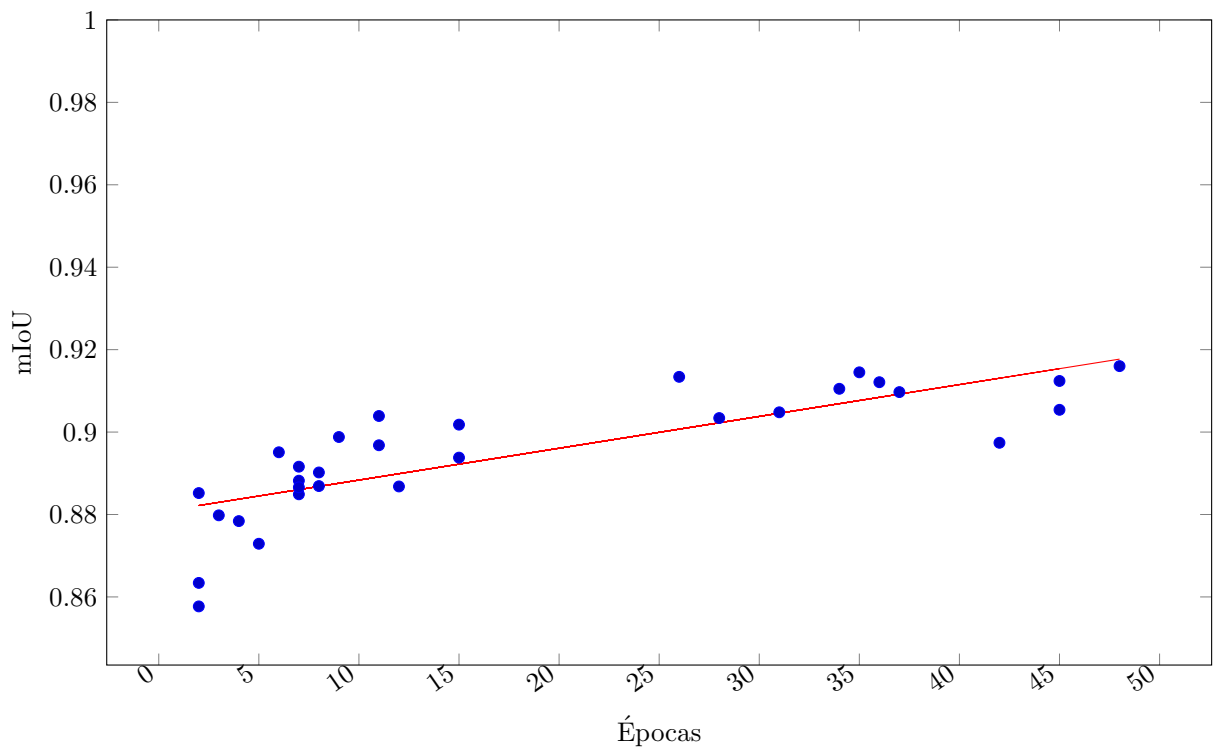
La figura 4.34 muestra la distribución del mIoU en función del tamaño del lote. La distribución indica una correlación negativa entre estas dos variables. Sin embargo, la mayoría de los entrenamientos se realizaron con los tamaños de lote más altos.

La figura 4.35 muestra la distribución del mIoU en función del número de épocas. Esta distribución tiene una correlación positiva. Sin embargo, un entrenamiento que utilizó sólo 25 épocas, la mitad del límite superior de 50 épocas, alcanzó una AEE de 0.9134 (en comparación, el mejor entrenamiento que entrenó durante 48 épocas y alcanzó una AEE de 0.916).

