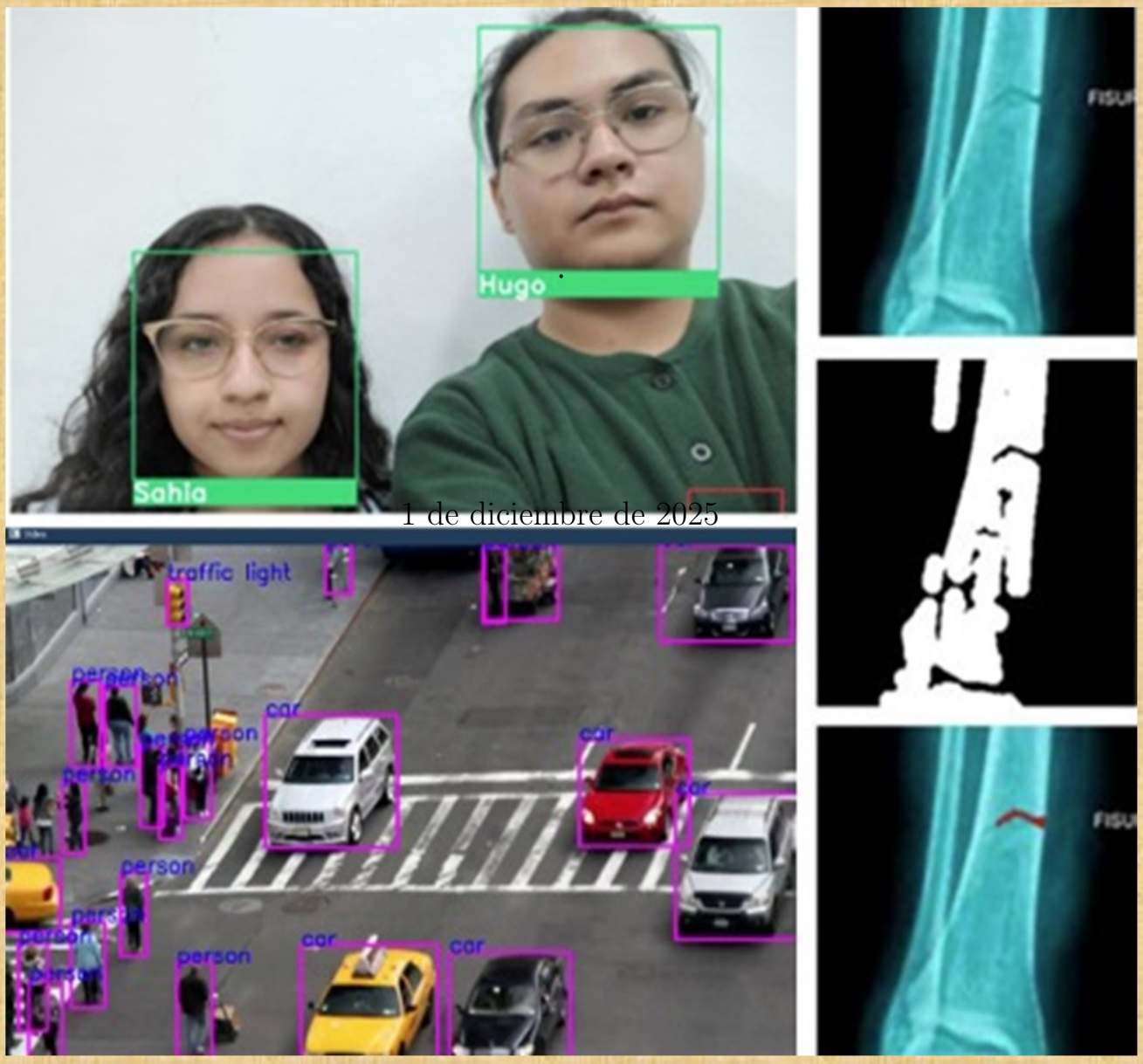
Apuntes en Vision por computadora



1 de diciembre de 2025

UNIVERSIDAD MODELO

Resumen

Este texto presenta una visión general de los conceptos fundamentales de la visión por computadora. Se aborda la definición y aplicaciones, los fundamentos del procesamiento de imágenes, los entornos de desarrollo y bibliotecas más utilizados, técnicas de preprocesamiento (filtrado, ajuste de contraste, supresión de ruido), así como la segmentación y extracción de características. Se concluye con una introducción a las transformaciones geométricas y afines, y con una bibliografía seleccionada para profundizar.

Índice general

Capítulo 1

Introducción

La visión por computadora es la disciplina que dota a las máquinas de la capacidad de interpretar imágenes y video para extraer información útil del entorno. Sus aplicaciones abarcan desde la robótica, inspección industrial y automoción, hasta la medicina y agricultura de precisión. Esta monografía sintetiza los fundamentos técnicos y las herramientas prácticas necesarias para un curso introductorio o material de apoyo didáctico.

