
# Rápida Obtención de Puntos Homólogos para Visión Robótica

Autor: Jorge Kamlofsky

Directora: Dra. María Lorena Bergamini

Co-Director: Dr. José Francisco Zelasco

Tesis presentada para obtener el título de

Magister en Tecnología Informática

(Marzo, 2021)

# Resumen

La visión robótica brinda a los robots la capacidad de reconocer objetos que se encuentran en su entorno. Esto puede lograrse con un par de cámaras que obtienen imágenes de una escena desde dos puntos de vista. Se obtiene un modelo tridimensional, en unidades de longitud mediante una técnica llamada Fotogrametría Digital o Metrología con Imágenes estereoscópicas o símplemente: estereoscopía. Para su cálculo, es necesario conocer las características físicas del sistema de adquisición de las imágenes, las cuales generalmente se obtienen por única vez cuando se calibra el sistema. Es fundamental tener las coordenadas de puntos en ambas imágenes que se corresponden con cada punto de la escena. Este proceso es llamado reconocimiento de puntos homólogos, puesta en correspondencia o apareo y posee muy alta complejidad de cálculo. Sobre este proceso se enfoca este trabajo.

Generalmente, es necesario reconocer automáticamente a los objetos en ambas imágenes. El uso de patrones de borde es un enfoque común. Un problema que surge inmediatamente es que las formas se encuentran en cualquier posición, tamaño y orientación dentro de las imágenes. Se presenta un patrón descriptor de las curvas de borde que resulta invariante frente a rotaciones, traslaciones y escalado, consistente en una curva discreta. Hallar un objeto en una imagen, consiste entonces, en medir la distancia entre el patrón descriptor del objeto a buscar y los patrones de cada objeto de la imagen y comparar su diferencia con un umbral de similitud predefinido. Dicho patrón es dependiente del punto de inicio. Sin embargo, tras la obtención de la orientación de la forma y dado que en estereoscopía, normalmente, ambas imágenes son similares, lo son también los objetos allí contenidos. Luego la identificación del punto de inicio se logra fácilmente.

Con ello, realizar el apareo de varios puntos de las curvas de borde de un objeto puede lograrse con menor dificultad, lo cual permite lograr una representación tridimensional aproximada que puede ser compatible con requerimientos de tiempo real. Se incluyen datos experimentales.



## Palabras clave:

*Visión robótica*, *puntos homólogos*, *visión 3d*, *estereoscopía*, *reconstrucción tridimensional*.

# Abstract

Robotic vision gives robots the ability to recognize objects in their environment. This can be accomplished with a couple of cameras that take images of a scene from two points of view. A three-dimensional model is obtained, in units of length by means of a technique called Digital Photogrammetry or Metrology with stereoscopic images or simply: stereoscopy. For its calculation, it is necessary to know the physical characteristics of the image acquisition system, which are generally obtained only once when the system is calibrated. It is essential to have the coordinates of points in both images that correspond to each point in the scene. This process is called homologous point recognition, matching or pairing and has a high computational complexity. This work is focused on this process.

Generally, it is necessary to automatically recognize the objects in both images. Using border patterns is a common approach. One problem that immediately arises is that shapes are in any position, size, and orientation within images. A descriptive pattern of the edge curves is presented that is invariant against rotations, translations and scaling, consisting of a discrete curve. Finding an object in an image then consists of measuring the distance between the descriptor pattern of the object to be found and the patterns of each object in the image and comparing their difference with a predefined threshold of similarity. This pattern is dependent on the starting point. However, after obtaining the orientation of the shape and given that in stereoscopy, normally, both images are similar, so are the objects contained therein. Then the identification of the starting point is easily accomplished.

Thus, pairing several points of the edge curves of an object can be achieved with less difficulty, which allows to achieve an approximate three-dimensional representation that can be compatible with real-time requirements. Experimental data is included.



## Keywords:

*Robotic vision*, *homologous points*, *3d vision*, *stereoscopy*, *three-dimensional reconstruction*.

# Agradecimientos

Mi especial agradecimiento es para mi querida esposa Lorena (Lolo), quien me soporta en las tareas de desarrollo de las investigaciones, acompañándome durante días feriados, fines de semana, días libres y de vacaciones dedicados a la experimentación, lectura y escritura de trabajos, y también asistiendo a mis charlas en congresos, dándome a diario el ánimo necesario para avanzar.

A mis padres, quienes me han educado como una persona de bien, de esfuerzo y estudio para alcanzar las sucesivas metas.

A mis sobrinos Cindy, Emiliano, Geraldine y la pequeña Lara, quienes ven en mi una fuente de inspiración y me acompañan en la vida y me alegran con su presencia.

A mis amigos de toda la vida: Leo (Fili), Roberto, mi hermanos Dany y Gastón, mi primo Raúl, mi tío Mingo, quienes están siempre presentes, y mis otros primos y parientes, que también los tengo presentes.

Y también a mis mentores: docentes y colegas principalmente de la Universidad Abierta Interamericana (UAI) pero también a mis docentes y mentores de otras universidades como Facultad de Ingeniería del Ejército (FIE), la Universidad Tecnológica Nacional (UTN), Universidad de Buenos Aires (UBA), Universidad Nacional de Lomas de Zamora (UNLZ), y de la Universidad Nacional de La Plata (UNLP) quienes han participado en mi formación de Investigador y Docente. En especial, mi agradecimiento a los Directores de Tesis: Lorena y José por su tiempo y dedicación, no solo en la corrección de este trabajo, sino por su trabajo en los años previos que permitieron ir dando forma a esta Tesis.

A todos ellos, y a cada uno de ellos, mil gracias.

# Índice de Contenidos

# Índice de ilustraciones

# Índice de tablas

# 1. Introducción

## 1.1. Trabajos relacionados

La visión robótica pretende emular a la visión humana de modo de otorgar a los robots la capacidad de interpretar el ambiente en el que se desenvuelven y actuar en consecuencia. Para ello, se equipa a los robots con cámaras que actuando a modo de ojos, obtienen imágenes cuyo procesamiento permite obtener información acerca de su entorno. El abanico de aplicaciones de visión robótica es muy amplio. Gran parte de ellas, las más simples, tienen condiciones acotadas y poco cambiantes. En otras aplicaciones se requiere la identificación de objetos en movimiento a partir de capturas de video. Muchos desarrollos buscan mayor nivel de autonomía. Para ello, el reconocimiento debe obtenerse en ambientes muy cambiantes y en muy poco tiempo. En algunas, incluso, se requiere que esto se logre en tiempo real.

Las cámaras adquieren datos del entorno en forma de imágenes digitales. En sistema de visión robótica se caracteriza según el método que usa para la identificación de objetos en las imágenes. Las estrategias conocidas para reconocer objetos en imágenes se basan en el análisis de los bordes o el estudio integral de la imagen (o de una ventana parcial) donde es probable que el objeto esté (Zhang y Lu, 2004). Las estrategias basadas en el análisis de bordes utilizan técnicas para detectar líneas y bordes mediante técnicas de umbralizado (Davis, 1975; Rosenfeld y Kak, 2014), o mediante el uso de diferentes filtros (Gonzalez y Woods, 1993) para luego extraer las características de la forma utilizando Topología Digital (Eckhardt y Latecki, 1994; Rosenfeld, 1979) o reconociendo patrones obtenidos desde el borde. Las estrategias basadas en el análisis del área completa de la imagen obtienen características del objeto contenido en la imagen mediante técnicas de entrenamiento y reconocimiento de patrones utilizando algoritmos de inteligencia artificial (iniciales: IA) como ser: redes neuronales donde el algoritmo aprende a partir de un conjunto de imágenes, algoritmos genéticos (Rowley et al., 1998) o bien, mediante el uso de la transformada wavelets (Sarria, 2007).

La discretización de los datos de la imagen, la resolución de la cámara, la falta o el exceso de brillo, la ofuscación, la pérdida de claridad o el ruido en la imagen, la variedad de objetos a identificar, son aspectos a tener en cuenta, ya que en muchos casos, esto requiere un tratamiento adicional (Weaver et al., 1991; Rudin et al., 1992; Nguyen y Debled-Rennesson, 2007), lo que compromete el cumplimiento de requerimientos de tiempo real. Por el contrario, en condiciones controladas, sencillas y acotadas, el tratamiento es más sencillo, lo que facilita la identificación de objetos. Entonces, en aplicaciones donde el reconocimiento de objetos debe hacerse en tiempo real, es muy conveniente que las condiciones se simplifiquen de modo de lograr un equilibrio que permita un procesamiento correcto en el tiempo requerido.

La visión robótica requiere convertir la información bidimensional de la escena obtenida de imágenes grabadas por cámaras en un modelo tridimensional para reconocer objetos y lugares, a fin de ejecutar correctamente las tareas asignadas. La estereoscopía es una técnica que imitado a la visión humana, partiendo de dos (o más) imágenes de una escena logran una reconstrucción tridimensional de la misma (Moravec, 1996). El cálculo se realiza luego de haber finalizado el proceso de reconocimiento de puntos homólogos o puesta en correspondencia o apareo en ambas imágenes. Esto es: la obtención de los pares de coordenadas de puntos en ambas imágenes que se corresponden al mismo punto del objeto. Es conveniente, obviamente, que previo a ello los objetos

en cada imagen hayan sido reconocidos. Sin embargo, esta tarea requiere probar la coincidencia de millones de píxeles en millones de combinaciones posibles (Zelasco et al., 2000), lo que compromete los requisitos de reconocimiento en tiempo real.

El reconocimiento de los objetos en una imagen es un tema central en este trabajo. Las técnicas que usan algoritmia de IA (especialmente: Redes neuronales) son hoy ampliamente aceptadas por su velocidad y robustez. En el caso de la visión robótica existen múltiples aplicaciones que reconocen objetos y realizan tareas mediante algoritmia basada en IA. Entre otras se pueden mencionar: (Burgard et al., 1999; Monar et al., 2014; Garcia y Garcia, 2013; Peña Cabrera et al., 2004; Malpartida y Sobrado, 2003; Egmont Petersen et al., 2002; Rautaray y Anupam, 2015). En aplicaciones donde el robot se desempeña en ambientes muy cambiantes, de gran complejidad, o inexplorados, como por ejemplo en robots de rescate para zonas de catástrofe (Greer et al., 2002; Naidoo et al., 2015; Murphy y Stover, 2008; Kamlofsky et al., 2018), es conveniente otro enfoque.

En Kamlofsky et al. (2018) se presentó, un esquema de Visión Robótica con reconocimiento de objetos basado en un enfoque topológico (usando análisis de bordes). En el mismo se identifican tres instancias: calibración de las cámaras, apareo y estereoscopía. En este trabajo se presenta en detalle un simple proceso de reconocimiento de puntos homólogos o apareo, lo cual permitiría poder lograr el modelo tridimensional, que promete ser compatible con requerimientos de tiempo real.

## 1.2. Objetivos de esta tesis

El objetivo principal de este trabajo es presentar un algoritmo que permita lograr en forma rápida el apareo de puntos entre pares de imágenes estéreo de una misma escena, con muy baja complejidad computacional, de modo de aproximarse a exigentes requerimientos de tiempo real.

Los objetivos intermedios son: presentación de algoritmos para la aproximación poligonal de las curvas de borde, presentación del patrón descriptor de la curva y con él, mostrar una forma de hallar objetos en imágenes. Con ello, el proceso de apareo se logra muy fácilmente, sin elevada complejidad de cálculo.

Un objetivo subyacente de este trabajo es mostrar que el enfoque topológico de análisis de bordes para la identificación de objetos puede ser efectivo, más en aplicaciones con ambientes muy cambiantes, y/o con requerimientos de procesamiento en tiempo real.

## 1.3. Relevancia del Tema

Las técnicas de visión robótica tienen un abanico cada vez más amplio de aplicaciones. El eficiente modelado tridimensional de escenas en tiempo real y en ambientes muy cambiantes permitiría su implementación en sistemas autónomos en una gran variedad de ambientes: vehículos de transporte, robots humanoides, drones, etc.

Visión robótica en tiempo real es un tema abierto, de gran relevancia y de vanguardia tecnológica.

## 1.4. Estructura del Trabajo

En el capítulo 2 se detalla la metodología de la investigación utilizada para realizar este trabajo. En el capítulo 3 se presenta el marco teórico, donde se definen conceptos básicos obtenidos a partir de literatura clásica del tema en cuestión. El capítulo 4 expone el desarrollo técnico del enfoque abordado para el proceso de visión robótica, detallando cada una de las etapas. En especial, este desarrollo se enfoca en la obtención de puntos homólogos en pares de imágenes estéreo. En el capítulo 5 se presenta una aplicación experimental conteniendo datos experimentales del proceso propuesto y su análisis. Finaliza con la conclusión y trabajos futuros.

# 2. Metodología de la investigación

## 2.1. Marco de investigación de esta tesis

La presente Tesis se desarrolló a partir de los resultados de las investigaciones iniciadas en el proyecto denominado: *Herramientas de análisis de imágenes digitales para la visión artificial* dirigido por la Dra. María Lorena Bergamini, iniciado en 2012 y radicado en el Caeti[1].

En 2015 estas investigaciones se incorporaron y se continuaron en marco del proyecto binacional llamado "Semi-Autonomous robots for search and rescue operations". Este proyecto fue seleccionado para ser ejecutado en el trienio 2014-2016, en el marco del Programa de Cooperación Científico-Tecnológica entre el Ministerio de Ciencia, Tecnología e Innovación Productiva de la República Argentina (MINCYT) y el Departamento de Ciencia y Tecnología de la República de Sudáfrica (DST), bajo el código SA – 13/13.

El responsable de la parte sudafricana fue el Dr. Glen Bright, del Departamento de Mecánica de la Universidad Kwazulu-Natal, mientras que por la parte de Argentina, el responsable fue el Dr. José Zelasco, del Departamento de Mecánica de la Facultad de Ingeniería de la Universidad de Buenos Aires.

Los estudiantes de posgrado Nicol Naidoo (Sudáfrica) y Jorge Kamlofsky (Argentina) han progresado en sus investigaciones. Estudiantes de ingeniería de ambos países han participado en los desarrollos y asistieron a las presentaciones del progreso del proyecto.

## 2.2. Enfoque metodológico

El plan de desarrollo de esta investigación se orientó hacia el cumplimiento del objetivo principal: presentar algoritmia que permita lograr un rápido esquema de visión 3D que pueda aproximarse a los requerimientos de tiempo real. Para ello se usó un enfoque metodológico cuantitativo (Hernandez et al., 2014) a partir de datos experimentales. Se subdividió al proceso en diferentes tareas alineadas con los objetivos intermedios, de modo de poder presentar, analizar y evaluar avances parciales.