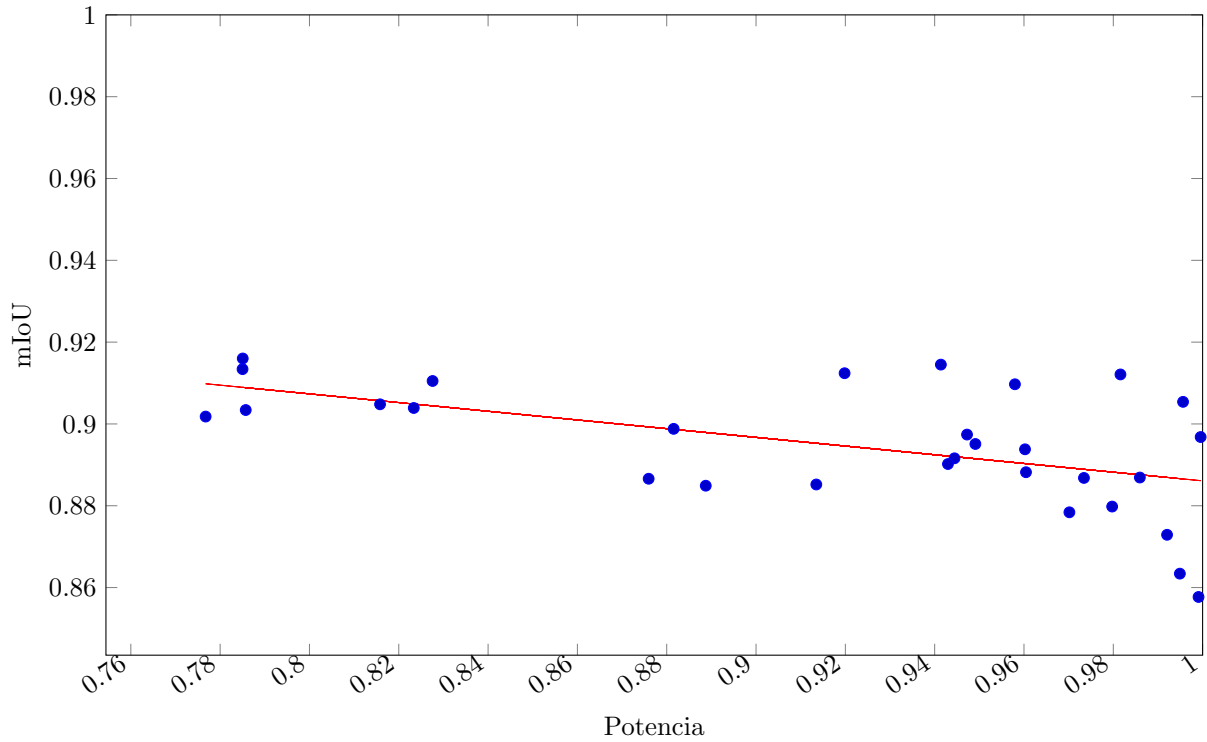
La figura 4.36 muestra la distribución del mIoU en función de la potencia. Esta distribución tiene una correlación negativa. La figura 4.37 muestra la distribución del mIoU en función del porcentaje de inicio, que tiene una correlación positiva, y finalmente la figura 4.38 muestra la distribución del mIoU en función de la tasa máxima de aprendizaje. Ninguna de estas tres variables muestra una alta correlación con el mIoU, sin embargo, dado que todas ellas afectan al comportamiento del planificador cambiante, una de ellas afecta a cómo todas las demás afectan al mIoU resultante.



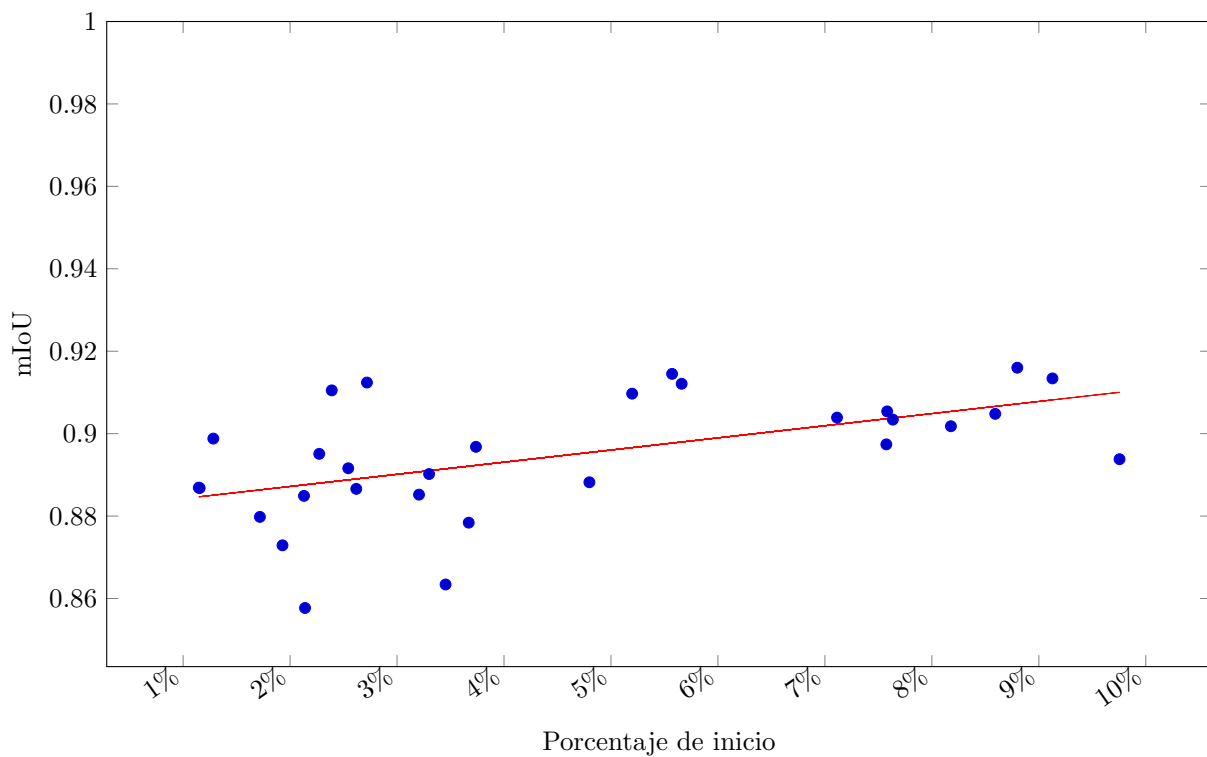
**Figura 4.34:** AEE en función de la cantidad del tamaño del lote. Coeficiente de correlacion para la linea de mejor ajuste  $R = -0,451$