1.1. Definición y aplicaciones de la visión por computadora

1.1.1. Definición

La visión por computadora (Computer Vision) estudia algoritmos que permiten a un sistema computacional procesar, analizar y entender imágenes digitales. El objetivo puede variar: detección (¿hay un objeto?), localización (¿dónde está?), reconocimiento (¿qué es?) o reconstrucción (¿cómo es la escena 3D?).

1.1.2. Aplicaciones relevantes

- **Robótica:** navegación, SLAM, manipulación con visión.
- **Industria:** inspección de calidad, conteo, clasificación.
- **Automoción:** detección de peatones, control de carril, conducción autónoma.
- **Medicina:** segmentación de imágenes médicas, ayuda al diagnóstico.
- **Agricultura:** detección de enfermedades, estimación de rendimiento.
- **Seguridad y vigilancia:** reconocimiento de actividad, detección de intrusos.

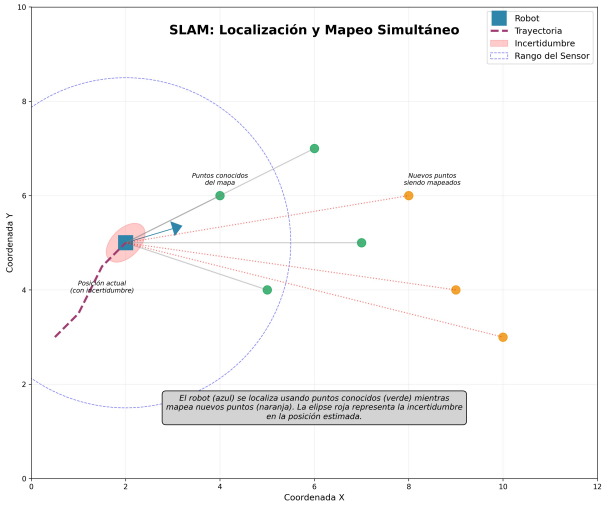


Figura 1.1: SLAM (del inglés simultaneous localization and mapping), es una técnica usada por robots y vehículos autónomos para construir un mapa de un entorno desconocido en el que se encuentra, a la vez que estima su trayectoria al desplazarse dentro de este entorno.

1.2. Fundamentos de procesamiento de imágenes

1.2.1. Representación de la imagen

Una imagen digital en escala de grises puede representarse como una función $I(x, y)$ muestreada en una cuadrícula discreta: $I : \mathbb{Z}^2 \to \{0, \ldots, 255\}$ para 8 bits por píxel. En color, se utilizan espacios como RGB, HSV o CIELab, que permiten manipulaciones distintas (p. ej. separar intensidad de crominancia).

1.2.2. Muestreo y cuantización

El muestreo espacial determina la resolución; la cuantización determina el número de niveles de intensidad. Ambos procesos introducen errores (aliasing, cuantilización) que deben considerarse en el diseño de algoritmos.

1.2.3. Operaciones elementales

- **Histogramas:** distribución de intensidades. La ecualización de histograma mejora el contraste.
- **Operaciones puntuales:** umbralización, corrección gamma.
- **Operaciones locales:** convolución con máscaras (filtrado lineal y no lineal).

1.2.4. Convolución

El filtrado lineal se expresa como convolución discreta:

$$(I * K)(x, y) = \sum_u \sum_v I(x - u, y - v)K(u, v),$$

donde K es el kernel (p. ej. Gaussian blur, filtro de media).

1.3. Entorno de desarrollo y bibliotecas

1.3.1. Lenguajes y entornos comunes

- **Python:** popular por su ecosistema (OpenCV, scikit-image, NumPy, SciPy, PyTorch, TensorFlow).
- **C++:** rendimiento, integración nativa con OpenCV y bibliotecas de visión industrial.
- **MATLAB:** muy usado en docencia e investigación por su toolbox de visión y prototipado rápido.

1.3.2. Bibliotecas y herramientas

OpenCV: biblioteca abierta con cientos de algoritmos para procesamiento, visión y machine learning.

scikit-image: conjunto de algoritmos algorítmicos en Python, fácil de integrar con NumPy.

PyTorch / TensorFlow: para modelos de aprendizaje profundo (detección, segmentación semántica).

MATLAB Computer Vision Toolbox: funciones de calibración, detección y seguimiento.