El cronograma preliminar de cada una de las tareas se basó en instancias factibles de ser cumplidas. Así, al lograrse un objetivo intermedio se analizaron resultados medibles, junto con la factibilidad de integración con instancias anteriores, en marco del plan global.

La forma de realizar las investigaciones parciales consiste en cumplir las siguientes tareas: investigación bibliográfica acerca del objetivo, planteo de posibles soluciones al problema, desarrollo de prototipos de software, análisis de los datos generados por los prototipos y validación de los resultados.

Las validaciones parciales mayormente se lograron tras el desarrollo, presentación y aceptación de documentos técnicos que reflejaron el avance parcial del proyecto en eventos científicos de

1 CAETI: Centro de Altos Estudios en Tecnología informática, dependiente de la Facultad de Tecnología informática de la UAI. Sitio Web: http://caeti.uai.edu.ar.

tecnología con referato, como ser: congresos, seminarios, workshops nacionales e internacionales, o en revistas o libros científicos. En todos los casos se incluyen datos experimentales.

En este trabajo se incorporan nuevos datos experimentales generados para el cálculo de complejidad algorítmica, junto con su correspondiente análisis, para la elaboración de las conclusiones.

## 2.3. Algunos resultados de investigación que alimentan a este trabajo

El proyecto se inició luego de la presentación de la tesis de grado de Kamlofsky (2011) en la que se expuso acerca de la *topología digital* y al *análisis de bordes* como un enfoque que podría tratar eficientemente al problema de visión artificial. Esto sentó las bases para el inicio del proyecto dirigido por la Dra. María Lorena Bergamini previamente mencionado.

El primer resultado parcial se logró tras la presentación y aceptación en las Jornadas Argentinas de Robótica de un primer avance (Kamlofsky y Bergamini, 2012): la simplificación de las curvas de borde mediante la poligonalización.

El análisis de curvas de borde se realiza sobre curvas cerradas. Al simplificarse, las curvas se transforman en polígonos cerrados. En Kamlofsky y Bergamini (2013b) se presentó un patrón descriptor de curvas cerradas simplificadas basado en la evolución de la curvatura a lo largo del perímetro. En Kamlofsky y Bergamini (2013a) se incluyó el análisis de la orientación de la forma para hacer más sencillo el hallazgo de un objeto en una imagen usando el patrón descriptor. En Kamlofsky y Bergamini (2014a y 2014b) y en Bergamini y Kamlofsky (2015) se mostraron aplicaciones.

En 2015 se comenzó a estudiar a la visión en tres dimensiones en el marco del proyecto binacional mencionado anteriormente. En Kamlofsky y Bergamini (2015) se presentaron ventajas en el uso de cuaterniones en rotaciones en el espacio. En Bergamini et al. (2016a y 2016b) se lograron publicaciones internacionales conjuntas con referato entre el equipo argentino y el sudafricano relacionadas con el proceso de calibración del sistema de visión robótica.

En 2017, en Bergamini y Kamlofsky (2017) se presentó otro enfoque para la poligonalización de curvas digitales. En Kamlofsky y Bergamini (2017) se presentó la idea del rápido hallazgo de puntos homólogos. Esto se incorporó luego en el trabajo de Naidoo et al. (2017) donde se integraron los avances de las investigaciones de los equipos de Sudáfrica y Argentina.

En 2018 se ampliaron estas ideas y se las incorporó en un artículo (Kamlofsky et al., 2018) donde además se presentan notorias mejoras en los tiempos para la obtención de puntos homólogos a menos de un segundo. En este artículo participaron los integrantes de los equipos de investigación de ambos países. Es destacable que en Junio de 2020 este artículo ha sido indexado en Scopus[2].

2 Vínculo al sitio de Scopus con los detalles del documento: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85061819621&partnerID=40&md5=567acf7c58e5d96274510690e0a79cbf.

# 3. Marco teórico

## 3.1. La imagen digital.

### 3.1.1. La imagen analógica

Cotidianamente se pueden observar innumerables imágenes bidimensionales. Por ejemplo: la pintura hecha por un artista, una imagen natural capturada por una cámara, un telescopio o un microscopio. Representan una variación continua de sombras y tonos. Por esta razón, las imágenes de este tipo reciben el nombre de imágenes continuas o analógicas (Rosenfeld, 1979).

### 3.1.2. Digitalización: El píxel

Para almacenar estas imágenes en una computadora es necesario digitalizarlas. Se inicia en una primera etapa denominada Adquisición: ciertos dispositivos con sensores ópticos sensibles son capaces de obtener variaciones de tonos y sombras. En Rosenfeld y Kak (2014) se brindan más detalles acerca del proceso de adquisición de imágenes. Luego se genera una imagen bidimensional y se la almacenan electrónicamente. A este proceso se lo denomina *digitalización*. Algunos dispositivos de adquisición o digitalización de imágenes son: scanners, cámaras fotográficas digitales, tomógrafos. Matemáticamente, la digitalización es una transformación de un subconjunto del dominio real infinito (objeto o escenario analógico) al codominio digital, discreto y finito (imagen digital). Durante este proceso, se divide a la imagen analógica obtenida en celdas a las que se les asigna un determinado color. A cada una de estas celdas se las denomina píxel (contracción del inglés de las palabras *picture element*).

Un píxel en su definición es sólo una unidad de división de la imagen, sin un tamaño real concreto. Por ejemplo, que una imagen tenga 200 x 100 píxeles, significa que la imagen tiene 200 píxeles de ancho y 100 píxeles de alto, sin saber qué tamaño real tiene. Sólo se sabe que se la ha dividido en 20.000 celdas. Cuando a esa imagen se le asigna la resolución definida por el dispositivo de adquisición, se puede conocer el tamaño tiene la misma y la medida de cada píxel. Si una imagen tiene una resolución de 10 píxeles por pulgada o dpi (iniciales: *dots per inch*), significa que en cada pulgada (2,54 cm) habrá 10 celdas. Cada píxel equivaldrá a un cuadrado de 0,254 cm. de lado. El tamaño de la imagen puede calcularse: 50,8cm x 25,4cm. Mayor resolución implica, más calidad de imagen, pero ocupará más espacio de la memoria.

### 3.1.3. La imagen digital

Una imagen digital es una función $f : A \in Z \times Z \rightarrow Z^n$, $n \in N$ que asigna a cada píxel $(x,y) \in A$ un valor $f(x,y) \in Z^n$ que representa el color de dicho píxel, con Z: el conjunto de los números enteros.

Mientras que $(x,y)$ brindan coordenadas espaciales del píxel, el valor $f(x,y)$ representa su nivel de brillo. El valor $n$ queda determinado según el modo de confección de la imagen digital. Indica en cuántas componentes se descompone al brillo. Si $n = 1$, el brillo se descompone en su escala de grises mientras que si $n > 1$ al brillo se lo descompone en $n$ bandas del espectro de colores (Rosenfeld y Kak, 2014).

Los modos de confección de imágenes digitales pueden clasificarse según su uso y a su vez, según la cantidad de color. Son comúnmente muy usados los modos RGB y CYMK. El modo RGB (iniciales de: red, green, blue) se usa para presentación de imágenes en pantallas (como ser: monitores y televisores) y el color de cada píxel se conforma variando la intensidad de cada una de las tres componentes: rojo, verde y azul, cada una de ellas variando entre 0 y 255 (un byte por componente). Mientras todas las componentes con su máximo valor generan el color blanco, todas las componentes con el mínimo valor generan el negro. Por otro lado, el modo CYMK (cyan, yellow, magenta, key - black) es usado para impresión de imágenes en color. El color de cada píxel se conforma con las cuatro componentes: cyan, amarillo, magenta y negro, cada una de ellas variando entre 0 y 255. Mientras que todas las componentes con su máximo valor generan el negro, todas las componentes con el mínimo valor generan el blanco. Más específicamente, las tres primeras componentes pueden formar la totalidad de los colores, mientras que variando la cuarta, puede formarse toda la escala de grises.

Respecto de la cantidad de color, las imágenes pueden clasificarse en monocromáticas, grises, y color: las imágenes monocromáticas poseen una sola componente, y su valor es 1 ó 0. Las imágenes en escala de grises poseen una sola componente entre 0 y 255: el promedio de las componentes RGB o CYMK. Las imágenes color poseen varias componentes entre 0 y 255 cada una. La Tabla 1 presenta la cantidad de color en imágenes RGB y CYMK.

*Tabla 1: Cantidad de color en imágenes monocromáticas, grises, color, RGB y CYMK*

| Tipos de Imagen / Modo | Definición | MODO RGB | | MODO CYMK | |
|---|---|---|---|---|---|
| | | Conjunto de llegada | Cantidad de colores posibles | Conjunto de llegada | Cantidad de colores posibles |
| Monocromática | $f: A \subset Z \times Z \rightarrow B$ | $B = \{0, 1\}^1$ | 2 | $B = \{0, 1\}^1$ | 2 |
| Gris | $f: A \subset Z \times Z \rightarrow G$ | $G = \{0, 1, \ldots, 255\}^1$ | 256 | $G = \{0, 1, \ldots, 255\}^1$ | 256 |
| Color | $f: A \subset Z \times Z \rightarrow C$ | $C = \{0, 1, \ldots, 255\}^3$ | 16777216 | $C = \{0, 1, \ldots, 255\}^4$ | 4294967296 |

En la ilustración 1[3] se muestra una comparación de la conformación de colores en imágenes en modos RGB y CYMK.

3 Ilustración basada en la imagen presentada en: https://i.ytimg.com/vi/YtH9eXWuf3Y/maxresdefault.jpg

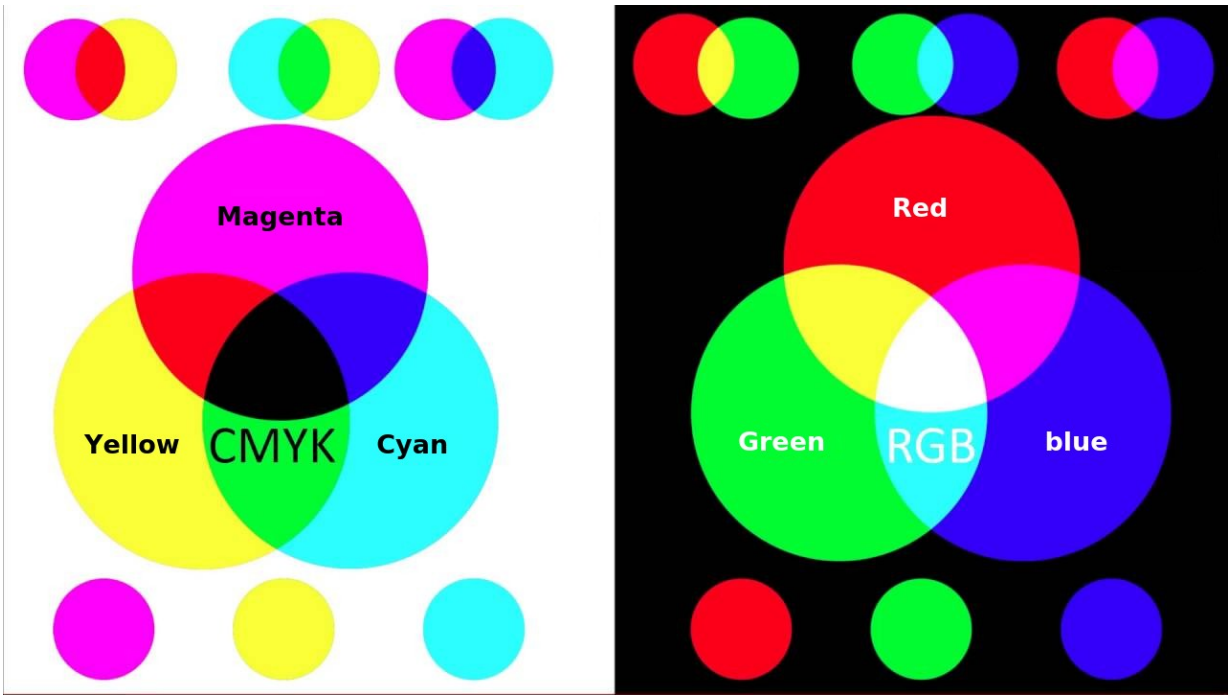



*Ilustración 1: Conformación de colores en modos RGB y CYMK.*

### 3.1.4. Matriz asociada a una imagen digital

Una imagen digital puede considerarse como una matriz cuyos índices de filas y columnas identifican el punto en la imagen, y su valor correspondiente identifica el nivel de gris en ese punto. A dicha matriz se la llama la matriz asociada a la imagen digital. Entonces, una imagen de N x M píxeles tiene asociada una matriz $A \in M$ x $N$. Y la matriz A se conformará de la siguiente manera:

$$\forall (x,y)\, con: 0 \le x < M \wedge 0 \le y < N \wedge x \in Z \wedge y \in Z: \tag{1a}$$

$$\mathrm{A=f(x,y)=}\begin{pmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{pmatrix} \tag{1b}$$

**Ejemplo 1:** En la ilustración 1(a) se muestra una imagen de una letra K, sombreada. Posee 28px de ancho por 28px de alto. La ilustración 1(b) muestra la misma imagen, ampliada 1600%, donde se muestran las grillas que separan a los píxeles.

Entre las propiedades de esa imagen, se sabe que la resolución de la imagen es: 96dpi (96dpi = 96 píxeles/pulgada). Con ello puede calcularse el tamaño de la imagen. Entonces:

$$\mathrm{Resolución=}\frac{\mathrm{cant.\,pixeles}}{\mathrm{longitud}} \Rightarrow \mathrm{longitud=}\frac{\mathrm{cant.\,pixeles}}{\mathrm{Resolución}} \tag{2a}$$

y

$$\text{longitd} = \frac{28\,\text{px}}{96\,\text{px/inch}} = 0{,}292\,\text{inch} \Rightarrow \text{longitud} = 0{,}74\,\text{cm} \quad \text{(2b)}$$

Finalmente, la imagen tiene 0,74 cm x 0,74 cm.

En la ilustración 3 se presenta la imagen ampliada de la ilustración 2(b) donde en cada celda se cambió el color por el valor de la escala de grises correspondiente a cada celda. De ese modo, puede representarse en la imagen su matriz asociada (ecuación 1b) en modo RGB en escala de grises.

## 3.2. El procesamiento de imágenes

### 3.2.1. Algunas causas por las que se requiera el mejoramiento de imágenes

#### Imágenes con bajo contraste

Entender el contenido de una imagen digital parte de identificar en la misma sus componentes. Para ello, es conveniente que sus límites sean claros. En imágenes digitales de bajo contraste, los límites en las componentes no son fáciles de identificar.