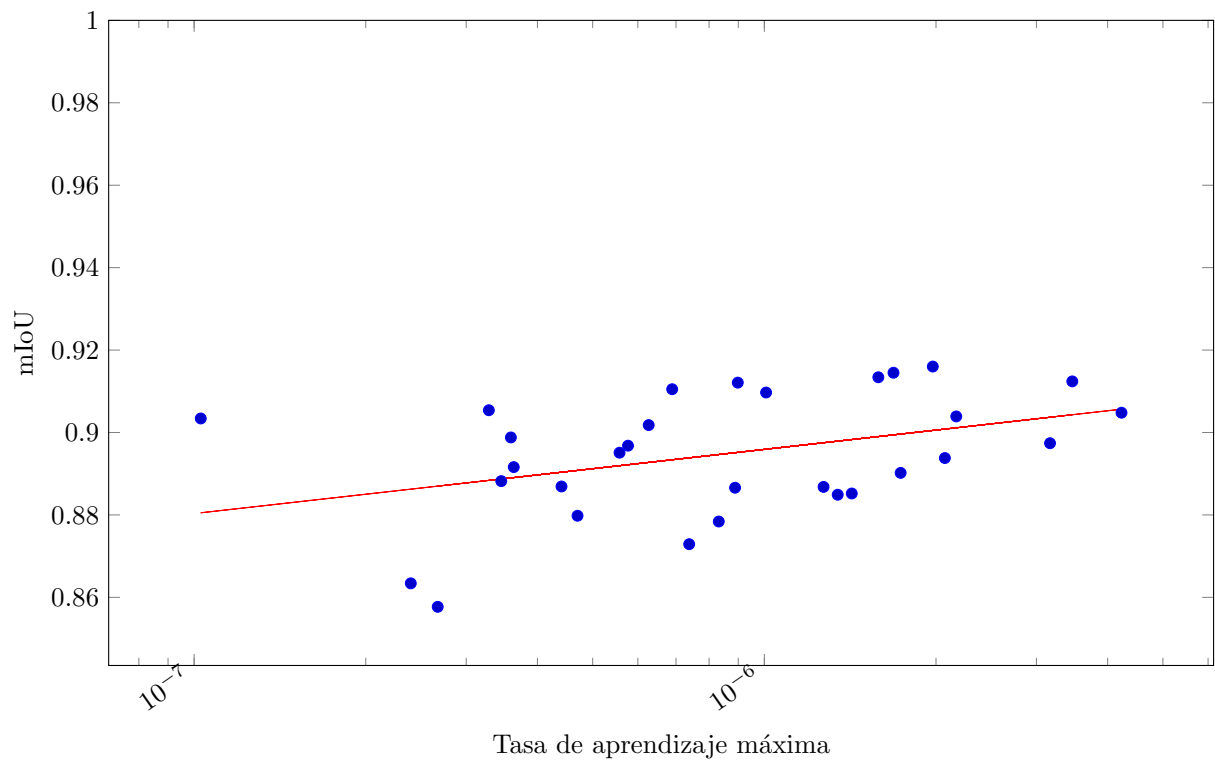
**Figura 4.35:** AEE en función de la cantidad de épocas. Coeficiente de correlacion para la linea de mejor ajuste  $R = 0,625$



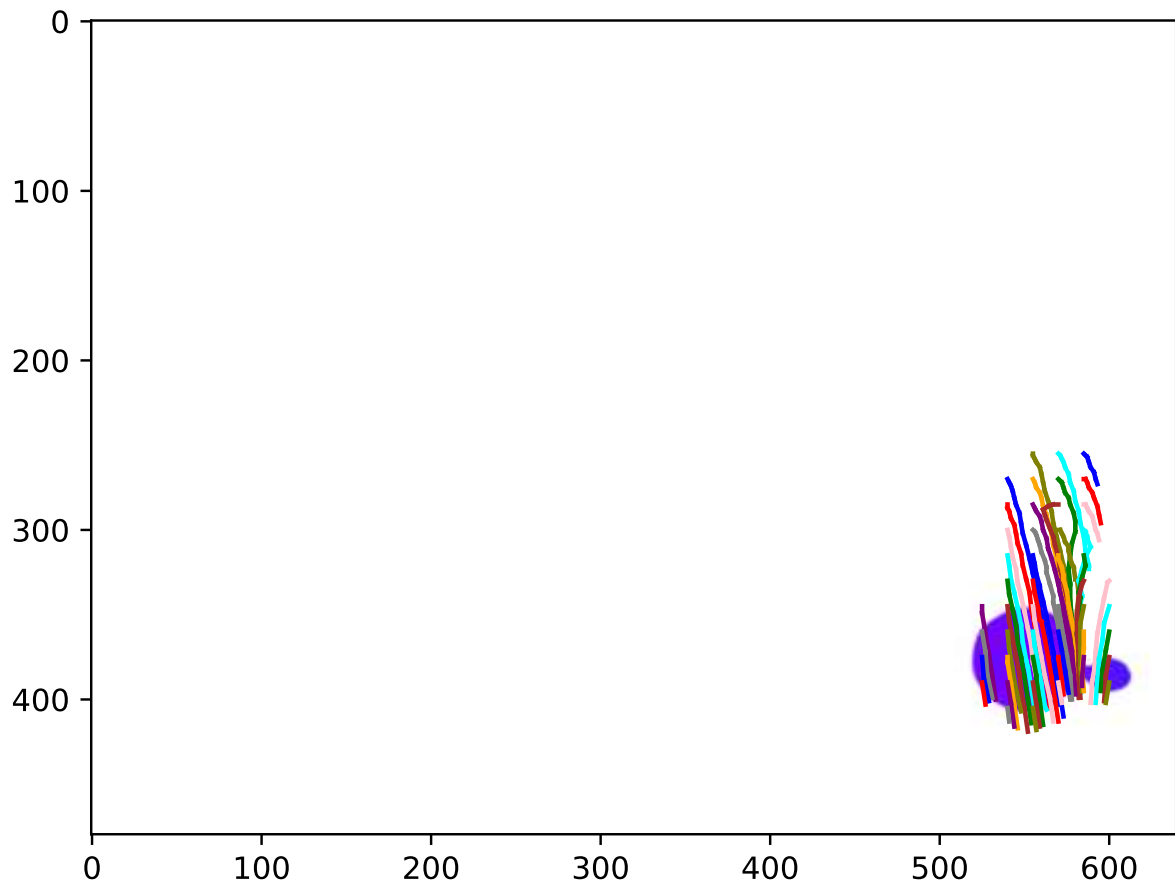
**Figura 4.36:** AEE en función del parámetro de potencia en el calendarizador de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste  $R = -0,410$