1.3.3. Ejemplo mínimo (Python + OpenCV)

```
import cv2
img = cv2.imread('imagen.jpg', cv2.IMREAD_COLOR)
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
blur = cv2.GaussianBlur(gray, (5,5), 1.0)
cv2.imwrite('preprocesada.jpg', blur)
```

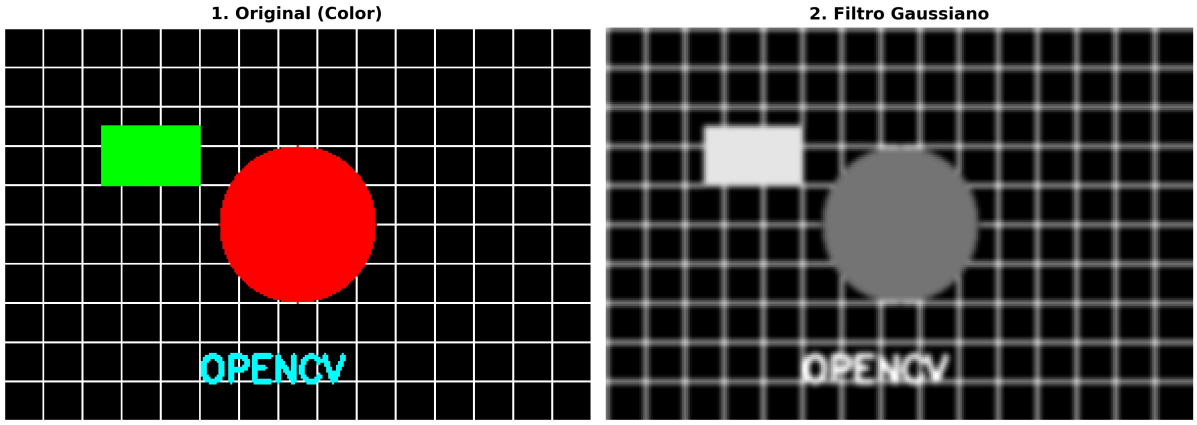


Figura 1.2: filtro blur

1.4. Preprocesamiento de imágenes: filtrado, ajuste de contraste, etc.

1.4.1. Filtrado y eliminación de ruido

- **Filtro de media:** reduce ruido de alta frecuencia pero desenfoca bordes.
- **Filtro gaussiano:** suavizado con preservación relativa de la estructura.
- **Filtro mediano:** eficaz contra ruido impulsivo (salt-and-pepper).
- **Filtro bilateral:** suaviza manteniendo bordes (no lineal).

1.4.2. Ajuste de contraste

- **Ecualización de histograma:** redistribuye niveles para extender el contraste.
- **CLAHE (Contrast Limited AHE):** evita sobre-amplificación del ruido en AHE.
- **Corrección gamma:** modifica la relación entre valores de píxel y luminancia percibida.

1.4.3. Normalización y reducción de iluminación

Para muchas tareas de visión es recomendable normalizar la iluminación (p. ej. mediante división por una estimación del fondo) o usar espacios de color donde la intensidad esté separada de la crominancia (HSV, YCrCb).

1.5. Segmentación de imágenes y extracción de características

1.5.1. Segmentación

La segmentación divide la imagen en regiones coherentes. Técnicas comunes:

- **Umbralización simple** y Otsu: para separar fondo/primer plano.
- **Segmentación basada en color:** filtros en espacio HSV/YCbCr.
- **Segmentación basada en bordes:** Canny + seguimiento de contornos.
- **Segmentación por regiones:** crecimientos de región, watershed.
- **Segmentación por clustering:** K-means sobre vectores de características (color + textura).
- **Segmentación profunda:** U-Net, Mask R-CNN, DeepLab (requieren datasets y entrenamiento).

1.5.2. Extracción de características

Características locales y globales permiten describir regiones u objetos.

- **Descriptores basados en intensidad:** histograma, HOG.
- **Descriptores de puntos clave:** SIFT, SURF (patentadas en el pasado), ORB (open-source).
- **Descriptores de contorno:** Fourier descriptors, Hu moments.
- **Matching de características:** FLANN, BFMatcher.

1.5.3. Ejemplo: detector ORB + matching en OpenCV (Python)

```
import cv2
img1 = cv2.imread('imgA.jpg',0)
img2 = cv2.imread('imgB.jpg',0)
orb = cv2.ORB_create(500)
k1, d1 = orb.detectAndCompute(img1, None)
k2, d2 = orb.detectAndCompute(img2, None)
bf = cv2.BFMatcher(cv2.NORM_HAMMING, crossCheck=True)
matches = bf.match(d1, d2)
matches = sorted(matches, key=lambda x: x.distance)
imgm = cv2.drawMatches(img1,k1,img2,k2,matches[:30], None,
   flags=2)
```

```
cv2.imwrite('matches.jpg', imgm)
```