**Imágenes con bajo contraste luminoso:** En condiciones adecuadas, las diferencias de luces y sombras facilitan la delimitación de componentes en una imagen. Una causa por la que se obtienen imágenes con bajo contraste es el exceso o falta de luminosidad. Es decir, se obtienen imágenes de bajo contraste cuando la luz es excesiva y no permite la aparición de sombras, o bien cuando es tan escasa que no presenta partes iluminadas que puedan contrastarse con las sombras.

**Imágenes con bajo contraste de color:** También es posible obtener imágenes con bajo contraste cuando a ambos lados de la frontera del objeto se presentan colores similares. En muchos procesos de visión artificial resulta más cómodo trabajar con imágenes en escala de grises: ello se logra con el promedio simple entre los valores de cada componente. Entonces, imágenes con colores diferentes pueden tener escala de gris similar, pero con muy bajo contraste.

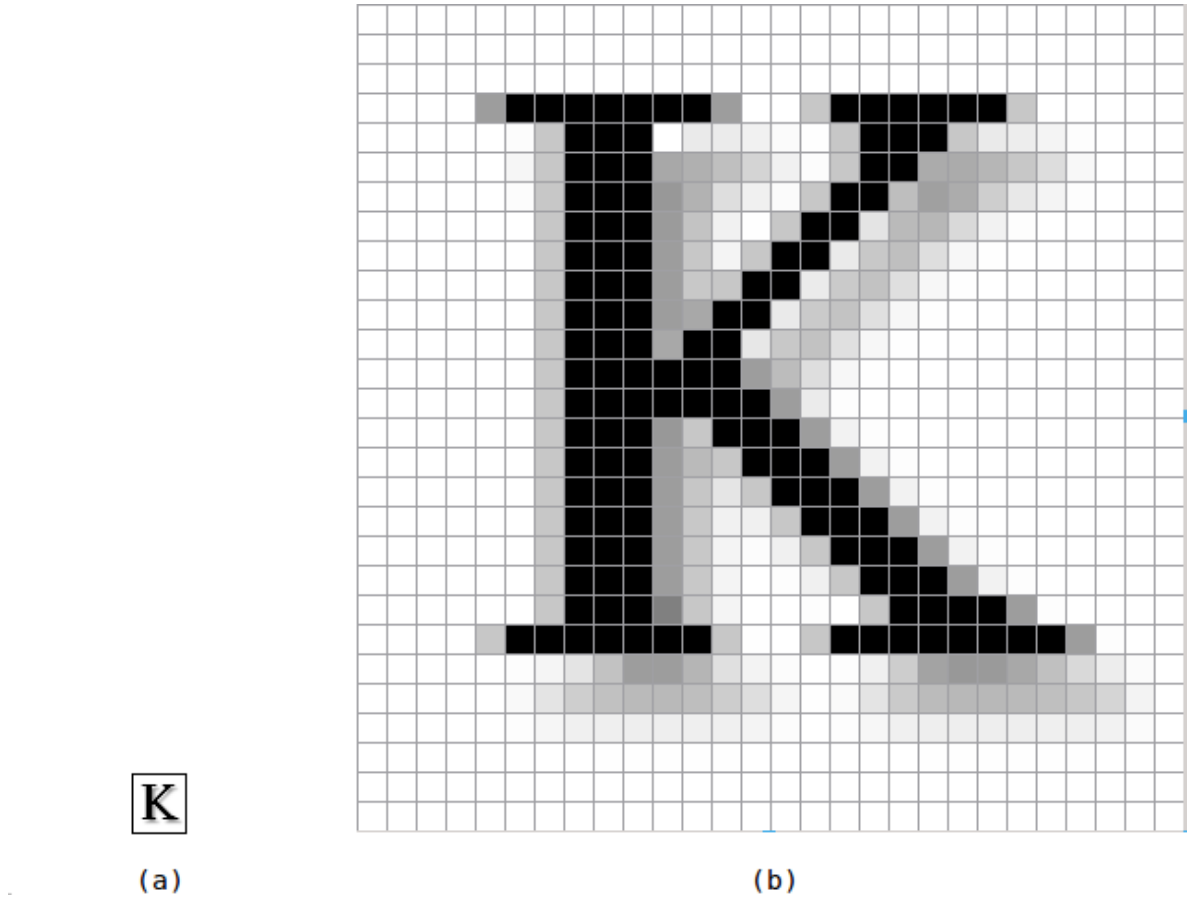



*Ilustración 2: (a) La imagen de una letra K. (b) Los píxeles de la imagen ampliada.*

## El problema del ruido

La digitalización es una transformación de un objeto analógico del dominio real infinito al codominio digital, discreto y finito. Esta transformación logra que objetos reales con infinitos detalles puedan almacenarse en un espacio finito, lo que supone una gran pérdida de información. Esa gran pérdida de información se refleja en la aparición de ruido: la resolución de un dispositivo de adquisición de imágenes define cuántos píxeles por unidad de longitud presentará en la imagen, del objeto analógico original, con infinitos píxeles por unidad de longitud. El ruido se evidencia notoriamente en las fronteras de los objetos donde pueden observarse formas de zig-zag o dientes de serrucho en partes que en la realidad son bordes rectos o suaves.

La imagen presentada en la ilustración 4 se basa en una imagen obtenida de Rosenfeld y Kak (2014). En (a) se presenta una foto aérea de una construcción. En (b) se muestra la parte que figura recuadrada en (a), ampliada 400%. Pueden observarse irregularidades (ruido) en un tramo de la gráfica que debiera ser recto.

Sin embargo, el ruido, tiene también otras fuentes: problemas del dispositivo de captura, mala iluminación, entre otras.

| | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 251 | 252 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 154 | 5 | 0 | 0 | 2 | 0 | 0 | 0 | 179 | 255 | 237 | 216 | 13 | 0 | 0 | 21 | 0 | 0 | 198 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 196 | 13 | 0 | 23 | 253 | 242 | 219 | 230 | 255 | 246 | 185 | 18 | 0 | 0 | 211 | 242 | 251 | 237 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 193 | 0 | 0 | 0 | 168 | 158 | 210 | 223 | 233 | 255 | 193 | 11 | 1 | 182 | 169 | 160 | 200 | 221 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 208 | 0 | 0 | 2 | 162 | 182 | 224 | 232 | 247 | 202 | 0 | 0 | 188 | 150 | 172 | 220 | 233 | 245 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 202 | 3 | 0 | 0 | 143 | 200 | 244 | 244 | 202 | 0 | 11 | 224 | 184 | 192 | 209 | 235 | 253 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 197 | 0 | 0 | 13 | 140 | 202 | 248 | 203 | 0 | 5 | 237 | 196 | 193 | 225 | 242 | 254 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 3 | 0 | 161 | 216 | 192 | 0 | 0 | 228 | 198 | 196 | 221 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 3 | 0 | 164 | 151 | 0 | 14 | 244 | 199 | 200 | 223 | 252 | 255 | 255 | 255 | 255 | 252 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 0 | 7 | 173 | 8 | 0 | 217 | 199 | 185 | 225 | 244 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 1 | 3 | 0 | 0 | 12 | 151 | 188 | 228 | 239 | 255 | 255 | 255 | 255 | 255 | 252 | 253 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 3 | 0 | 18 | 0 | 0 | 8 | 154 | 237 | 255 | 255 | 255 | 255 | 255 | 255 | 250 | 254 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 252 | 255 | 255 | 197 | 0 | 0 | 14 | 148 | 198 | 6 | 4 | 8 | 139 | 233 | 255 | 255 | 255 | 253 | 255 | 252 | 255 | 255 | 255 | 255 | 255 |
| 255 | 253 | 255 | 255 | 255 | 255 | 197 | 0 | 5 | 0 | 169 | 184 | 187 | 0 | 0 | 3 | 164 | 230 | 255 | 254 | 255 | 255 | 252 | 255 | 255 | 255 | 255 | 255 |
| 255 | 252 | 255 | 255 | 255 | 255 | 198 | 5 | 0 | 9 | 157 | 191 | 227 | 199 | 0 | 2 | 2 | 155 | 242 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 250 |
| 255 | 255 | 255 | 253 | 249 | 252 | 200 | 0 | 1 | 4 | 146 | 209 | 242 | 250 | 209 | 0 | 0 | 28 | 138 | 231 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 255 | 255 | 255 | 255 | 255 | 255 | 213 | 0 | 1 | 10 | 155 | 189 | 255 | 233 | 231 | 206 | 7 | 0 | 13 | 157 | 234 | 255 | 255 | 255 | 255 | 255 | 255 | 253 |
| 255 | 255 | 255 | 255 | 255 | 255 | 206 | 0 | 0 | 0 | 177 | 200 | 232 | 255 | 245 | 248 | 210 | 0 | 0 | 15 | 142 | 245 | 245 | 251 | 255 | 255 | 255 | 255 |
| 255 | 255 | 253 | 255 | 255 | 254 | 201 | 0 | 8 | 0 | 120 | 201 | 236 | 241 | 255 | 237 | 251 | 203 | 0 | 0 | 2 | 16 | 149 | 255 | 239 | 255 | 255 | 255 |
| 255 | 255 | 254 | 255 | 172 | 16 | 0 | 0 | 0 | 4 | 0 | 8 | 202 | 248 | 255 | 201 | 0 | 0 | 11 | 2 | 17 | 0 | 0 | 0 | 189 | 227 | 254 | 252 |
| 248 | 252 | 255 | 255 | 255 | 255 | 251 | 225 | 204 | 153 | 160 | 167 | 245 | 233 | 251 | 251 | 241 | 249 | 204 | 154 | 139 | 175 | 166 | 193 | 205 | 235 | 255 | 255 |
| 255 | 251 | 255 | 255 | 255 | 246 | 213 | 212 | 189 | 190 | 182 | 197 | 200 | 223 | 252 | 253 | 255 | 211 | 208 | 197 | 178 | 178 | 193 | 188 | 204 | 219 | 239 | 251 |
| 255 | 255 | 255 | 255 | 253 | 249 | 246 | 243 | 241 | 240 | 238 | 238 | 240 | 245 | 250 | 253 | 252 | 249 | 244 | 239 | 237 | 237 | 239 | 240 | 241 | 244 | 249 | 254 |
| 254 | 255 | 255 | 255 | 255 | 253 | 251 | 249 | 253 | 252 | 251 | 250 | 251 | 253 | 255 | 255 | 255 | 255 | 253 | 250 | 249 | 255 | 255 | 255 | 255 | 255 | 253 | 255 |
| 253 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 254 | 254 | 254 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |
| 254 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 254 | 254 | 254 | 254 | 253 | 253 | 252 | 251 | 252 | 253 | 254 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 | 255 |

*Ilustración 3: El valor de brillo (escala de grises) en cada píxel de una imagen.*

## 3.2.2. Técnicas básicas para el tratamiento de imágenes digitales

### Segmentación

Cuando se pretende obtener información de una imagen es común referirse a esta etapa como análisis de la imagen. La primera parte en el proceso de análisis de imágenes es la Segmentación que consiste en subdividir a la imagen en partes: fondo y objetos.

La segmentación automática es un problema de difícil tratamiento. El nivel y la forma en que se realiza la segmentación depende del problema en sí (Gonzalez & Woods, 1993). Por eso, es muy común prestar especial atención en incluir en los equipos de adquisición sensores y filtros, como ser cámaras térmicas, cámaras con filtros infrarrojos, etc.

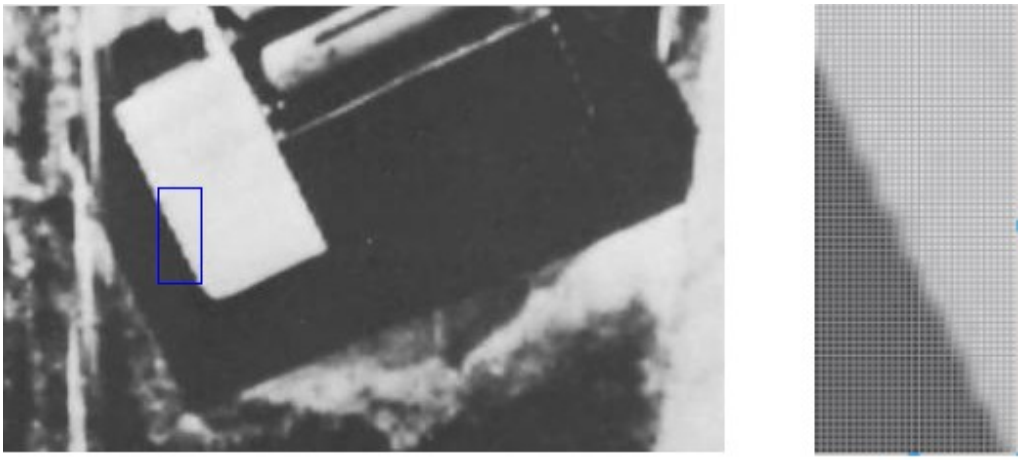



*Ilustración 4: El efecto del ruido en los contornos de un objeto*

Los algoritmos de segmentación para imágenes en escalas de grises generalmente se basan en dos propiedades básicas: la similaridad y la discontinuidad. La primera permite agrupar píxeles vecinos similares en un mismo objeto. Discontinuidades bruscas pueden indicar bordes o fronteras del objeto.

Un enfoque muy sencillo llamado *segmentación por umbralizado* se presentó en Rosenfeld (1979): El objetivo consiste en separar a los objetos de interés del fondo de la imagen. Se determina un valor de gris umbral (dependiente del problema). Luego se recorre la imagen píxel a píxel. Si el valor de gris del píxel es menor que el umbral, se dice que el píxel pertenece al objeto (y viceversa), y se lo marca como tal.

Sea $U$: valor de gris umbral. Sea $f(x,y)$ el valor de gris de la imagen en el píxel $(x,y)$, el umbralizado en el píxel $(x,y)$, puede definirse:

$$u(x,y)=\begin{cases}0 \; Si \; f(x,y)<U \\ 255 \; Si \; f(x,y)\geq U\end{cases} \qquad \text{(3.a.)}$$

La función de umbralizado genera en modo RGB una imagen blanco y negro, donde con negro se destacan los píxeles de interés.

## Umbralizado Múltiple

En imágenes que poseen diferentes valores de gris, se desea determinar zonas que poseen colores similares. Para ello, se determinan varios valores de color umbral según las características de las imágenes.

Sea $C$ el universo de colores RGB. Al rango total de colores $C$ se lo separa en $n$ intervalos. Los extremos de los intervalos conforman los umbrales parciales.