**Figura 4.37:** AEE del valor máximo de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste  $R = 0,486$



**Figura 4.38:** AEE en función del porcentaje de épocas del valor máximo de la tasa de aprendizaje. Coeficiente de correlacion para la linea de mejor ajuste  $R = -0,315$



**Figura 4.39:** Visualización del movimiento a lo largo de diferentes cuadros

## 4.7. Visualización

La figura 4.39 muestra la visualización del movimiento en una serie de imágenes. Se define una cuadrícula inicial de puntos  $y$ , a medida que se infiere el movimiento en una serie de imágenes, los puntos de la cuadrícula se trazan en la imagen. Esto da lugar a los trazos de “cabello” como trayectorias en la imagen.

# Capítulo 5

## Conclusiones

La enfermedad de la roya de café afecta las economías y vidas de la población en países productores. Como aporte al estudio de los fenómenos asociados a la propagación del hongo que la causa, este trabajo propone una estrategia de rastreo de esporas y de gotas de agua en secuencias de imágenes.

El sistema actual consta de un generador de datos sintéticos que representa al montaje físico real, y de dos modelos entrenados para la inferencia tanto del flujo óptico como de la segmentación de las imágenes. Además, el sistema también incluye un sistema real utilizado para capturar imágenes del evento estudiado.

Se entrenaron redes de flujo óptico individuales para la inferencia del flujo óptico de las esporas y las gotas de agua. Estas redes fueron entrenadas en base a modelos preentrenados, con esporas los modelos preentrenados *Chairs* y *Things*, ambos utilizando la arquitectura RAFT; estos modelos también fueron superiores a cualquier otra arquitectura.

Se determinó que la incorporación del factor FAEE en la función de costes aumenta el valor al que converge la función de costes, aumenta el AEE y el FAEE. Este comportamiento es un indicador de que esta modificación de la función de costes no permite optimizar el FAEE en las condiciones de ensayo realizadas.

A partir del análisis de los resultados de las estimaciones de los modelos entrenados de sólo esporas, se determina que dos modelos mejoran la estimación sólo en las regiones donde hay movimiento respecto a los modelos preentrenados. Sin embargo, la dirección del movimiento está sobreajustada a dos direcciones específicas. La FAEE de los modelos entrenados es 1.39 veces mayor que la FAEE del modelo preentrenado. Además, el modelo de *apricot-hill* tiene una AEE 3.07 veces menor que el mejor modelo preentrenado, pero una FAEE 2.46 veces mayor. Esto indica que estos modelos son mejores en la predicción de la falta de movimiento en el fondo, pero la predicción del movimiento del objeto empeoró.

El modelo preentrenado tiene una FAEE 2.26 veces menor que el modelo clásico de Lucas-Kanade. La comparación de la FAEE de los modelos mejor entrenados de la evaluación cualitativa muestra que la FAEE es 1.89 veces inferior a la de Lucas-Kanade.

Para las gotas de agua, el modelo sintético utilizado para la generación de datos representa correctamente la iluminación, los fondos, el número de objetos y los fotogramas por segundo de la escena.

También se entrenaron dos redes integradas para la inferencia del movimiento tanto de las esporas como de las gotas de agua. Estas se basaron en las arquitecturas RAFT y GMA; RAFT dado que para ambos casos de las redes individuales se obtuvo el mejor resultado con esta red, y GMA que es una iteración nueva sobre esta misma arquitectura. La red RAFT alcanza un AEE mínimo de 0.2172, mientras que la GMA alcanza un mínimo de 0.2007. Al compararlas con las imágenes del conjunto de datos, la RAFT puede inferir todos los elementos, mientras que la GMA tiene problemas con la inferencia del movimiento de las esporas.

Utilizando la optimización de hiperparámetros para la red RAFT, la red de flujo óptico optimizada alcanzó una AEE de 0.187. Tanto el número de épocas como el tamaño del lote resultaron ser factores que redujeron la AEE, mientras que una mayor tasa de aprendizaje máxima o un porcentaje de inicio condujeron a un aumento de la AEE. En futuras iteraciones deberían considerarse más hiperparámetros, como otros optimizadores o un itinerario de entrenamiento diferente, que podrían conducir a una mejora de la AEE.