1.6. Transformaciones geométricas

1.6.1. Transformaciones básicas (2D)

Sea un punto $\mathbf{p} = (x, y)^\top$. Las transformaciones elementales son:

- **Traslación:** $\mathbf{p}' = \mathbf{p} + \mathbf{t}$.
- **Escalado:** $\mathbf{p}' = S\mathbf{p}$, con $S = \mathrm{diag}(s_x, s_y)$.
- **Rotación:** $\mathbf{p}' = R\mathbf{p}$, con

$$R(\theta) = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix}.$$

Combinando estas operaciones se crean transformaciones más complejas. El uso de coordenadas homogéneas permite representar traslación como multiplicación matricial:

$$\begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} = \begin{pmatrix} R & \mathbf{t} \\ \mathbf{0}^\top & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}.$$

1.6.2. Warpping e interpolación

Al aplicar transformaciones continuas en imágenes discretas, es necesario interpolar (nearest, bilinear, bicubic) y gestionar coordenadas fuera de la imagen (border modes).

1.7. Transformaciones afines

1.7.1. Definición

Una transformación afín en 2D tiene forma:

$$\mathbf{p}' = A\mathbf{p} + \mathbf{b},$$

donde $A \in \mathbb{R}^{2\times 2}$ y $\mathbf{b} \in \mathbb{R}^2$. Conserva paralelismo y relaciones lineales, pero no necesariamente distancias ni ángulos.

1.7.2. Homografías y transformaciones proyectivas

Cuando hay perspectiva (p. ej. cambiamos el punto de vista), la transformación general entre planos es una homografía $H \in \mathbb{R}^{3\times 3}$ tal que, en coordenadas homogéneas,

$$\tilde{\mathbf{p}}' \sim H\tilde{\mathbf{p}}.$$

La homografía requiere al menos 4 correspondencias no colineales para estimarse.

1.7.3. Estimación y aplicación

- **Estimación por correspondencias:** usando RANSAC para robustez frente a outliers.
- **Aplicación:** mosaicing (stitching), rectificación de imágenes, corrección de la perspectiva.

1.8. Conclusiones

Los módulos presentados constituyen la base para desarrollar cursos prácticos y proyectos en visión por computadora. La combinación de teoría (modelos matemáticos), algoritmos clásicos (filtrado, detección de bordes, segmentación) y herramientas modernas (OpenCV, frameworks de Deep Learning) permite enfrentar problemas reales en robótica, industria y ciencia.

Notas

Se recomienda a los lectores consultar las referencias para profundizar en los temas y utilizar los tutoriales prácticos sugeridos para consolidar habilidades de programación.

Bibliografía

[1] R. C. Gonzalez and R. E. Woods. *Digital Image Processing*, 3rd edition. Pearson, 2008.

[2] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, 2010.

[3] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 2008.

[4] OpenCV Development Team. OpenCV documentation. `https://docs.opencv.org/` (consultado en línea).

[5] MathWorks. Computer Vision Toolbox Documentation. `https://www.mathworks.com/help/vision/` (consultado en línea).

[6] *Visión por Computador* — libro en línea (capítulos). Disponible en: `https://www.visionporcomputador.es/libroVision/VisionPorComputador.pdf`

[7] Ibarra (compilador). *Visión Artificial y Procesamiento Digital de Imágenes con MATLAB*. Disponible en línea (manual/compendio).

Capítulo 2

Tema II: Detección y Seguimiento de Objetos

Resumen

La detección y el seguimiento de objetos constituyen dos pilares esenciales de la visión por computadora moderna. Su desarrollo ha sido impulsado tanto por avances matemáticos en el análisis de imágenes como por el surgimiento de redes neuronales profundas capaces de procesar información visual con niveles de precisión similares o superiores al ser humano. Esta monografía presenta una revisión histórica, fundamentos matemáticos relevantes y un análisis detallado de los algoritmos contemporáneos utilizados para detectar y seguir objetos en imágenes y video.

2.1. Introducción

Desde sus orígenes en la década de 1960, la visión por computadora ha tenido como objetivo dotar a las máquinas de la capacidad de interpretar el mundo visual. En sus primeras etapas, el análisis de imágenes se basaba principalmente en operaciones matemáticas relativamente simples: gradientes, transformadas y filtros lineales. Con el tiempo, el campo evolucionó hacia técnicas más sofisticadas como modelos probabilísticos, geometría epipolar y sistemas basados en aprendizaje automático.

Hoy en día, la detección y el seguimiento de objetos son fundamentales para aplicaciones como vehículos autónomos, drones inteligentes, sistemas biomédicos, inspección industrial y seguridad. Para comprender plenamente estos avances, es necesario revisar el recorrido histórico y matemático que dio origen a los métodos actuales.