Entonces:

$$C = C_1 \cup C_2 \cup ... \cup C_i \cup ... \cup C_n \; con \; C_1 = [0, u_1], \; C_2 = (u_1, u_2], \; C_i = (u_{i-1}, u_i], \; C_n = (u_{n-1}, 255]$$

Así, cualquier píxel $j$ de una imagen de ancho $a$ y alto $b$, tendrá un color $c_i$ que está en alguno de los intervalos de $C$:

$$f(x_j; y_j) = c_i \in C_i, \; i \in [1; n], \; j \in [0, a * b)$$ .

Los valores de color en cada intervalo $C_i$ se reemplazan por un único valor $a_i$ . Así, la expresión de la función de umbralizado presentada (3.a) puede ampliarse y generalizarse:

$$u(x_j; y_j) = \begin{cases} \alpha_1 \; si \; f(x_j; y_j) = c_1 \in C_1 \\ \alpha_2 \; si \; f(x_j; y_j) = c_2 \in C_2 \\ ... \\ \alpha_n \; si \; f(x_j; y_j) = c_n \in C_n \end{cases} \qquad (3.b)$$

El umbralizado múltiple se utiliza en el tratamiento de imágenes adquiridas con cámaras y/o dispositivos de adquisición especiales, como ser: en imágenes adquiridas mediante cámaras térmicas, donde a cada rango de temperatura se le asigna un color específico.

De este modo, aplicando umbralizado múltiple pueden destacarse zonas en un rango de temperatura de interés, y desecharse al resto. Se lo utiliza también, en imágenes de tomografías computadas, resonancias magnéticas con o sin contraste, entre otras.

## Estiramiento de contraste

Es un proceso que facilita la detección de discontinuidades y similitudes ya que en las zonas de frontera o de cambios, los valores de colores son muy similares, lo que promueve a la formación de ruido en caso de aplicarse a la imagen algún tipo de procesamiento.

El objetivo de esta técnica consiste en obtener una imagen con mayor contraste que la original. Y esto se logra haciendo más oscuros a los píxeles que tienen valor de gris menor que un cierto valor de gris frontera y más claros a los mayores haciendo más nítidas las zonas de frontera y evitando ruidos no deseados, unificando los colores similares.

Una forma de lograrlo es aplicando estiramiento de contraste donde los colores dentro de un intervalo $C_i$ se reemplazan por un único valor $a_i$, siendo este $a_i$, el promedio de colores de toda la zona. Resulta una imagen nítida, con delimitaciones claras en las fronteras, y de fácil tratamiento en instancias de procesamiento.

## Algunas operaciones con imágenes

**Sustracción de imágenes:** La diferencia de imágenes tiene numerosas aplicaciones en segmentación de imágenes. A una imagen obtenida se le puede restar una imagen previamente adquirida de modo de obtener así más detalles. También puede usarse para detección de movimientos (Gonzalez & Woods, 1993).

**Promediado de imágenes:** Si se considera una imagen ruidosa, el ruido puede atenuarse mediante la conformación de una nueva imagen lograda tras el cálculo del promedio de n imágenes ruidosas obtenidas en distintas tomas. Los sectores sin ruido no tendrán variabilidad, mientras que en las zonas ruidosas, el promedio atenuará el ruido.

**Uso de máscaras especiales:** El enfoque consiste en sumar los productos entre los coeficientes de la máscara ($w_i$) y de las intensidades de los píxeles ($z_i$) en una ubicación especial de la imagen. Entonces, el nivel de gris bajo una máscara (lineal) de 3x3 será: $R=w_1.z_1+w_2.z_{12}...+w_9.z_9$.

Se usan algunas máscaras para suavizado o difuminado, para acentuar detalles, para detectar puntos aislados, líneas o bordes (Gonzalez & Woods, 1993).

# 3.3. Transformaciones lineales geométricas

## 3.3.1. Introducción

Una transformación lineal (TL) geométrica se puede aplicar a un vector posición geométrico o a un conjunto de vectores posición geométricos. Por ser TL una transformación lineal, existirá una matriz operadora de la TL. Eso significa que esa matriz pre-multiplica a una matriz que contiene a los vectores posición (en columna), de modo que en una sola operación matricial se logra la transformación lineal geométrica de un vector o de un conjunto de vectores.

Ya sea en el caso de las imágenes digitales o en modelos tridimensionales, ese vector o conjunto de vectores puede contener a los vectores de un conjunto de puntos aislados, a los vectores del conjunto de vértices de un polígono, de los puntos del borde de un objeto, de los puntos de un objeto, de una porción o ventana de la imagen o bien a toda la imagen.

Dado que un polígono puede definirse a partir de sus vértices ordenados (y dado que este puede ser la simplificación de los bordes de un objeto), al polígono se lo puede definir como una matriz donde cada columna corresponde con el vector posición de cada vértice. De ese modo, puede aplicarse una TL sobre todo un polígono con una sola operación: el producto entre la matriz de transformación y la matriz del polígono.

A continuación se presentan ejemplos de su uso:

**Ejemplo 2:** Si se conoce que una cámara está rotada 5° en comparación con otra cámara que apunta al mismo objeto y luego se desean comparar sus imágenes, generalmente se comenzaría con alinear la orientación de la imágenes, rotando la primer imagen 5° en sentido opuesto.

**Ejemplo 3:** Otro ejemplo se presenta cuando se usan scanners automáticos de alimentación múltiple. En muchos casos, el mecanismo de transporte de las hojas posee algún desbalance mecánico. Resulta entonces, que todas las imágenes aparecen algo rotadas. Esto se corrige aplicando la correspondiente rotación inversa al total de las imágenes.

En esta sub-sección, se definen las nociones elementales de las TL que se aplican en este trabajo.

## 3.3.2. Transformaciones Lineales: Definición

Sea $T$ una función en un espacio vectorial $V$ a un espacio vectorial $W$. Entonces $T$ es una transformación lineal si cumple:

$$I.\ \forall\ \bar{u} \in V\ ,\forall\ \bar{v} \in W : T\left(\bar{u}+\bar{v}\right)=T\left(\bar{u}\right)+T\left(\bar{v}\right) \tag{4a}$$

$$II.\ \forall\ \bar{u} \in V\ ,\forall\ c \in \mathbb{R} : T\left(c\ .\ \bar{u}\right)=c\ .\ T\left(\bar{u}\right) \tag{4b}$$

## 3.3.3. Transformaciones Lineales: Propiedad

La siguiente propiedad presenta a las matrices como operadores de transformaciones lineales.

$$Sea\ T : TL,\ X \in \mathbb{R}^{nxm},\ A \in \mathbb{R}^{mxn} / T\left(X\right)=A\ .\ X \Rightarrow A\ es\ la\ matriz\ asociada\ a\ la\ transformación\ lineal\ T\ . \tag{5}$$

## 3.3.4. Transformaciones lineales geométricas: Definición

Son transformaciones lineales que se realizan sobre uno o más vectores geométricos, en el plano o en el espacio. Puede hallarse información en Lengyel (2011) y detalles de implementación en Kamlofsky y Bergamini (2014b).

## 3.3.5. Escalados

Se llama escalado del vector $X$ a la TL:

$$T\left(X\right)=E\ .\ X=X\ ' \tag{6a}$$

donde $E$ es una matriz cuadrada, diagonal con elementos no nulos en la diagonal, de orden 2 o 3, dependiendo si la TL es en el plano o en el espacio y $X$' es el vector que resulta del escalado del vector $X$.

El escalado, es un proceso que puede revertirse mediante el uso de la matriz inversa al escalado $E^{-1}$, que al tratarse de una matriz diagonal cuadrada con elementos no nulos en la diagonal, existe, y su obtención es sencilla. Su cálculo se presenta más adelante.

Pre-multiplicando por $E^{-1}$ a ambos miembros de la expresión (6.a):

$$\begin{aligned} &E^{-1}.E.X=E^{-1}.X\ ' \\ &E^{-1}.E.X=E^{-1}.X\ ' \\ &I.X=E^{-1}.X\ ' \\ &X=E^{-1}.X\ ' \end{aligned} \tag{6b}$$

### Escalado en el plano

Dados dos escalares no nulos $a$, $b$, se llama escalado del vector $X$ en el plano a la TL:

$$T(X)=\begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}.\begin{pmatrix} x \\ y \end{pmatrix}=\begin{pmatrix} x' \\ y' \end{pmatrix}=X\ ' \tag{6c}$$

con:

$$X = \begin{pmatrix} x \\ y \end{pmatrix}$$

el vector posición del punto a transformar (que puede ser una matriz de puntos), y

$$E = \begin{pmatrix} a & 0 \\ 0 & b \end{pmatrix}$$

la matriz de escalado, donde los escalares no nulos $a$, $b$ escalan $X$ en las direcciones de los ejes $x$ e $y$ respectivamente.

**Ejemplo 4:** Sea el rectángulo $X$ conformado por: $X=\{A(0;0), B(2;0), C(2;1), D(0;1)\}$. Obtener $X'$: el escalado de $X$, 3/2 veces en $x$ y 2 veces en $y$.

Aplicando ecuación 6a y 6c:

$$X' = T(X) = E \,.\, X = \begin{pmatrix} \frac{3}{2} & 0 \\ 0 & 2 \end{pmatrix} . \begin{pmatrix} 0 & 2 & 2 & 0 \\ 0 & 0 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 3 & 3 & 0 \\ 0 & 0 & 2 & 2 \end{pmatrix}$$

Finalmente: $X'=\{A'(0;0), B'(3;0), C'(3;2), D'(0;2)\}$

*Nota: Las expresiones matriciales de las transformaciones que se presentan en esta sección deben realizarse con vectores columna.*

La ilustración 5 ilustra el ejemplo 4.

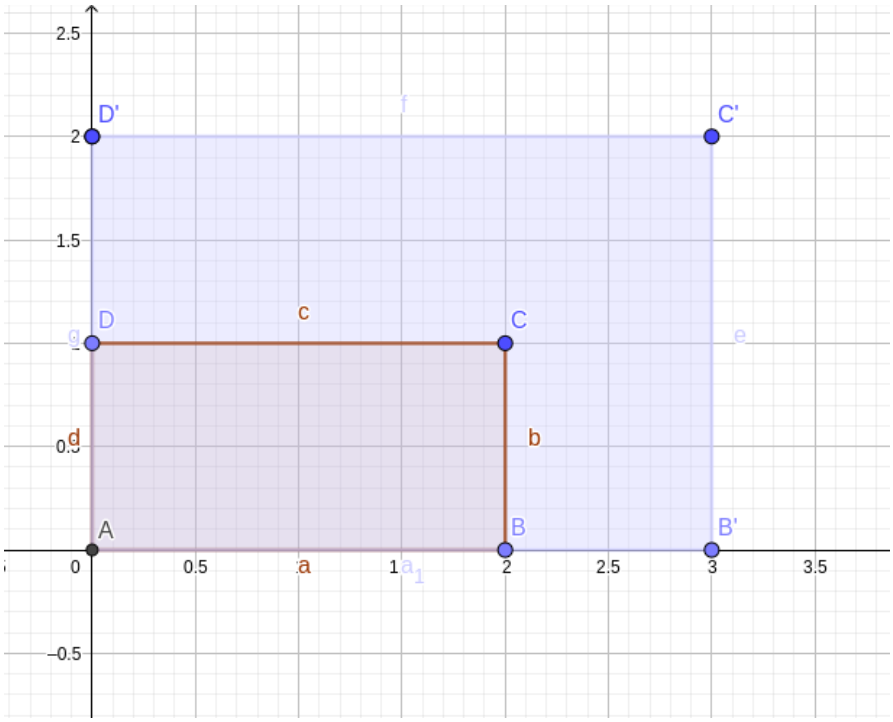



*Ilustración 5: Escalado no uniforme en el plano de los vértices de un rectángulo*

## Inversa del escalado en el plano

Si $E$ es la matriz de escalado en el plano (según 6c), su matriz inversa es:

$$E^{-1}=\begin{pmatrix}\frac{1}{a} & 0\\ 0 & \frac{1}{b}\end{pmatrix}$$

Si todos los elementos de la diagonal de $E$ son no nulos, es fácil ver que existe $E^{-1}$: la inversa de $E$.

$$E^{-1}.E=E.E^{-1}=\begin{pmatrix}\frac{1}{a} & 0\\ 0 & \frac{1}{b}\end{pmatrix}.\begin{pmatrix}a & 0\\ 0 & b\end{pmatrix}=\begin{pmatrix}a & 0\\ 0 & b\end{pmatrix}.\begin{pmatrix}\frac{1}{a} & 0\\ 0 & \frac{1}{b}\end{pmatrix}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}=I$$

## Escalados especiales en el plano

Algunos escalados especiales en el plano se dan para valores especiales de los escalares $a$ y $b$:

Si $a = 1$ y $b = 1$: transformación identidad.

Si $a = -1$ y $b = 1$: simetría axial respecto al el eje $y$.

Si $a = 1$ y $b = -1$: simetría axial respecto al el eje $x$.

Si $a = -1$ y $b = -1$: simetría central (respecto del origen de coordenadas).

Si $a = b$: escalado uniforme.

## Escalado en el espacio

Dados tres escalares no nulos $a$, $b$, $c$, se llama escalado en el espacio a la TL:

$$T\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = X' \qquad \text{(6d)}$$

con:

$$X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

el vector posición del punto a transformar,

$$E = \begin{pmatrix} a & 0 & 0 \\ 0 & b & 0 \\ 0 & 0 & c \end{pmatrix}$$

es la matriz de escalado, donde los escalares no nulos $a$, $b$ y $c$ escalan $X$ en las direcciones de los eje $x$, $y$ y $z$ respectivamente.

**Ejemplo 5:** Sea el prisma rectangular $X$ conformado por sus vértices: $X = \{A(0;0;0), B(1;0;0), C(1;2;0), D(0;2;0), E(0;0;1), F(1;0;1), G(1;2;1), H(0;2;1)\}$. Obtener $X'$: el escalado de $X$, 2 veces en x, 3/2 veces en y y 2 veces en $z$.

Aplicando ecuación 6d:

$$T(X) = \begin{pmatrix} 2 & 0 & 0 \\ 0 & \frac{3}{2} & 0 \\ 0 & 0 & 2 \end{pmatrix} . \begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 2 & 2 & 0 & 0 & 2 & 2 & 0 \\ 0 & 0 & 3 & 3 & 0 & 0 & 3 & 3 \\ 0 & 0 & 0 & 0 & 2 & 2 & 2 & 2 \end{pmatrix} = X'$$

Finalmente: $X' = \{A(0;0;0), L(2;0;0), M(2;3;0), O(0;3;0), P(0;0;2), I(2;0;2), J(2;3;2), K(0;3;2)\}$

La ilustración 6 ilustra el Ejemplo 5.