Al compararlas con imágenes reales, ambas redes pueden detectar el movimiento de la gota de agua al caer, pero una vez que ésta impacta en la hoja, parte del movimiento se detecta correctamente. La red GMA infradetecta el movimiento, mientras que la RAFT detecta correctamente el movimiento para los movimientos pequeños, para los movimientos grandes tiene ruido. Ambos problemas tienen su origen en la falta de hojas en movimiento en el conjunto de datos, que se deberá abordar en futuros trabajos.

También se entrenaron dos redes integradas para la segmentación de la escena: una con una arquitectura ConvNext y otra con una arquitectura SwinNet. ConvNext alcanzó un mAcc máximo de 0.9094 y un mIoU de 0.9536, mientras que la de SwinNet alcanzó un mAcc máximo de 0.9052 y un 0.9515. Al comparar ambas redes con imágenes del conjunto de datos, todos los elementos se segmentan correctamente, sin problemas aparentes. Al compararlas con imágenes reales, el agua se detecta sin problemas, las hojas se detectan correctamente en su mayor parte, con problemas cuando la hoja más reflectante se detecta incorrectamente como agua. Las esporas no se detectan en absoluto y se identifican como hojas. Estos problemas con la segmentación deben ser tratados con cambios en el conjunto de datos para que se parezcan más a los datos reales, ya que la red no tiene problemas para segmentar los datos sintéticos.

En el caso de las redes de segmentación, la optimización de los hiperparámetros de la red ConvNext condujo a un aumento del mIoU de 0.916. El número de épocas, las iteraciones de calentamiento y la razón de calentamiento tienen una correlación positiva con el mIoU de la red. Mientras que la potencia y el tamaño del lote tienen una relación negativa con este valor. Al igual que en el caso del flujo óptico, en futuras iteraciones deberían explorarse más hiperparámetros, como otros programadores y optimizadores.

## 5.1. Trabajo futuro

Si se cambia la red de segmentación para que pueda considerar varios fotogramas o incluso un historial de los mismos para tener una noción del tiempo, se pueden conseguir mejoras en la segmentación, ya que diferentes características, como la reflexión en una hoja, se comportarán con el tiempo de forma diferente a la reflexión procedente de las esporas o de una gota de agua.

En este momento, hay una red para cada una de las tareas. Cada una de ellas tiene una red troncal independiente. Se puede integrar una red troncal para que la información que se retropropaga durante el entrenamiento de la red en una tarea pueda mejorar la otra tarea y viceversa.

Hay cambios más generales que se pueden hacer para mejorar la calidad de la imagen de los datos sintéticos de manera que se acerque más a la configuración experimental real.

Un mayor aumento de la imagen, incluyendo métodos como *Stable Diffusion*, puede mejorar la resistencia de los datos sintéticos a los cambios al pasar a la configuración experimental.

Se pueden explorar más hiperparámetros para encontrar mejores configuraciones para la red.



# Bibliografía

- [1] A. Appel, «Some techniques for shading machine renderings of solids,» 1968, pág. 37, ISBN: 0-521-21689-3. DOI: [10.1145/1468075.1468082](https://doi.org/10.1145/1468075.1468082).
- [2] S. Baker e I. Matthews, «Lucas-Kanade 20 years on: A unifying framework,» *International Journal of Computer Vision*, vol. 56, n.º 3, págs. 221-255, feb. de 2004, ISSN: 09205691. DOI: [10.1023/B:VISI.0000011205.11775.fd](https://doi.org/10.1023/B:VISI.0000011205.11775.fd). dirección: <http://www.ri.cmu.edu/projects/project>.
- [3] Y. Bengio, P. Simard y P. Frasconi, «Learning Long-Term Dependencies with Gradient Descent is Difficult,» *IEEE Transactions on Neural Networks*, vol. 5, n.º 2, págs. 157-166, 1994, ISSN: 19410093. DOI: [10.1109/72.279181](https://doi.org/10.1109/72.279181).
- [4] A. Boudrot, J. Pico, I. Merle, E. Granados, S. Vílchez, P. Tixier, E. De Melo Virginio Filho, F. Casanoves, A. Tapia, C. Allinne, R. A. Rice y J. Avelino, «Shade effects on the dispersal of airborne *Hemileia vastatrix* uredospores,» *Phytopathology*, vol. 106, n.º 6, págs. 527-580, 2016, ISSN: 0031949X. DOI: [10.1094/PHYTO-02-15-0058-R](https://doi.org/10.1094/PHYTO-02-15-0058-R). dirección: <http://dx.doi.org/10.1094/PHYTO-02-15-0058-R>.
- [5] J.-y. Bouguet, «Pyramidal implementation of the affine lucas kanade feature tracker,» *Intel Corporation, Microprocessor Research Labs*, vol. 1, n.º 2, págs. 1-9, 2001.
- [6] J. H. van Boxel, «Numerical Model for the Fall Speed of Raindrops in a Rainfall Simulator,» *Proceedings of the Workshop on Wind and Water Erosion*, vol. 5, págs. 77-85, 1997. dirección: <http://dare.uva.nl>.
- [7] D. J. Butler, J. Wulff, G. B. Stanley y M. J. Black, «A naturalistic open source movie for optical flow evaluation,» en *Lecture Notes in Computer Science (including sub-series Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 7577 LNCS, 2012, págs. 611-625, ISBN: 9783642337826. DOI: [10.1007/978-3-642-33783-3\\_44](https://doi.org/10.1007/978-3-642-33783-3_44).
- [8] K. Cho, B. van Merriënboer, D. Bahdanau e Y. Bengio, «On the Properties of Neural Machine Translation: Encoder–Decoder Approaches,» págs. 103-111, 2015. DOI: [10.3115/v1/w14-4012](https://doi.org/10.3115/v1/w14-4012). arXiv: [1409.1259](https://arxiv.org/abs/1409.1259).
- [9] J. Chung, C. Gulcehre, K. Cho e Y. Bengio, «Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,» págs. 1-9, 2014. arXiv: [1412.3555](https://arxiv.org/abs/1412.3555). dirección: <http://arxiv.org/abs/1412.3555>.