2.2. Fundamentos históricos de la detección de objetos

Los primeros enfoques para la localización de objetos se basaban en técnicas como:

- la transformada de Hough (1962),

- el análisis de bordes mediante gradientes (Sobel, Prewitt),
- modelos basados en plantillas rígidas.

Aunque limitados, estos métodos establecieron los cimientos para trabajos posteriores. En los años 1990 surgieron modelos más avanzados como:

- las características SIFT (Scale-Invariant Feature Transform) de David Lowe (1999),
- los descriptores HOG (Histogram of Oriented Gradients) de Dalal y Triggs (2005),
- el detector Viola–Jones (2001), uno de los primeros en funcionar en tiempo real.

La revolución definitiva llegó con las redes neuronales profundas en 2012, tras el éxito de AlexNet. Desde entonces, arquitecturas como YOLO, SSD y Faster R-CNN dominan el estado del arte.

2.3. Fundamentos matemáticos de la detección

La detección moderna se basa en regresión multivariable y clasificación mediante funciones de pérdida del tipo:

$$L = L_{cls} + \lambda L_{bbox}$$

donde:

- L_{cls} es la pérdida de clasificación (normalmente entropía cruzada),
- L_{bbox} mide la precisión de la predicción del cuadro delimitador.

La regresión de los bounding boxes se realiza usando mínimos cuadrados o variantes robustas:

$$L_{bbox} = \sum_i \left\| (x_i, y_i, w_i, h_i) - (\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i) \right\|^2$$

En modelos modernos, se utilizan parámetros ancla y métricas como el IoU (Intersection over Union):

$$IoU = \frac{Area(B_{pred} \cap B_{gt})}{Area(B_{pred} \cup B_{gt})}$$

2.3.1. YOLO

YOLO convierte toda la imagen en un solo problema de regresión. Divide la imagen en celdas y predice múltiple información por celda. Su diseño lo hace extremadamente rápido y adecuado para sistemas en tiempo real.

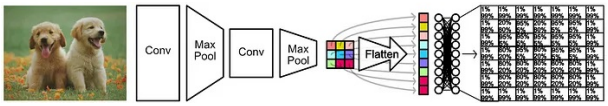


Figura 2.1: Diagrama conceptual Yolo

2.3.2. SSD

SSD utiliza múltiples mapas de características a diferentes escalas, permitiendo detectar objetos pequeños y grandes de forma simultánea.

2.3.3. Faster R-CNN

En contraste con YOLO y SSD, Faster R-CNN emplea una red adicional llamada *Region Proposal Network* (RPN) que propone regiones candidatas antes de realizar la clasificación final. Este enfoque es más preciso, pero menos rápido.

2.4. Clasificación y etiquetado de objetos

Una vez identificada la ubicación de un objeto, el sistema debe asignarle una categoría semántica. La clasificación moderna se basa en redes convolucionales profundas, cuyos fundamentos matemáticos derivan de convoluciones discretas:

$$(f * g)(x, y) = \sum_{u} \sum_{v} f(u, v)\, g(x - u, y - v)$$

El aprendizaje se realiza mediante descenso de gradiente sobre millones de parámetros entrenables.

2.5. Seguimiento de objetos: historia y fundamentos

El seguimiento de objetos tiene raíces profundas en el análisis de movimiento. En los años 1980, Horn y Schunck propusieron el flujo óptico basado en variaciones continuas de intensidad:

$$I_x u + I_y v + I_t = 0$$

Este principio matemático dio origen a algoritmos más robustos como Lucas–Kanade (KLT), que asume pequeñas variaciones y soluciona un sistema lineal local:

$$A\mathbf{v} = \mathbf{b}$$

donde:

$$A = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix}, \quad \mathbf{b} = -\begin{bmatrix} I_x I_t \\ I_y I_t \end{bmatrix}$$

2.6. Algoritmos de seguimiento

2.6.1. KLT (Lucas–Kanade Tracker)

KLT selecciona puntos de interés (como esquinas de Harris) y los rastrea a través del tiempo minimizando la diferencia entre ventanas locales.

Figura 2.2: Visualización del rastreo de puntos mediante KLT.

2.6.2. Mean-Shift

Mean-Shift busca maximizar la densidad de probabilidad de características (normalmente color) desplazando iterativamente una ventana hacia la región de mayor densidad.

Su fundamento matemático se basa en estimación de densidades:

$$m(x) = \frac{\sum_i x_i K(x - x_i)}{\sum_i K(x - x_i)}$$

CamShift extiende Mean-Shift permitiendo adaptar dinámicamente el tamaño de la ventana.

2.7. Aplicaciones modernas

Los algoritmos descritos se encuentran en prácticamente todos los sistemas inteligentes de última generación:

- Vehículos autónomos: detección de peatones, semáforos y otros vehículos.
- Robótica: navegación, SLAM y manipulación de objetos.
- Industria 4.0: inspección automática y conteo de producción.
- Medicina: análisis de movimiento, seguimiento de células.