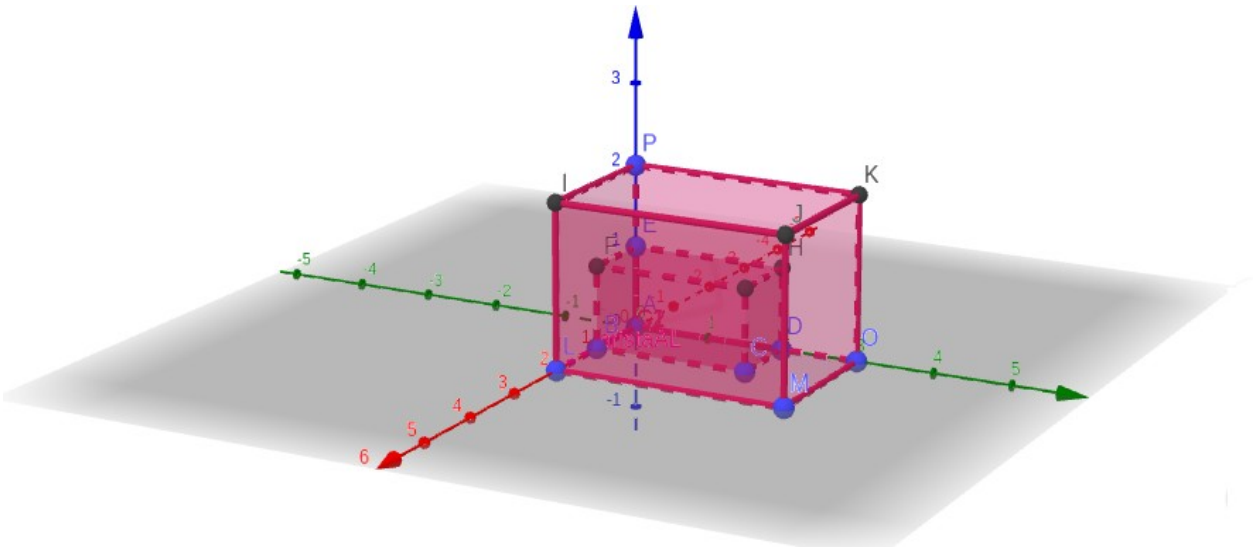



*Ilustración 6: Escalado no uniforme en el espacio de los vértices de un prisma.*

## Inversa del escalado en el espacio

Si $E$ es la matriz de escalado en el espacio (según la ecuación 6d), su matriz inversa es:

$$E^{-1}=\begin{pmatrix}\frac{1}{a} & 0 & 0\\ 0 & \frac{1}{b} & 0\\ 0 & 0 & \frac{1}{c}\end{pmatrix}$$

Es fácil ver que $E^{-1}$ es la inversa de $E$ ya que:

$$E^{-1}.E=E.E^{-1}=\begin{pmatrix}\frac{1}{a} & 0 & 0\\ 0 & \frac{1}{b} & 0\\ 0 & 0 & \frac{1}{c}\end{pmatrix}.\begin{pmatrix}a & 0 & 0\\ 0 & b & 0\\ 0 & 0 & c\end{pmatrix}=\begin{pmatrix}a & 0 & 0\\ 0 & b & 0\\ 0 & 0 & c\end{pmatrix}.\begin{pmatrix}\frac{1}{a} & 0 & 0\\ 0 & \frac{1}{b} & 0\\ 0 & 0 & \frac{1}{c}\end{pmatrix}=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 1\end{pmatrix}=I$$

## Algunos escalados especiales en el espacio

Se dan para valores especiales de los escalares $a$, $b$ y $c$:

Si $a = 1$, $b = 1$ y $c = 1$: transformación Identidad.

Si $a = -1$, $b = 1$ y $c = 1$: simetría respecto al plano $yz$.

Si $a = 1$, $b = -1$ y $c = 1$: simetría respecto al plano $xz$.

Si $a = -1$, $b = 1$ y $c = -1$: simetría respecto al plano $xy$.

Si $a = -1$, $b = -1$ y $c = 1$: simetría axial respecto al eje $z$.

Si $a = 1$, $b = -1$ y $c = -1$: simetría axial respecto al eje $x$.

Si $a = -1$, $b = 1$ y $c = -1$: simetría axial respecto al eje $y$.

Si $a = -1$, $b = -1$ y $c = -1$: simetría central (respecto del origen de coordenadas).

Si $a = b = c$: escalado uniforme.

## 3.3.6. Proyecciones ortogonales sobre ejes y planos coordenados

Las proyecciones ortogonales sobre ejes o planos coordenados son transformaciones geométricas no inversibles donde la matriz de transformación es una matriz cuadrada, diagonal, de orden 2 o 3 (según si la transformación se da en el plano o en el espacio) que solo contiene ceros y unos en la diagonal, con al menos un uno y un cero en ella.

### Proyección ortogonal de un vector en el plano sobre el eje x

Se llama proyección ortogonal en el plano del vector $X$ sobre el eje x a la TL:

$T(X) = P_x.X = X'$

$$T(X)=\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}.\begin{pmatrix}x\\ y\end{pmatrix}=\begin{pmatrix}x\\ 0\end{pmatrix}=X' \qquad (6e)$$

con:

$$X=\begin{pmatrix}x\\ y\end{pmatrix}$$

el vector posición del punto a transformar (que puede ser una matriz de puntos),

$$P_x=\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}$$

la matriz de proyección ortogonal en el plano de un vector sobre el eje x.

**Ejemplo 6:** Sea el vector $\bar{a}$ el vector posición del punto $A(2;1)$. Obtener $\bar{a}'$: el vector de la proyección ortogonal del vector $\bar{a}$ sobre el eje $x$.

Aplicando la ecuación 6e:

$$T(\bar{a})=P_{x.}\bar{a}=\begin{pmatrix}1 & 0\\ 0 & 0\end{pmatrix}.\begin{pmatrix}2\\ 1\end{pmatrix}=\begin{pmatrix}2\\ 0\end{pmatrix}=\bar{a}'$$

La Ilustración 7 representa al ejemplo 6.

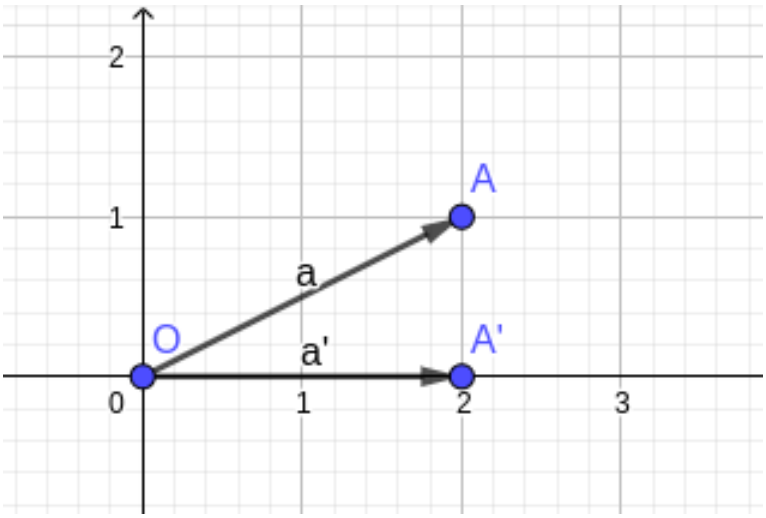



*Ilustración 7: Proyección de un vector sobre el eje x*

## Proyección ortogonal de un vector en el plano sobre el eje y

Se llama proyección ortogonal en el plano del vector $X$ sobre el eje y a la TL:

$T(X) = P_y.X = X'$

$$T(X)=\begin{pmatrix}0 & 0\\ 0 & 1\end{pmatrix}.\begin{pmatrix}x\\ y\end{pmatrix}=\begin{pmatrix}0\\ y\end{pmatrix}=X' \quad (6f)$$

## Proyección ortogonal de un vector en el espacio sobre el eje x

Se llama proyección ortogonal en el espacio del vector $X$ sobre el eje x a la TL:

$T(X) = P_x.X = X'$

$$T(X)=\begin{pmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 0\end{pmatrix}.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}x\\ 0\\ 0\end{pmatrix}=X' \quad (6g)$$

## Proyección ortogonal de un vector en el espacio sobre el eje y

Se llama proyección ortogonal en el espacio del vector $X$ sobre el eje y a la TL:

$T(X) = P_y.X = X'$

$$T(X)=\begin{pmatrix}0 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{pmatrix}.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}0\\ y\\ 0\end{pmatrix}=X' \quad (6h)$$

## Proyección ortogonal de un vector en el espacio sobre el eje z

Se llama proyección ortogonal en el espacio del vector $X$ sobre el eje z a la TL:

$T(X) = P_z.X = X'$

$$T(X)=\begin{pmatrix}0 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 1\end{pmatrix}.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}0\\ 0\\ z\end{pmatrix}=X' \qquad (6i)$$

## Proyección ortogonal de un vector en el espacio sobre el plano xy

Se llama proyección ortogonal en el espacio del vector $X$ sobre el plano $xy$ a la TL:

$T(X) = P_{xy}.X = X'$

$$T(X)=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{pmatrix}.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}x\\ y\\ 0\end{pmatrix}=X' \qquad (6j)$$

**Ejemplo 7:** Sea el vector $\bar{b}$ el vector posición del punto $B(3;4;2)$. Obtener $\bar{b}'$: la proyección ortogonal del vector $\bar{b}$ sobre el plano $xy$.

Aplicando la ecuación 6j:

$$T(\bar{b})=P_{xy}.\bar{b}=\begin{pmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ 0 & 0 & 0\end{pmatrix}.\begin{pmatrix}3\\ 4\\ 2\end{pmatrix}=\begin{pmatrix}3\\ 4\\ 0\end{pmatrix}=\bar{b}'$$

La Ilustración 8 representa al ejemplo 7.

## Proyección ortogonal de un vector en el espacio sobre el plano xz

Se llama proyección ortogonal en el espacio del vector $X$ sobre el plano $xz$ a la TL:

$T(X) = P_{xz}.X = X'$

$$T(X)=\begin{pmatrix}1 & 0 & 0\\ 0 & 0 & 0\\ 0 & 0 & 1\end{pmatrix}.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}x\\ 0\\ z\end{pmatrix}=X' \qquad (6k)$$

## Proyección ortogonal de un vector en el espacio sobre el plano yz

Se llama proyección ortogonal en el espacio del vector $X$ sobre el plano $yz$ a la TL:

$T(X) = P_{yz}.X = X'$

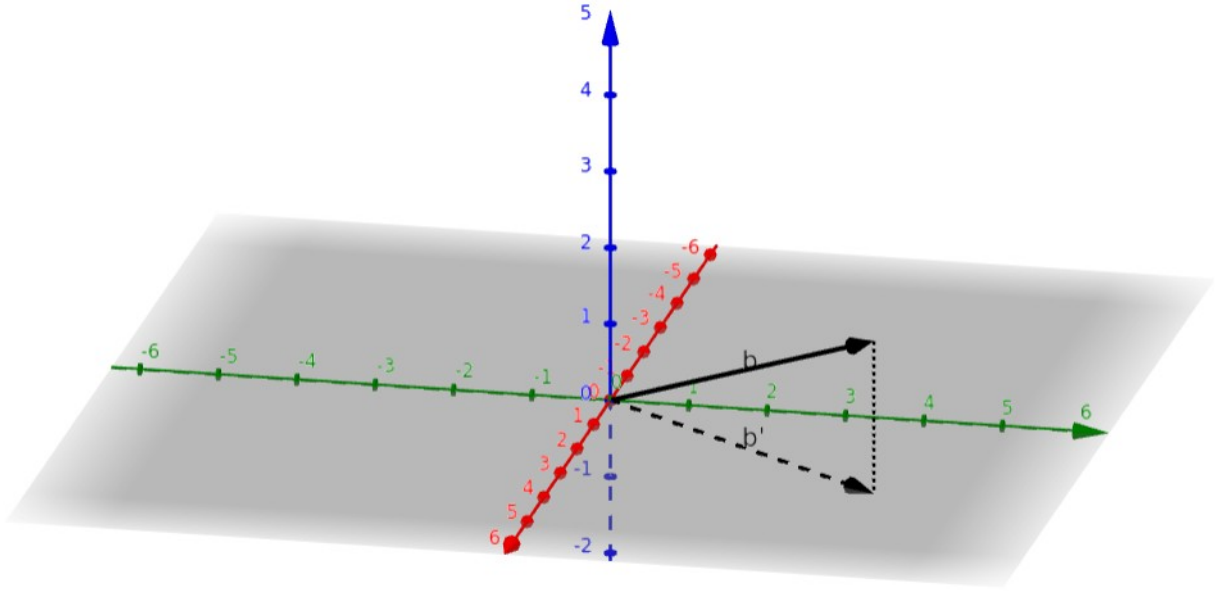



*Ilustración 8: Proyección del vector $\bar{b}$ sobre el plano coordenado xy*

$$T(X)=\begin{pmatrix}0&0&0\\0&1&0\\0&0&1\end{pmatrix}.\begin{pmatrix}x\\y\\z\end{pmatrix}=\begin{pmatrix}0\\y\\z\end{pmatrix}=X' \tag{6l}$$

## 3.3.7. Rotaciones

La rotación de un vector en un sentido antihorario un ángulo a es una transformación lineal dada por:

$$T(X)=R.X=X' \tag{7a}$$

donde $R$ es la matriz de rotación asociada a la transformación lineal T, con $R$ matriz ortonormal.

Las rotaciones, son también, procesos que puede revertirse mediante el uso de la matriz inversa. Y dado que las matrices de rotación son matrices ortonormales, entonces, su inversa es su traspuesta. Entonces:

$$R^{-1}=R^{T} \tag{7b}$$

### Rotación en el plano

La rotación en el plano de un vector en un sentido antihorario a un ángulo a es una transformación lineal dada por (según ecuación 7a):

$$T(X)=R.X=X'$$

$$T(X)=\begin{pmatrix}\cos\alpha & -sen\,\alpha\\ sen\,\alpha & \cos\alpha\end{pmatrix}.\begin{pmatrix}x\\ y\end{pmatrix}=\begin{pmatrix}x'\\ y'\end{pmatrix}=X' \qquad (7c)$$

con:

$$X=\begin{pmatrix}x\\ y\end{pmatrix}$$

el vector posición del punto a transformar,

$$R=\begin{pmatrix}\cos\alpha & -sen\,\alpha\\ sen\,\alpha & \cos\alpha\end{pmatrix}$$

es la matriz de rotación asociada a la transformación lineal $T$ que rota al vector $X$ un ángulo *a* en sentido anti-horario.