- [10] B. O. Community, *Blender - a 3D modelling and rendering package*, Blender Foundation, Stichting Blender Foundation, Amsterdam, 2021. dirección: <http://www.blender.org>.
- [11] M. Contributors, *MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark*, <https://github.com/open-mmlab/msegmentation>, 2020.
- [12] J. M. Davis, A. D. Eisner, R. W. Wiener y C. E. Main, «A flow visualization study of spore release using a wind tunnel-mounted laser light sheet,» *Plant Disease*, vol. 81, n.º 9, págs. 1057-1065, 1997, ISSN: 01912917. DOI: [10.1094/PDIS.1997.81.9.1057](https://doi.org/10.1094/PDIS.1997.81.9.1057).
- [13] A. Dosovitskiy, P. Fischery, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. V. D. Smagt, D. Cremers y T. Brox, «FlowNet: Learning optical flow with convolutional networks,» en *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2015 Inter, 2015, págs. 2758-2766, ISBN: 9781467383912. DOI: [10.1109/ICCV.2015.316](https://doi.org/10.1109/ICCV.2015.316). arXiv: [1504.06852](https://arxiv.org/abs/1504.06852). dirección: <https://ieeexplore.ieee.org/abstract/document/7410673>.
- [14] E. Dressaire, L. Yamada, B. Song y M. Roper, «Mushrooms use convectively created airflows to disperse their spores,» *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, n.º 11, págs. 2833-2838, mar. de 2016, ISSN: 10916490. DOI: [10.1073/pnas.1509612113](https://doi.org/10.1073/pnas.1509612113).
- [15] H. Edelsbrunner, *Delaunay Triangulations*, Durham, NC, 2008. dirección: <https://courses.cs.duke.edu/fall08/cps230/Lectures/L-21.pdf>.
- [16] J. García Brenes, «Evaluación del desempeño para las estimaciones de flujo óptico sin referencias,» Proyecto de Graduacion, Tecnológico de Costa Rica, 2021.
- [17] I. Goodfellow, Y. Bengio y A. Courville, *Deep Learning*. MIT Press, 2016, <http://www.deeplearningbook.org>, ISBN: 978-0262035613.
- [18] H. Gouraud, «Computer Display of Curved Surfaces,» Tesis doct., University of Utah, 1971. DOI: [10.1007/978-3-662-28615-9\\_3](https://doi.org/10.1007/978-3-662-28615-9_3).
- [19] *Historia del Café de Costa Rica*, 2015. dirección: <http://www.icafe.cr/nuestro-cafe/historia/>.
- [20] S. Hochreiter y J. Uergen Schmidhuber, «Long Shortterm Memory,» *Neural Computation*, vol. 9, n.º 8, pág. 1735-1780, 1997. dirección: [http://www7.informatik.tu-muenchen.de/~%5Csim\\$hochreit%0Ahttp://www.idsia.ch/~%5Csim\\$juergen](http://www7.informatik.tu-muenchen.de/~%5Csim$hochreit%0Ahttp://www.idsia.ch/~%5Csim$juergen).
- [21] B. K. Horn y B. G. Schunck, «Determining optical flow.,» *Computer vision*, págs. 185-203, 1981. DOI: [10.7551/mitpress/1413.003.0014](https://doi.org/10.7551/mitpress/1413.003.0014).
- [22] M. E. Hronek Rojas, «Estimación del movimiento de esporas de la roya del cafeto utilizando aprendizaje profundo,» Proyecto de Graduacion, Tecnológico de Costa Rica, 2021.