2.8. Conclusión

La detección y el seguimiento de objetos representan uno de los logros más importantes de la visión por computadora. Sus raíces matemáticas son profundas y abarcan desde análisis de gradientes hasta regresiones sofisticadas sobre redes neuronales profundas. La combinación de historia, teoría y computación moderna ha permitido alcanzar niveles de precisión y eficiencia nunca vistos, abriendo paso a nuevas aplicaciones industriales, científicas y sociales.

Bibliografía

1. Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). “You Only Look Once: Unified, Real-Time Object Detection.” CVPR.
2. Ren, S., He, K., Girshick, R., & Sun, J. (2015). “Faster R-CNN.” NIPS.
3. Liu, W., et al. (2016). “SSD: Single Shot MultiBox Detector.” ECCV.
4. Horn, B. K. P. & Schunck, B. (1981). “Determining Optical Flow.” Artificial Intelligence.
5. Lucas, B. & Kanade, T. (1981). “An Iterative Image Registration Technique.” DARPA.
6. Yilmaz, A., Javed, O., & Shah, M. (2006). “Object Tracking: A Survey.” ACM Computing Surveys.
7. Dalal, N. & Triggs, B. (2005). “Histograms of Oriented Gradients.” CVPR.

Capítulo 3

Aplicaciones avanzadas

3.1. Introducción

La visión por computadora es un campo interdisciplinario que combina matemáticas, física, biología, estadística, ciencias de la computación e ingeniería para permitir que las máquinas interpreten el contenido visual del mundo, tal como lo hace el ser humano. En sus orígenes, la visión artificial se inspiró en estudios neurobiológicos relacionados con la percepción humana. Posteriormente, su evolución fue marcada por el desarrollo de la transformada de Fourier, el análisis de señales, los algoritmos de optimización y, más recientemente, las redes neuronales profundas.

Históricamente, la visión por computadora se desarrolló en tres grandes etapas:

- **1960–1980**: Algoritmos simbólicos, detección de bordes, geometría básica de cámaras.
- **1980–2000**: Métodos matemáticos robustos, optimización, visión estereoscópica y SLAM inicial.
- **2000–presente**: Aprendizaje automático, CNNs, redes profundas, transformers, visión 3D, visión multimodal.

Hoy en día, su impacto es visible en áreas críticas como diagnóstico médico, cirugía asistida, conducción autónoma, manufactura inteligente y robótica humanoide.

3.2. Aplicaciones avanzadas de la visión por computadora

3.2.1. Aplicaciones en Medicina

La adopción de visión por computadora en medicina ha crecido exponencialmente debido al aumento en la disponibilidad de datos clínicos, el avance en hardware computacional y la capacidad predictiva demostrada por las redes neuronales profundas.

Antecedentes históricos

Los primeros trabajos datan de la década de 1970, cuando se realizaron experimentos para segmentar radiografías usando umbralización y transformadas matemáticas. Con la llegada de técnicas como la *transformada de Hough*, la *transformada wavelet*, y más tarde las *Redes Convolucionales (CNN)*, el procesamiento médico comenzó a alcanzar niveles diagnósticos comparables a expertos humanos.

Fundamentos matemáticos relevantes

Muchas de las técnicas usadas en medicina se basan en:

- **Segmentación**: minimización de energía, métodos variacionales, modelos de nivel-set.
- **Detección**: convoluciones multidimensionales, operadores diferenciales discretos.
- **Reconstrucción 3D**: geometría epipolar, transformadas radiales, métodos iterativos.
- **Clasificación**: funciones de pérdida como cross-entropy, optimización SGD, gradiente descendente.

Aplicaciones destacadas

1. **Detección temprana de cáncer** (mama, piel, pulmón).
2. **Segmentación automática de órganos**: hígado, corazón, cerebro.
3. **Análisis de imágenes de resonancia magnética (MRI)**.
4. **Microscopía asistida por IA**.
5. **Cirugía robótica con visión estereoscópica**.

La combinación de visión por computadora y medicina está transformando la práctica clínica hacia un enfoque más preventivo, preciso y automatizado.

3.2.2. Visión por Computadora en la Industria Automotriz y Robótica

La industria automotriz y la robótica han sido dos de los campos más beneficiados por la visión por computadora debido a su necesidad constante de autonomía, precisión y capacidad para percibir el entorno en tiempo real.