**Ejemplo 8:** Sea el triángulo $X=\{A(0;0),\ B(2;0),\ C(2;1)\}$. Obtener $X'$: la rotación de $X$, 30° en sentido antihorario.

Según ecuación 7a:

$$T(X)=R\,.\,X=X'$$

Aplicando ecuación 7c:

$$T\left[\begin{pmatrix}0 & 2 & 2\\ 0 & 0 & 1\end{pmatrix}\right]=\begin{pmatrix}\frac{\sqrt{3}}{2} & \frac{-1}{2}\\ \frac{1}{2} & \frac{\sqrt{3}}{2}\end{pmatrix}.\begin{pmatrix}0 & 2 & 2\\ 0 & 0 & 1\end{pmatrix}=\begin{pmatrix}0 & \sqrt{3} & \sqrt{3}-\frac{1}{2}\\ 0 & 1 & 1+\frac{\sqrt{3}}{2}\end{pmatrix}=X'$$

Finalmente: $X'=\{A(0;0),B(\sqrt{3};1),C(\sqrt{3}-\frac{1}{2};1+\frac{\sqrt{3}}{2})\}$

La ilustración 9 grafica el ejemplo 8.

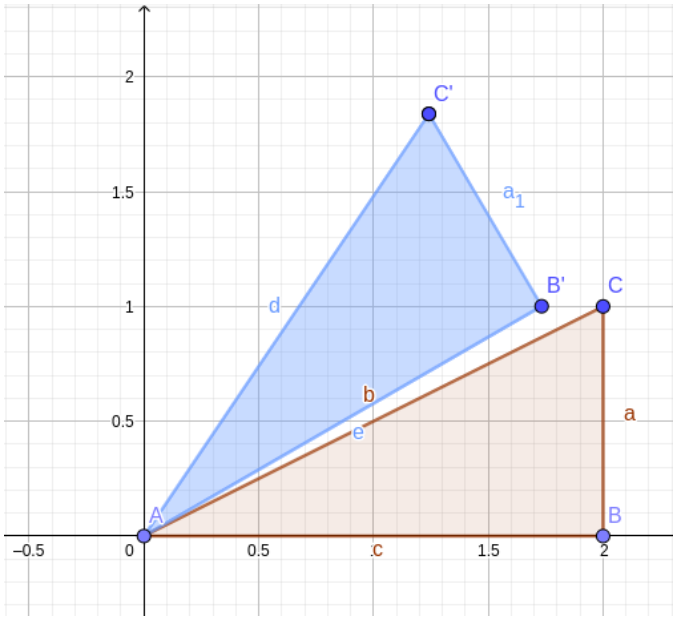



*Ilustración 9: Rotación de los vértices de un triángulo.*

## Inversa de la rotación en el plano

Si $R$ es la matriz de rotación en el plano (presentada en 7c), su matriz inversa es, según 7b, su traspuesta:

$$R^{-1}=\begin{pmatrix}\cos(\alpha) & sen(\alpha)\\ -sen(\alpha) & \cos(\alpha)\end{pmatrix}$$

Es fácil ver que $R^{-1}$ es la inversa de $R$ ya que:

$$R^{-1}.R=R.R^{-1}=\begin{pmatrix}\cos(\alpha) & sen(\alpha)\\ -sen(\alpha) & \cos(\alpha)\end{pmatrix}.\begin{pmatrix}\cos(\alpha) & -sen(\alpha)\\ sen(\alpha) & \cos(\alpha)\end{pmatrix}=\begin{pmatrix}\cos(\alpha) & -sen(\alpha)\\ sen(\alpha) & \cos(\alpha)\end{pmatrix}.\begin{pmatrix}\cos(\alpha) & sen(\alpha)\\ -sen(\alpha) & \cos(\alpha)\end{pmatrix}=\begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}=I$$

## Rotaciones en el espacio

La expresión 7a puede extenderse para el espacio:

$$T\begin{pmatrix}x\\ y\\ z\end{pmatrix}=R.\begin{pmatrix}x\\ y\\ z\end{pmatrix}=\begin{pmatrix}x'\\ y'\\ z'\end{pmatrix}=X' \qquad (7d)$$

donde $R$ es la matriz de rotación del vector $X$ un ángulo a.

$$R=\begin{pmatrix}1 & 0 & 0\\ 0 & \cos(\alpha) & -sen(\alpha)\\ 0 & sen(\alpha) & \cos(\alpha)\end{pmatrix} \qquad (7e)$$

$R$ en la ecuación 7e realiza la rotación de $X$ un ángulo a en sentido positivo en torno al eje $x$.

$$R=\begin{pmatrix}\cos(\alpha) & 0 & sen(\alpha)\\ 0 & 1 & 0\\ -sen(\alpha) & 0 & \cos(\alpha)\end{pmatrix} \qquad (7f)$$

$R$ en la ecuación 7f realiza la rotación de $X$ un ángulo a en sentido positivo en torno al eje $y$.

$$R=\begin{pmatrix}\cos(\alpha) & -sen(\alpha) & 0\\ sen(\alpha) & \cos(\alpha) & 0\\ 0 & 0 & 1\end{pmatrix} \qquad (7g)$$

$R$ en la ecuación 7g realiza la rotación de $X$ un ángulo a en sentido positivo en torno al eje $z$.

**Ejemplo 9:** Sea el prisma $X=\{A(0;0;0),\ B(1;0;0),\ C(1;2;0),\ D(0;2;0),\ E(0;0;1),\ F(1;0;1),\ G(1;2;1),\ H(0;2;1)\}$. Obtener $X'$: una rotación de $X$ a un ángulo de 180° alrededor del eje $x$.

Aplicando 7e en 7d:

$$T(X)=\begin{pmatrix}1 & 0 & 0\\ 0 & -1 & 0\\ 0 & 0 & -1\end{pmatrix}.\begin{pmatrix}0 & 1 & 2 & 0 & 0 & 1 & 1 & 0\\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2\\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1\end{pmatrix}=\begin{pmatrix}0 & 1 & 2 & 0 & 0 & 1 & 1 & 0\\ 0 & 0 & -2 & -2 & 0 & 0 & -2 & -2\\ 0 & 0 & 0 & 0 & -1 & -1 & -1 & -1\end{pmatrix}=X'$$

Finalmente:

$X'=\{A(0;0;0),\ L(1;0;0),\ M(2;-2;0),\ O(0;-2;0),\ P(0;0;-1),\ I(1;0;-1),\ J(1;-2;-1),\ K(0;-2;-1)\}$

La ilustración 10 grafica al ejemplo 9.

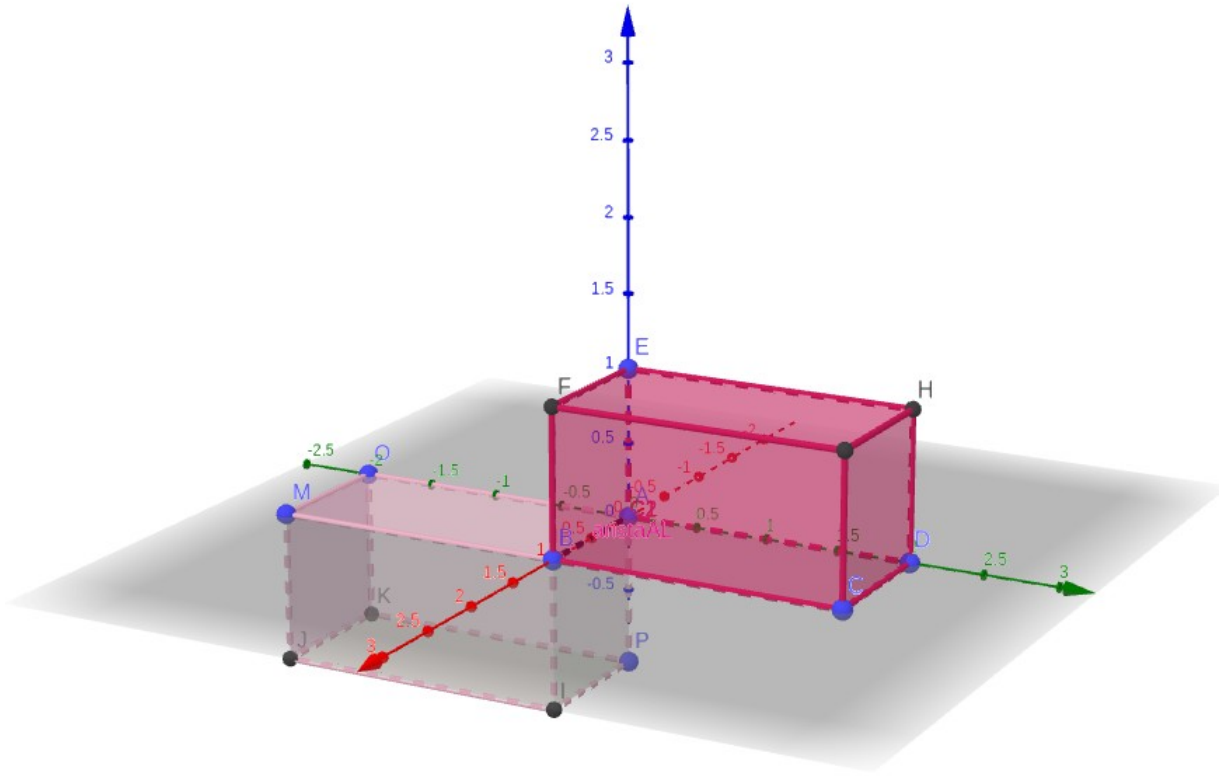



*Ilustración 10: Rotación del prisma alrededor del eje x*

## Inversa de las rotaciones en el espacio

$R^{-1}$ es la matriz de rotación inversa de $R$ (presentada en la ecuación 7d) que rota al vector $X$ un ángulo a.

$$R^{-1}=\begin{pmatrix}1 & 0 & 0\\ 0 & \cos(\alpha) & sen(\alpha)\\ 0 & -sen(\alpha) & \cos(\alpha)\end{pmatrix} \tag{7h}$$

La matriz $R$ (presentada en la ecuación 7e) rota al vector $X$ un ángulo a alrededor del eje $x$. $R^{-1}$ presentada en la ecuación 7h realiza la rotación inversa de la matriz de rotación $R$.

$$R^{-1}=\begin{pmatrix}\cos(\alpha) & 0 & -sen(\alpha)\\ 0 & 1 & 0\\ sen(\alpha) & 0 & \cos(\alpha)\end{pmatrix} \tag{7i}$$

La matriz $R$ (presentada en la ecuación 7f) rota al vector $X$ un ángulo a alrededor del eje $y$. $R^{-1}$ presentada en la ecuación 7i realiza la rotación inversa de la matriz de rotación $R$.

$$R^{-1}=\begin{pmatrix}\cos(\alpha) & sen(\alpha) & 0\\ -sen(\alpha) & \cos(\alpha) & 0\\ 0 & 0 & 1\end{pmatrix} \tag{7j}$$

La matriz $R$ (presentada en la ecuación 7g) rota al vector $X$ un ángulo a alrededor del eje $z$. $R^{-1}$ presentada en la ecuación 7j realiza la rotación inversa de la matriz de rotación $R$.

## 3.3.8. Composición de transformaciones geométricas

La composición de transformaciones lineales geométricas es equivalente al producto de las matrices asociadas a cada transformación lineal. Esto es:

$$Si: T_1(X) = A.X\ ,\ T_2(X) = B.X \Rightarrow (T_1 \circ T_2)(X) = T_1(T_2(x)) = A.B.X \tag{8}$$

**Ejemplo 10:** Sea el triángulo $X$ definido por sus vértices: $X=\{A(0;0),\ B(2;0),\ C(2;1)\}$. Realizar una rotación de $X$ 30° en sentido antihorario. Luego, al resultado obtenido, realizarle un escalado que escale sus dimensiones al doble en $x$ y al triple en $y$.

En este caso, $T_1$ es la rotación (lo primero que se hace), y la siguiente transformación $T_2$ es el escalado. Entonces aplicando la ecuación 8:

$$Si: T_1(X) = R.X\ ,\ T_2(X) = E.X \Rightarrow (T_2 \circ T_1)(X) = E.R.X = X'$$

$$E.R.X = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix} . \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix} . \begin{pmatrix} 0 & 2 & 2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} \sqrt{3} & -1 \\ \frac{3}{2} & \frac{3}{2}\sqrt{3} \end{pmatrix} . \begin{pmatrix} 0 & 2 & 2 \\ 0 & 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 2\sqrt{3} & 2\sqrt{3}-1 \\ 0 & 3 & \frac{3}{2}\sqrt{3}+3 \end{pmatrix} = X'$$

Finalmente: $X' = \{A(0;0), D(2\sqrt{3};3), E(2\sqrt{3}-1; \frac{3}{2}\sqrt{3}+3)\}$

La ilustración 11 grafica al ejemplo 10.

*Observación: En este ejemplo se calculó (E.R).X, donde E.R es la matriz asociada a la transformación lineal. Pero como el producto de matrices cumple la propiedad asociativa, podría haberse calculado E.(R.X) y el resultado hubiera sido el mismo X'.*

Puede notarse que dado un vector posición, se lo puede transformar en cualquier otro vector posición del espacio mediante composiciones de las transformaciones anteriores: Mientras que la dirección y el sentido se puede lograr componiendo rotaciones (multiplicando matrices de rotación alrededor de los ejes), el tamaño adecuado se puede lograr componiendo a las rotaciones con un escalado uniforme.

**Propiedad:** El producto de las matrices de transformación no es conmutativo.

Por ello, debe respetarse el orden al realizarse la composición.