- [23] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy y T. Brox, «FlowNet 2.0: Evolution of optical flow estimation with deep networks,» en *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, 2017, págs. 1647-1655, ISBN: 9781538604571. DOI: [10.1109/CVPR.2017.179](https://doi.org/10.1109/CVPR.2017.179). arXiv: [1612.01925](https://arxiv.org/abs/1612.01925). dirección: <http://lmb..>
- [24] I. Ishii, T. Taniguchi, K. Yamamoto y T. Takaki, «High-frame-rate optical flow system,» *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, n.º 1, págs. 105-112, 2012, ISSN: 10518215. DOI: [10.1109/TCSVT.2011.2158340](https://doi.org/10.1109/TCSVT.2011.2158340).
- [25] B. Jähne, *Digital image processing: concepts*. Springer-Verlag Berlin Heidelberg, 1997, pág. 557, ISBN: 978-3-662-21817-4.
- [26] J. Janai, F. Güney, J. Wulff, M. Black y A. Geiger, «Slow flow: Exploiting high-speed cameras for accurate and diverse optical flow reference data,» en *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, 2017, págs. 1406-1416, ISBN: 9781538604571. DOI: [10.1109/CVPR.2017.154](https://doi.org/10.1109/CVPR.2017.154). dirección: <http://www.cvlibs.net/projects/slow>.
- [27] S. Jiang, D. Campbell, Y. Lu, H. Li y R. Hartley, «Learning to Estimate Hidden Motions with Global Motion Aggregation,» en *Proceedings of the IEEE International Conference on Computer Vision*, 2021, págs. 9752-9761, ISBN: 9781665428125. DOI: [10.1109/ICCV48922.2021.00963](https://doi.org/10.1109/ICCV48922.2021.00963). arXiv: [2104.02409](https://arxiv.org/abs/2104.02409). dirección: <https://github.com/zacjiang/GMA..>
- [28] F. Kainz, R. Bogart y P. Stanczyk, «Technical introduction to OpenEXR,» *Industrial light and magic*, pág. 21, 2009.
- [29] D. Kartofelev, *Lecture No10: Attractor and strange attractor, chaos, analysis of Lorenz attractor, Lyapunov exponents, predictability horizon, examples of chaos*, mar. de 2020. dirección: <http://dx.doi.org/10.1175/1520-0469>.
- [30] D. P. Kingma y J. L. Ba, «Adam: A method for stochastic optimization,» *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, págs. 1-15, 2015. arXiv: [1412.6980](https://arxiv.org/abs/1412.6980).
- [31] M. S. Kutchbach, «Los productores directos en el siglo del café,» *Revista de Historia*, n.º 7, págs. 123-217, 1978.
- [32] D. T. Lee y B. J. Schachter, «Two algorithms for constructing a Delaunay triangulation,» *International Journal of Computer & Information Sciences*, vol. 9, n.º 3, págs. 219-242, 1980, ISSN: 00917036. DOI: [10.1007/BF00977785](https://doi.org/10.1007/BF00977785).
- [33] J. Leskovec, A. Rajaraman, J. D. Ullman y u. u. undefined undefined, «Finding Similar Items,» en *Mining of massive datasets*. Cambridge University Press, 2016, págs. 74-74.
- [34] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin y B. Guo, «Swin Transformer: Hierarchical Vision Transformer using Shifted Windows,» 2022, págs. 9992-10 002, ISBN: 9781665428125. DOI: [10.1109/iccv48922.2021.00986](https://doi.org/10.1109/iccv48922.2021.00986). arXiv: [2103.14030](https://arxiv.org/abs/2103.14030). dirección: <https://github..>

- [35] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell y S. Xie, «A ConvNet for the 2020s,» 2022. arXiv: [2201.03545](https://arxiv.org/abs/2201.03545). dirección: <https://github.com/facebookresearch/ConvNeXt%20http://arxiv.org/abs/2201.03545>.
- [36] B. D. Lucas y T. Kanade, «ITERATIVE IMAGE REGISTRATION TECHNIQUE WITH AN APPLICATION TO STEREO VISION.,» en *Proceedings of the 7th international joint conference on Artificial intelligence.*, vol. 2, 1981, págs. 674-679.
- [37] L. M. Madrigal, «Producción de café en Costa Rica creció 12%,» *Delfino*, mar. de 2020. dirección: <https://delfino.cr/2020/03/produccion-de-cafe-en-costa-rica-crecio-12>.
- [38] S. McCook, *Coffee is Not Forever: A Global History of the Coffee Leaf Rust*, ép. Ecology and History Series. Ohio University Press, 2019, ISBN: 9780821423868. dirección: <https://books.google.co.cr/books?id=gISTwgEACAAJ>.
- [39] K. Museth, P. Cucka, M. Aldén y D. Hill, *Welcome*, jun. de 2022. dirección: <https://www.openvdb.org/>.
- [40] B. T. Phong, «Illumination for Computer Generated Pictures,» *Communications of the ACM*, vol. 18, n.º 6, págs. 311-317, 1975, ISSN: 15577317. DOI: [10.1145/360825.360839](https://doi.org/10.1145/360825.360839).
- [41] F. F. Quesada Alfaro, «Predicción del esparcimiento de esporas de roya en plantas de café mediante algoritmos de aprendizaje automático,» Proyecto de Graduación, Tecnológico de Costa Rica, 2022.
- [42] A. Ranjan y M. J. Black, «Optical flow estimation using a spatial pyramid network,» en *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, Institute of Electrical y Electronics Engineers Inc., nov. de 2017, págs. 2720-2729, ISBN: 9781538604571. DOI: [10.1109/CVPR.2017.291](https://doi.org/10.1109/CVPR.2017.291). arXiv: [1611.00850](https://arxiv.org/abs/1611.00850). dirección: <http://arxiv.org/abs/1611.00850>.
- [43] Z. Ren, J. Yan, B. Ni, B. Liu, X. Yang y H. Zha, «Unsupervised deep learning for optical flow estimation,» en *31st AAAI Conference on Artificial Intelligence, AAAI 2017*, 2017, págs. 1495-1501.
- [44] M. Roper, A. Seminara, M. M. Bandi, A. Cobb, H. R. Dillard y A. Pringle, «Dispersal of fungal spores on a cooperatively generated wind,» *Proceedings of the National Academy of Sciences of the United States of America*, vol. 107, n.º 41, págs. 17474-17479, 2010, ISSN: 00278424. DOI: [10.1073/pnas.1003577107](https://doi.org/10.1073/pnas.1003577107). dirección: [www.pnas.org/cgi/doi/10.1073/pnas.1003577107](http://www.pnas.org/cgi/doi/10.1073/pnas.1003577107).
- [45] D. Shreiner y B. Group, *OpenGL Programming Guide: The Official Guide to Learning OpenGL, Versions 3.0 and 3.1*. Pearson Education, 2009, ISBN: 9780321669278. dirección: <https://books.google.co.cr/books?id=xPu3mN2FP14C>.
- [46] H. F. Silverman, «A Class of Algorithms for Fast Digital Image Registration,» *IEEE Transactions on Computers*, vol. C-21, n.º 2, págs. 179-186, 1972, ISSN: 00189340. DOI: [10.1109/TC.1972.5008923](https://doi.org/10.1109/TC.1972.5008923).