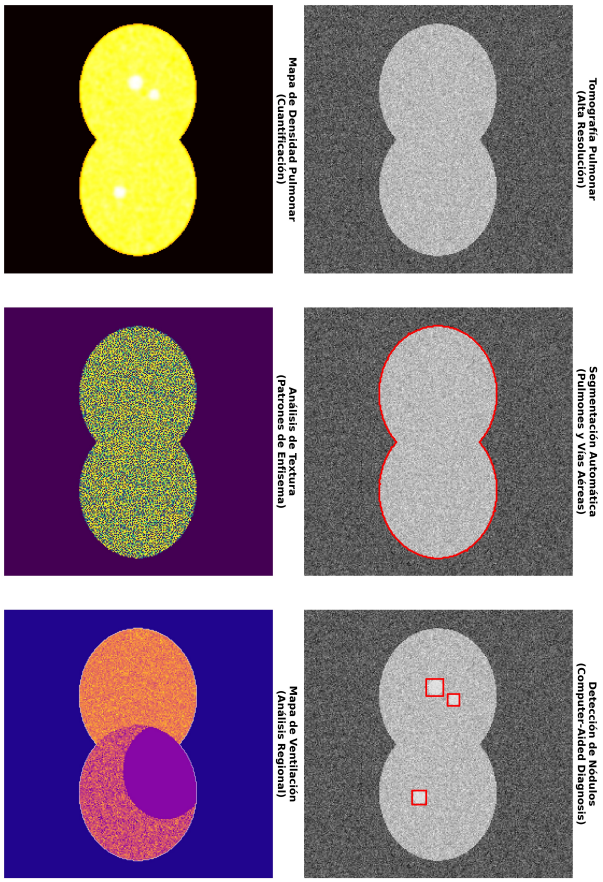


Figura 3.1: Ejemplo de segmentación médica, análisis pulmonar avanzado

Breve historia en automoción

Los primeros prototipos de conducción asistida se desarrollaron en la década de 1980 en el proyecto alemán *PROMETHEUS*. A partir de 2010, con el auge del Deep Learning y sensores avanzados (LiDAR, cámaras estéreo, cámaras infrarrojas), se produjeron avances cruciales en:

- Detección de peatones
- Frenado automático
- Seguimiento de carril
- Percepción 3D del entorno

Robótica: orígenes matemáticos

En robótica, la visión se apoya en:

- **Cinemática y dinámica**: modelos matriciales y ecuaciones de Euler-Lagrange.
- **Estimación de pose**: Cuaterniones, ángulos de Euler, filtros de Kalman.
- **SLAM visual**: optimización no lineal (Levenberg–Marquardt, Bundle Adjustment).

Aplicaciones clave en automoción

1. Reconocimiento de señales de tránsito.
2. Asistencia ALKS (Automated Lane Keeping System).
3. Percepción 360° para conducción autónoma.
4. Detección de objetos en movimiento en tiempo real.

Aplicaciones clave en robótica

- Robots colaborativos (CoBots) con visión 3D.
- Manipulación robótica en ambientes no estructurados.
- Robots humanoides con visión para equilibrio y caminata.
- Navegación autónoma basada en mapas visuales.

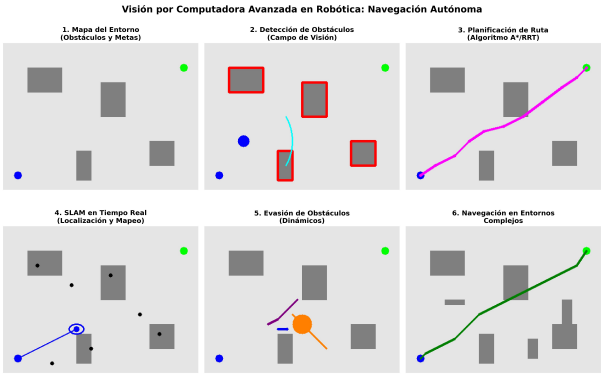


Figura 3.2: Ejemplo de percepción robótica

3.2.3. Desarrollo de Proyectos Prácticos

Esta sección sintetiza cómo estudiantes e ingenieros pueden llevar las ideas anteriores a proyectos reales.

Metodología general para proyectos de visión

1. **Definición del problema**: objetivos, métricas, entorno físico.
2. **Adquisición de datos**: cámaras, sensores, fuentes abiertas.
3. **Preprocesamiento**: normalización, rotación, filtrado, aumento de datos.
4. **Diseño del algoritmo**: tradicional (OpenCV) o Deep Learning (PyTorch/TensorFlow).

5. **Entrenamiento y validación**: métricas como IoU, AP, precisión, sensibilidad.
6. **Implementación en hardware**: Jetson Nano, Raspberry Pi, ESP32-CAM.
7. **Despliegue**: optimización, quantización, pruebas reales.