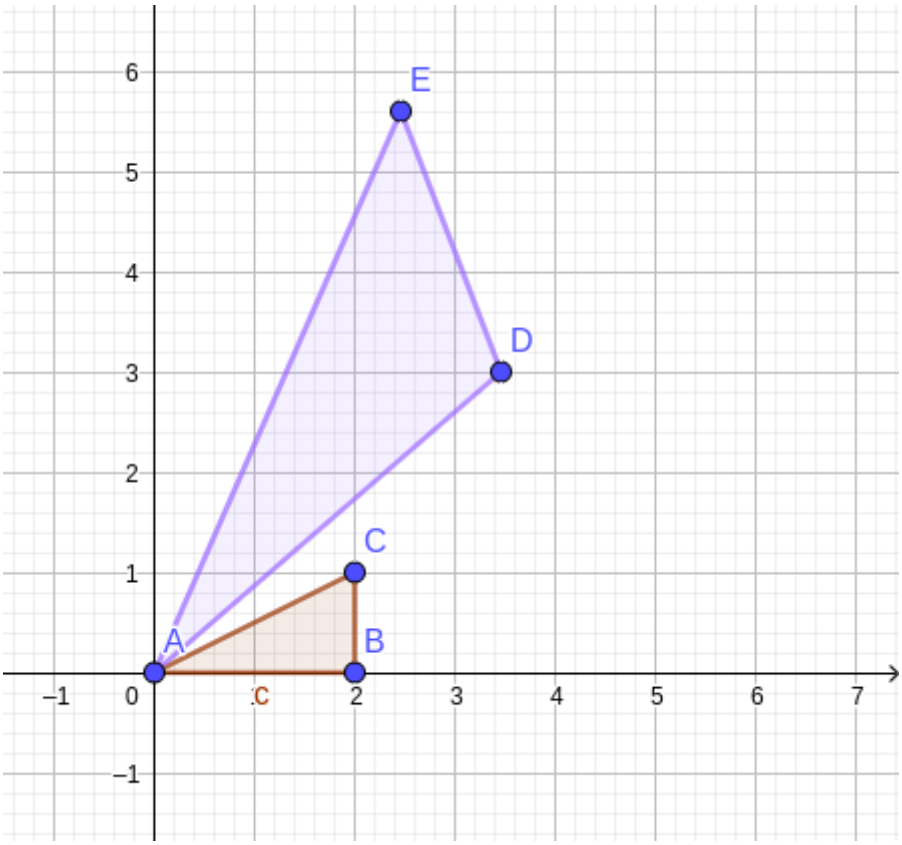



*Ilustración 11: Composición de Transformaciones en un triángulo.*

### 3.3.9. Traslación

Una traslación de un vector $X$ es un desplazamiento fijo $K$ a una dirección, sentido y longitud. Es la transformación:

$$X' = T(X) = X + K \tag{9a}$$

Realizar el camino inverso de una traslación es posible. Para ello, basta con sumar el opuesto al vector de desplazamiento $K$. Esto es:

$$X = X' + (-K) = X' - K \tag{9b}$$

Para esta transformación, se habla de opuesto en lugar de inverso: la operación para regresar al punto anterior es la suma en lugar del producto, y su elemento neutro es la matriz nula en lugar de la matriz identidad.

#### Traslaciones en el plano

Una Traslación en el plano de un vector $X$ en la dirección, sentido y longitud fija $K$ es la transformación dada por:

$$T(X)=\begin{pmatrix}x\\y\end{pmatrix}+\begin{pmatrix}k_1\\k_2\end{pmatrix}=\begin{pmatrix}x+k_1\\y+k_2\end{pmatrix}=X'$$ (9c)

con:

$$X=\begin{pmatrix}x\\y\end{pmatrix}$$

el vector posición del punto a transformar,

$$K=\begin{pmatrix}k_1\\k_2\end{pmatrix}$$

el vector de desplazamiento fijo con componente $k_1$ en $x$ y $k_2$ en $y$.

**Ejemplo 11:** Sea el triángulo $X$ definido por sus vértices: $X=\{A(0;0), B(2;0), C(2;1)\}$. Obtener $X'$: la traslación de $X$, según el vector de desplazamiento $K$ = (1,2).

Aplicando la expresión 9c:

$$X'=T\begin{pmatrix}x\\y\end{pmatrix}=\begin{pmatrix}0&2&2\\0&0&1\end{pmatrix}+\begin{pmatrix}1&1&1\\2&2&2\end{pmatrix}=\begin{pmatrix}1&3&3\\2&2&3\end{pmatrix}$$

Finalmente: $X'=\{D(1;2), E(3;2), F(3;3)\}$.

La ilustración 12 grafica a la transformación del ejemplo 11.

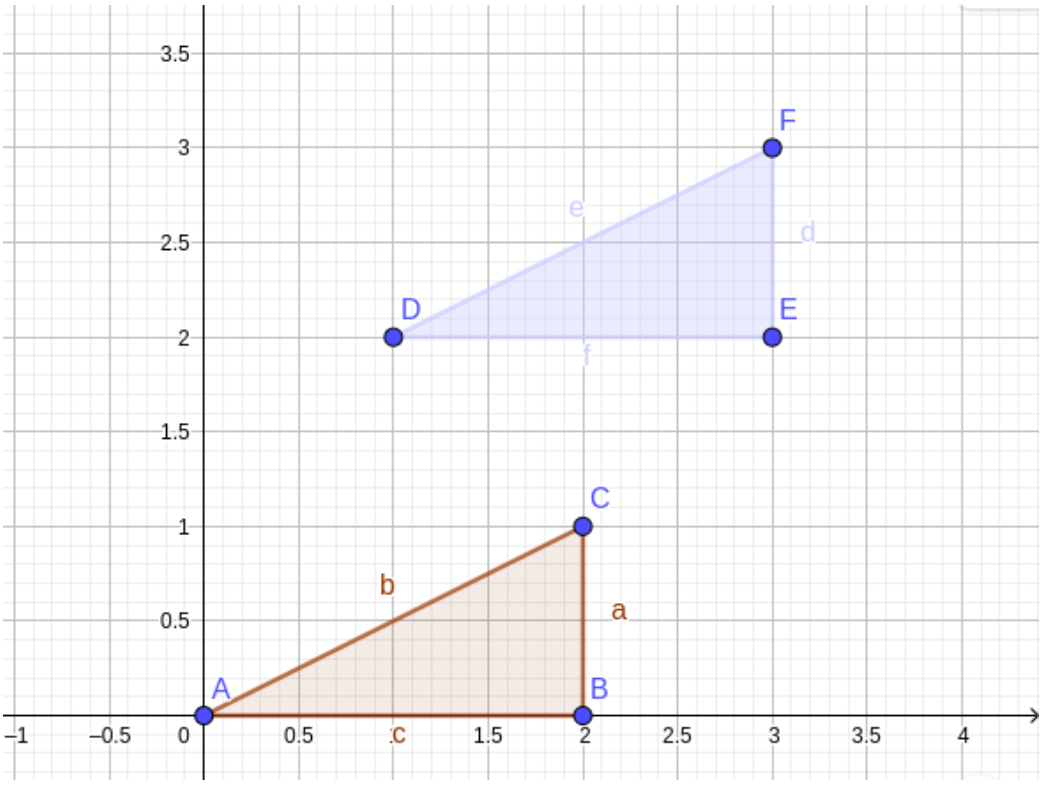



*Ilustración 12: Traslación de un triángulo en el plano*

## Opuesto de la traslación en el plano

Si $K = (k_1 ; k_2)$ es el vector de desplazamiento fijo en el plano, su opuesto es: $K' = (-k_1 ; -k_2)$. Es fácil ver que $\overline{K}'$ es su opuesto:

$$X = X' + K' = \begin{pmatrix} x + k_1 \\ y + k_2 \end{pmatrix} + \begin{pmatrix} -k_1 \\ -k_2 \end{pmatrix} = \begin{pmatrix} x \\ y \end{pmatrix}$$

## Traslaciones en el espacio

Una traslación en el espacio de un vector $\overline{X}$ en la dirección, sentido y longitud fija $K$ es la transformación dada por:

$$X' = T \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} + \begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix} = \begin{pmatrix} x + k_1 \\ y + k_2 \\ z + k_3 \end{pmatrix} \qquad (9d)$$

con:

$$X = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

el vector posición del punto a transformar,

$$K=\begin{pmatrix} k_1 \\ k_2 \\ k_3 \end{pmatrix}$$

el vector de desplazamiento fijo con componente $k_1$ en $x$, $k_2$ en $y$ y $k_3$ en $z$.

**Ejemplo 12:** Sea el prisma $X$ definido por sus vértices: $X=\{A(0;0;0), B(1;0;0), C(1;2;0), D(0;2;0), E(0;0;1), F(1;0;1), G(1;2;1), H(0;2;1)\}$. Obtener $X'$: la traslación de $X$, según el vector $K$ = (1,2,1).

Aplicando la expresión 9d:

$$X'=T\begin{pmatrix} x \\ y \\ z \end{pmatrix}=\begin{pmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}+\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}=\begin{pmatrix} 1 & 2 & 2 & 1 & 1 & 2 & 2 & 1 \\ 2 & 2 & 4 & 4 & 2 & 2 & 4 & 4 \\ 1 & 1 & 1 & 1 & 2 & 2 & 2 & 2 \end{pmatrix}$$

Finalmente $X'=\{N(1;2;1), L(2;2;1), M(2;4;1), O(1;4;1), P(1;2;2), I(2;2;2), J(2;4;2), K(1;4;2)\}$

La ilustración 13 grafica a la transformación del ejemplo 12.

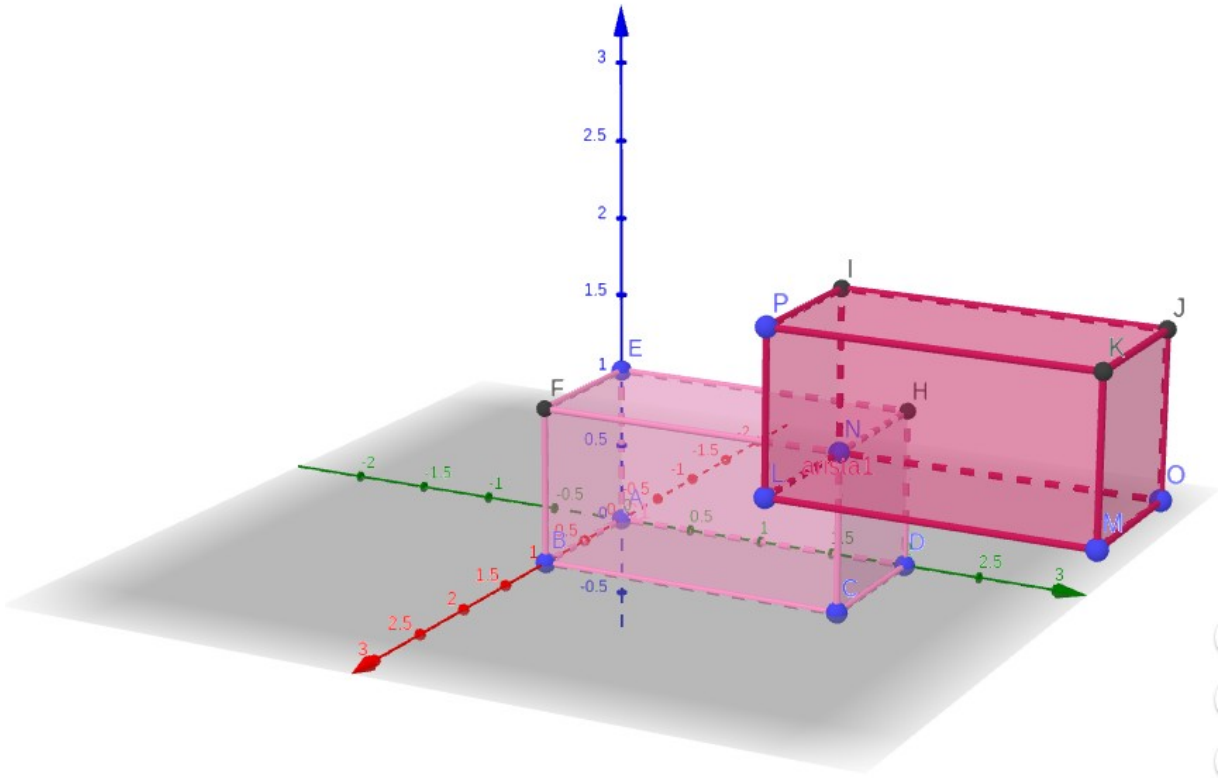



*Ilustración 13: Traslación de un prisma en el espacio.*

## Opuesto de la Traslación en el Espacio

Si $K$ = ($k_1$ ; $k_2$ ; $k_3$) es el vector de desplazamiento fijo en el plano, su opuesto es: $K'$ = ($-k_1$ ; $-k_2$ ; $-k_3$). Es fácil ver que $K'$ es su opuesto:

$$X = X' + K' = \begin{pmatrix} x + k_1 \\ y + k_2 \\ z + k_3 \end{pmatrix} + \begin{pmatrix} -k_1 \\ -k_2 \\ -k_3 \end{pmatrix} = \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

Observación Importante:

$$Si\ h \neq 0 \vee k \neq 0 \Rightarrow T\ no\ es\ \ TL\ ya\ que\ \ T\begin{pmatrix} 0 \\ 0 \end{pmatrix} = \begin{pmatrix} h \\ k \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad (9e)$$

Es decir, las traslaciones no son TL, entonces, no puede presentarse una matriz asociada que pre-multiplicada por la matriz de los vectores posición de los puntos logre la traslación. Ello se soluciona mediante el uso de transformaciones en coordenadas homogéneas, según se muestra a continuación.

## 3.3.10. Transformaciones en coordenadas homogéneas

Una matriz de transformación puede resultar de la composición entre dos o más transformaciones lineales geométricas básicas.

Las traslaciones no son transformaciones lineales geométricas por no ser TL. Sin embargo, es posible representar la composición de las transformaciones lineales geométricas básicas con traslaciones en el espacio bidimensional por medio de una única TL tridimensional del plano z = 1. De manera similar, es posible representar las mencionadas transformaciones en el espacio tridimensional por medio de una única TL cuatridimensional en el hiper-plano w = 1. A estas transformaciones se las llama *transformaciones geométricas en coordenadas homogéneas* (Lengyel, 2011).

Las matrices de las transformaciones en coordenadas homogéneas se obtienen extendiendo una coordenada a la matriz de transformación original, agregando en la última columna al vector de traslación y completando la matriz y los vectores con 0 y 1 según corresponde.

### Transformaciones en Coordenadas Homogéneas en el Plano

Se representa la composición de las transformaciones geométricas básicas junto con las traslaciones en el plano mediante una única TL tridimensional en el plano z = 1, según se muestra a continuación:

$$T\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} & h \\ t_{21} & t_{22} & k \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ 1 \end{pmatrix} \quad (10a)$$

$$donde \begin{pmatrix} t_{11} & t_{12} \\ t_{21} & t_{22} \end{pmatrix} es\ \ TL\ \ compuesta\ y\ \ \begin{pmatrix} h \\ k \end{pmatrix}\ \ coordenadas\ \ de\ \ una\ \ traslación\ \ de\ \ h, k. \quad (10b)$$

**Ejemplo 13:** Sea el triángulo $X$ definido por sus vértices: $X$={$A$(0;0), $B$(2;0), $C$(2;1)}. Realizar una rotación de 30° en sentido antihorario. Luego, sobre el resultado realizar un escalado que estire sus dimensiones al doble en $x$ y al triple en $y$. Finalmente, al resultado obtenido trasladarlo 2 unidad en $x$ y 1 en $y$.