- [47] C. H. Skiadas y C. Skiadas, *Handbook of applications of chaos theory*. CRC Press, 2017, págs. 1-928, ISBN: 9781466590441. DOI: [10.1201/b20232](https://doi.org/10.1201/b20232).
- [48] L. N. Smith y N. Topin, «Super-Convergence: Very Fast Training of Residual Networks Using Large Learning Rates,» *CoRR*, vol. abs/1708.07120, 2017. arXiv: [1708.07120](https://arxiv.org/abs/1708.07120). dirección: <http://arxiv.org/abs/1708.07120>.
- [49] J. Snoek, H. Larochelle y R. P. Adams, «Practical Bayesian Optimization of Machine Learning Algorithms,» en *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou y K. Weinberger, eds., vol. 25, Curran Associates, Inc., 2012. dirección: <https://proceedings.neurips.cc/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf>.
- [50] S. M. Solano Acuña, «Implementación de un algoritmo de rastreo de gotas de agua que fomentan la dispersión de esporas de Roya,» Proyecto de Graduacion, Tecnológico de Costa Rica, 2021.
- [51] A. S. R. Srinivasa Rao y C. R. Rao, *Integrated population biology and modeling, Part B*. North Holland, 2019, pág. 635, ISBN: 0444641521.
- [52] D. Sun, X. Yang, M. Y. Liu y J. Kautz, «PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume,» en *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, dic. de 2018, págs. 8934-8943, ISBN: 9781538664209. DOI: [10.1109/CVPR.2018.00931](https://doi.org/10.1109/CVPR.2018.00931). arXiv: [1709.02371](https://arxiv.org/abs/1709.02371).
- [53] S. Sundberg, «Size matters for violent discharge height and settling speed of Sp-hagnum spores: Important attributes for dispersal potential,» *Annals of Botany*, vol. 105, n.º 2, págs. 291-300, feb. de 2010, ISSN: 03057364. DOI: [10.1093/aob/mcp288](https://doi.org/10.1093/aob/mcp288).
- [54] P. Talhinhos, D. Batista, I. Diniz, A. Vieira, D. N. Silva, A. Loureiro, S. Tavares, A. P. Pereira, H. G. Azinheira, L. Guerra-Guimarães, V. Várzea y M. d. C. Silva, «The coffee leaf rust pathogen *Hemileia vastatrix*: one and a half centuries around the tropics,» *Molecular Plant Pathology*, vol. 18, n.º 8, págs. 1039-1051, 2017, ISSN: 13643703. DOI: [10.1111/mpp.12512](https://doi.org/10.1111/mpp.12512).
- [55] Z. Teed y J. Deng, «RAFT: Recurrent All-Pairs Field Transforms for Optical Flow,» en *Computer Vision – ECCV 2020*, A. Vedaldi, H. Bischof, T. Brox y J.-M. Frahm, eds., Cham: Springer International Publishing, 2020, págs. 402-419, ISBN: 978-3-030-58536-5.
- [56] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser e I. Polosukhin, «Attention is All you Need,» en *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan y R. Garnett, eds., vol. 30, Curran Associates, Inc., 2017. dirección: <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.

- 
- [57] E. d. M. Virginio Filho y C. Astorga Domian, «Prevención y control de la roya del café: manual de buenas prácticas para técnicos y facilitadores,» CATIE, Turrialba (Costa Rica), inf. téc., 2015.
- [58] Y. Wang, Y. Yang, Z. Yang, L. Zhao, P. Wang y W. Xu, «Occlusion Aware Unsupervised Learning of Optical Flow,» en *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, págs. 4884-4893, ISBN: 9781538664209. DOI: [10.1109/CVPR.2018.00513](https://doi.org/10.1109/CVPR.2018.00513). arXiv: [1711.05890](https://arxiv.org/abs/1711.05890).
- [59] B. A. Weinberg y B. K. Bealer, «The world of caffeine: the science and culture of the world's most popular drug,» *Choice Reviews Online*, vol. 39, n.º 01, págs. 39–0354–39-0354, 2001, ISSN: 0009-4978. DOI: [10.5860/choice.39-0354](https://doi.org/10.5860/choice.39-0354).
- [60] H. Zhao, J. Shi, X. Qi, X. Wang y J. Jia, «Pyramid Scene Parsing Network,» *CoRR*, vol. abs/1612.01105, 2016. arXiv: [1612.01105](https://arxiv.org/abs/1612.01105). dirección: <http://arxiv.org/abs/1612.01105>.