Ejemplos de proyectos prácticos

- Sistema de conteo de vehículos basado en YOLO.
- Clasificación automática de radiografías (torax, manos).
- Seguimiento de objetos móviles en drones.
- Brazos robóticos con retroalimentación visual (visual servoing).
- Detcción de defectos en procesos industriales.

Orígenes teóricos usados en los proyectos

- Optimización convexa para entrenamiento.
- Teoría de la información y entropía.
- Análisis de señales y convoluciones 2D.
- Geometría proyectiva para visión 3D.

3.3. Conclusiones

La visión por computadora es hoy una de las disciplinas más influyentes en tecnología. Su impacto en medicina, automoción, robótica y la educación está habilitando capacidades que antes se consideraban ciencia ficción. Las aplicaciones avanzadas estudiadas muestran la profunda relación entre modelos matemáticos, avances en inteligencia artificial y el desarrollo de sistemas reales de alto impacto.

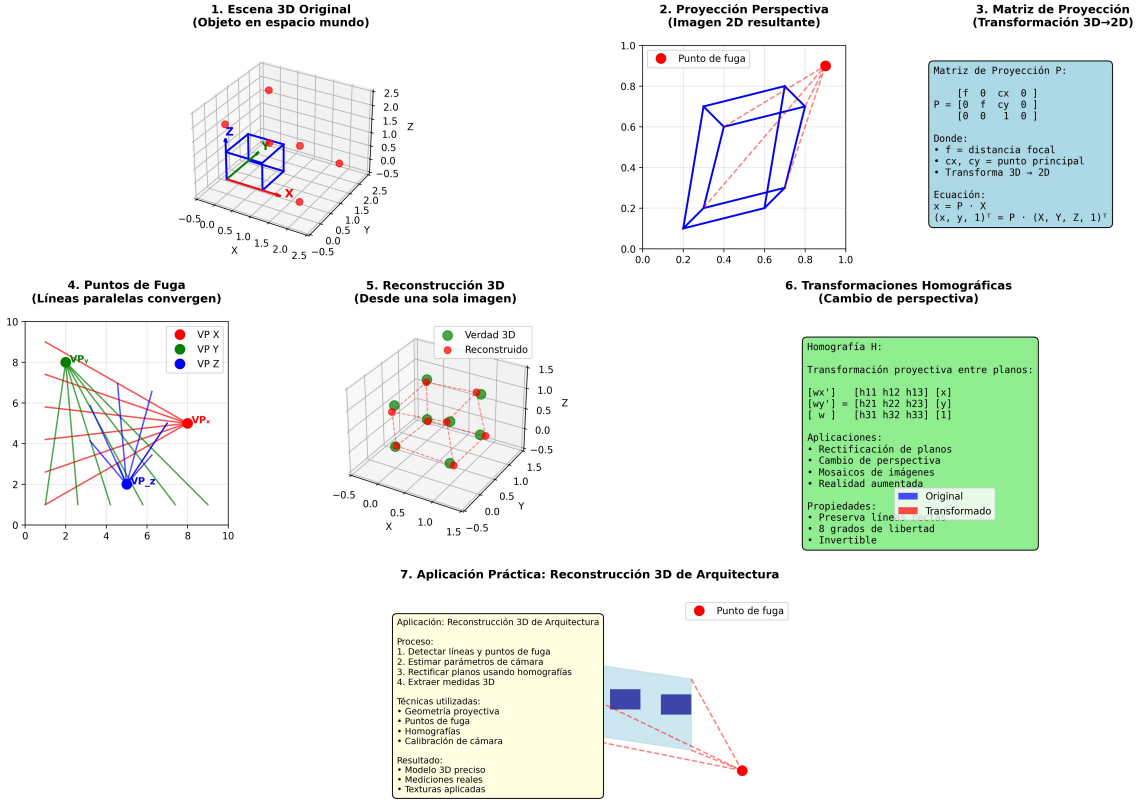


Figura 3.3: Ejemplos Geometria Proyectiva

Bibliografía

[1] R. C. Gonzalez, R. E. Woods, *Digital Image Processing*. Pearson, 2018.

[2] R. Szeliski, *Computer Vision: Algorithms and Applications*. Springer, 2nd Ed., 2022.

[3] I. Goodfellow, Y. Bengio, A. Courville, *Deep Learning*. MIT Press, 2016.

[4] R. Hartley, A. Zisserman, *Multiple View Geometry*. Cambridge University Press, 2004.

[5] D. Forsyth, J. Ponce, *Computer Vision: A Modern Approach*. Pearson, 2011.

[6] D. Shen, G. Wu, H. Suk. "Deep Learning in Medical Image Analysis." *Annual Review of Biomedical Engineering*, 2017.

[7] S. Thrun et al., "Stanley: The Robot that Won the DARPA Grand Challenge." *Journal of Field Robotics*, 2006.