Observar que la transformación presentada en este ejemplo es similar a la transformación presentada en el ejemplo 10, pero difiere en una traslación. Por ello, puede esperarse que:

- El resultado de la transformación podría calcularse sumando el vector de traslación a cada uno de los vectores resultado del ejemplo 10.
- La matriz asociada a la TL que se necesita para el armado de la matriz en coordenadas homogéneas es la misma que la matriz asociada a la TL del ejemplo 10 y se obtiene calculando $E.R$.

Aplicando la expresión 10a:

$$T\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} & h \\ t_{21} & t_{22} & k \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & h \\ 0 & 1 & k \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} e_x & 0 & 0 \\ 0 & e_y & 0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} \cos(\alpha) & -sen(\alpha) & 0 \\ sen(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad (10c)$$

con:

$$E.R = \begin{pmatrix} 2 & 0 \\ 0 & 3 \end{pmatrix} \cdot \begin{pmatrix} \frac{\sqrt{3}}{2} & \frac{-1}{2} \\ \frac{1}{2} & \frac{\sqrt{3}}{2} \end{pmatrix} = \begin{pmatrix} \sqrt{3} & -1 \\ \frac{3}{2} & \frac{3}{2}\sqrt{3} \end{pmatrix}$$

entonces $T$ es:

$$T = \begin{pmatrix} \sqrt{3} & -1 & 2 \\ \frac{3}{2} & \frac{3}{2}\sqrt{3} & 1 \\ 0 & 0 & 1 \end{pmatrix}$$

la matriz de transformación en coordenadas homogéneas. Luego aplicando $T$ a $X$:

$$T\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} \sqrt{3} & -1 & 2 \\ \frac{3}{2} & \frac{3}{2}\sqrt{3} & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \Rightarrow T\begin{pmatrix} 0 & 2 & 2 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} \sqrt{3} & -1 & 2 \\ \frac{3}{2} & \frac{3}{2}\sqrt{3} & 1 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 0 & 2 & 2 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 2 & 2\sqrt{3}+2 & 2\sqrt{3}+1 \\ 1 & 4 & \frac{3}{2}\sqrt{3}+4 \\ 1 & 1 & 1 \end{pmatrix} = X'$$

Luego:

$$X' = \{D(2;1), E(2\sqrt{3}+2;4), F(2\sqrt{3}+1;\frac{3}{2}\sqrt{3}+4)\}$$

La Ilustración 14 grafica el ejemplo 13:

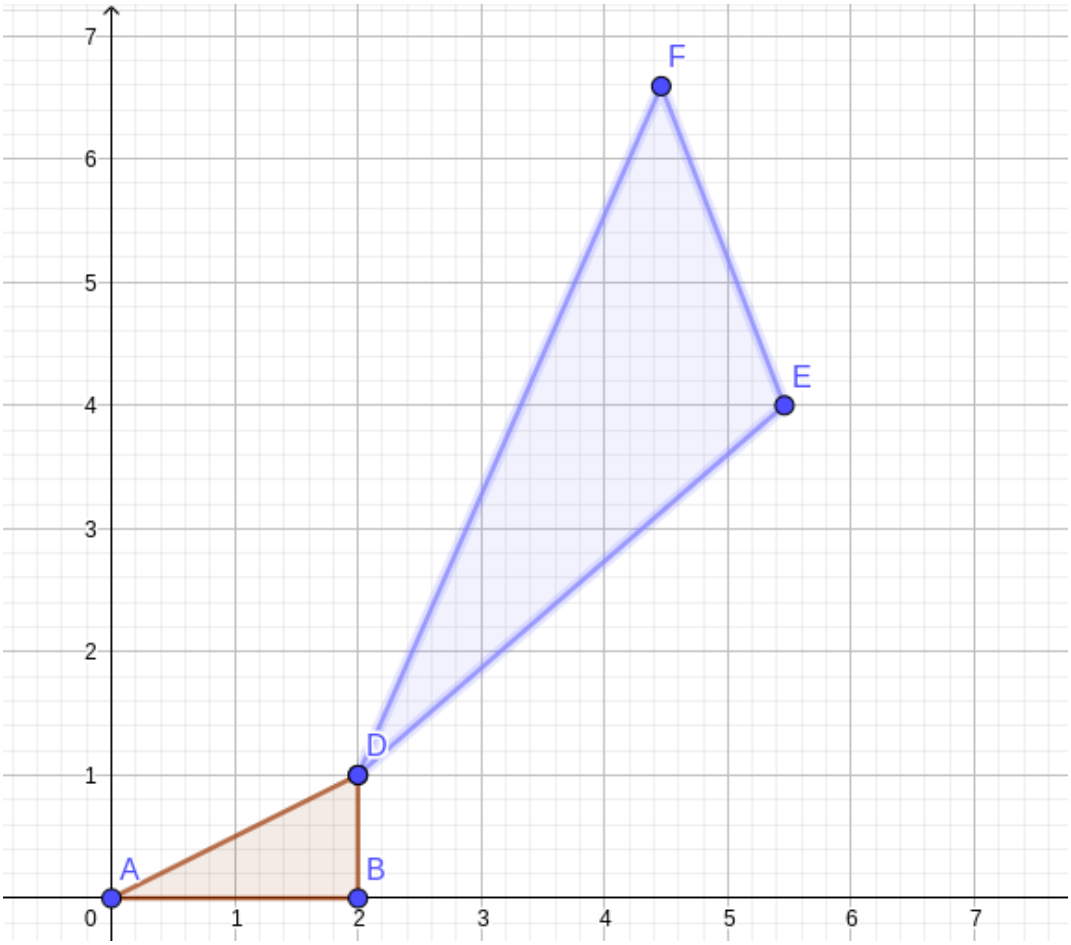



*Ilustración 14: Transformación de un triangulo usando coordenadas homogéneas.*

## Transformaciones en coordenadas homogéneas en el espacio

Se representa la composición de las transformaciones geométricas básicas junto con las traslaciones en el espacio mediante una única TL tridimensional en el hiper-plano $w = 1$, según se muestra a continuación:

$$T\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} = \begin{pmatrix} t_{11} & t_{12} & t_{13} & h \\ t_{21} & t_{22} & t_{23} & k \\ t_{31} & t_{32} & t_{33} & l \\ 0 & 0 & 0 & 1 \end{pmatrix} . \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \tag{10d}$$

$$donde: \begin{pmatrix} t_{11} & t_{12} & t_{13} \\ t_{21} & t_{22} & t_{23} \\ t_{31} & t_{32} & t_{33} \end{pmatrix} es\ TL\ compuesta\ y \begin{pmatrix} h \\ k \\ l \end{pmatrix} coordenadas\ de\ una\ traslación\ de\ h, k, l \tag{10e}$$

A estas transformaciones se las denomina también Transformaciones 4D o Transformaciones Cuatri-dimensionales (Lengyel, 2011).

**Ejemplo 14:** Sea el prisma $X$ definido por sus vértices: $X=\{A(0;0;0), B(1;0;0), C(1;2;0), D(0;2;0), E(0;0;1), F(1;0;1), G(1;2;1), H(0;2;1)\}$. Realizar una rotación de 90° en alrededor del eje x en sentido antihorario. Luego, sobre el resultado realizar un escalado que estire sus dimensiones al doble en *x*, al triple en *y* y se mantenga igual en z. Finalmente, al resultado obtenido, trasladarlo 2 unidad en $x$, 1 en $y$ y 2 en $z$.

Aplicando 10d:

$$T\begin{pmatrix}x\\y\\z\\1\end{pmatrix}=\begin{pmatrix}1&0&0&2\\0&1&0&1\\0&0&1&2\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}2&0&0&0\\0&3&0&0\\0&0&1&0\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}1&0&0&0\\0&0&-1&0\\0&1&0&0\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}x\\y\\1\end{pmatrix}=\begin{pmatrix}2&0&0&2\\0&0&-1&1\\0&3&0&2\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}x\\y\\1\end{pmatrix}$$

con:

$$T=\begin{pmatrix}2&0&0&2\\0&0&-1&1\\0&3&0&2\\0&0&0&1\end{pmatrix}$$

la matriz de transformación en coordenadas homogéneas. Luego aplicando $T$ a $X$:

$$T(X)=\begin{pmatrix}2&0&0&2\\0&0&-1&1\\0&3&0&2\\0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}0&1&1&0&0&1&1&0\\0&0&2&2&0&0&2&2\\0&0&0&0&1&1&1&1\\1&1&1&1&1&1&1&1\end{pmatrix}=\begin{pmatrix}2&4&4&2&2&4&4&2\\1&1&1&1&0&0&0&0\\2&2&8&8&2&2&8&8\\1&1&1&1&1&1&1&1\end{pmatrix}$$

Luego: X'={Q(2;1;2), R(4;1;2), J(4;1;8), K(2;1;8), O(2;0;2), P(4;0;2), I(4;0;8), S(2;0;8)}

La Ilustración 15 grafica el ejemplo 14.

## 3.3.11. Rotaciones con cuaterniones

### Introducción

Los cuaterniones son números hipercomplejos de cuatro componentes. A pesar de poseer propiedades algebraicas y operacionales muy buenas, poco después de su invención fueron opacados por el desarrollo del cálculo vectorial. En estos últimos años crece notablemente la cantidad de implementaciones de cuaterniones en diversas disciplinas atraídas por notables ventajas relacionadas con la simplicidad, eficiencia y características algebraicas. Su uso principalmente se da en el cálculo de rotaciones en el espacio con aplicaciones en navegación aeroespacial, articulaciones de brazos robóticos, visión robótica y gráfica 3D. Su ventaja radica en que su simpleza redunda en reducciones en tiempos de cálculo de rotaciones de hasta 75% en comparación con el cálculo matricial (Kamlofsky y Bergamini, 2015).

### Álgebra básica de cuaterniones

Un cuaternión es un número hipercomplejo: $q=w+xi+yj+zk$, donde $i,j,k$ son unidades imaginarias y $\bar{v} = (x,y,z)$ su vector asociado. Vectorialmente: $q = (w,\bar{v})$. Su notación cartesiana es: $q = (w,x,y,z)$.

La suma y resta de cuaterniones se define componente a componente.

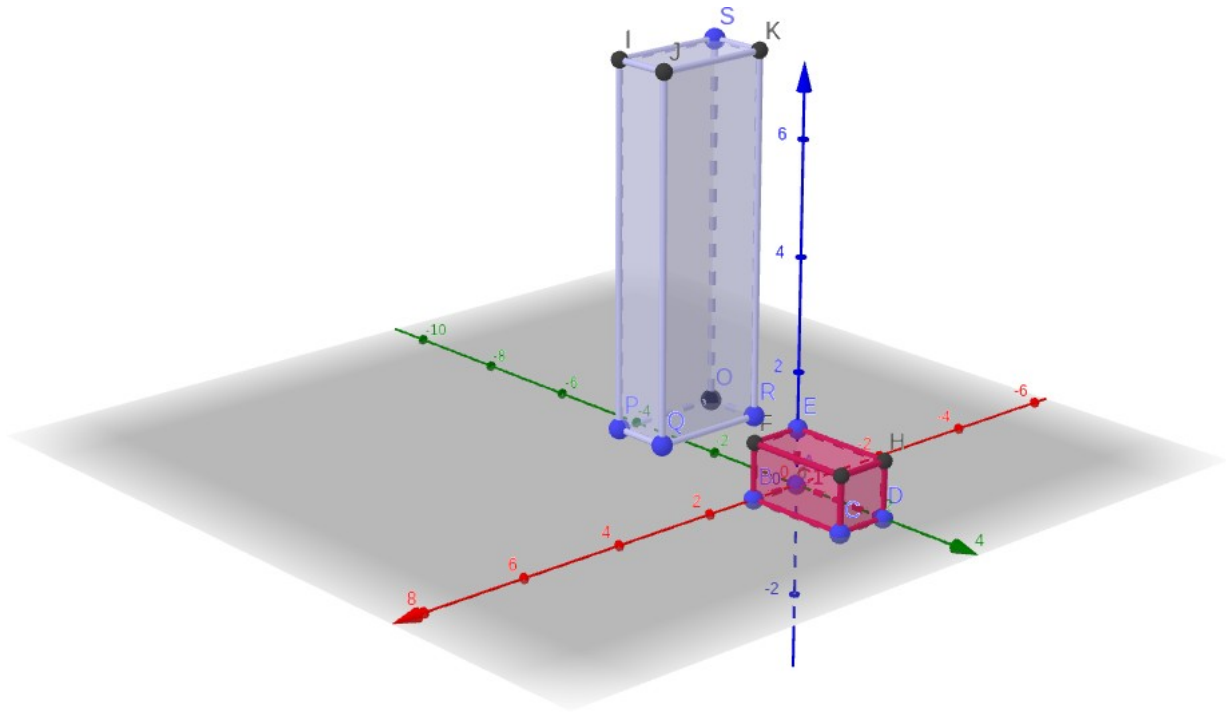



*Ilustración 15: Transformación en coordenadas homogéneas de un prisma en el espacio*

El producto de dos cuaterniones $q_1=(w_1,x_1,y_1,z_1)$ y $q_2=(w_2,x_2,y_2,z_2)$ es:

$$q_3=q_1.q_2=(w_3,x_3,y_3,z_3) \quad (11a)$$

donde:

$$w_3 = w_1.w_2 - x_1.x_2 - y_1.y_2 - z_1.z_2 \quad (11b)$$

$$x_3 = w_1.x_2 + x_1.w_2 + y_1.z_2 - z_1.y_2 \quad (11c)$$

$$y_3 = w_1.y_2 - x_1.z_2 + y_1.w_2 + z_1.x_2 \quad (11d)$$

$$z_3 = - w_1.z_2 + x_1.y_2 - y_1.x_2 + z_1.w_2 \quad (11e)$$

Como puede comprobarse fácilmente, este producto no es conmutativo.

Sea el cuaternión $q=(w,x,y,z)$

Su conjugado es:

$$q^*=(w,-x,-y,-z) \quad (12)$$

Su norma es:

$$|q|=\sqrt{w^2+x^2+y^2+z^2} \quad (13)$$

Puede denotarse la forma trigonométrica del cuaternión:

$$q=|q|.\left(\cos\frac{\alpha}{2},\breve{v}\,.\,sen\frac{\alpha}{2}\right) \tag{14a}$$

con:

$$\breve{v}=\frac{\bar{v}}{|\bar{v}|},\ \ \bar{v}=(x\,;y\,;z) \tag{14b}$$

y

$$\alpha=2\arccos\left(\frac{w}{|q|}\right) \tag{14c}$$

Con *a* : su ángulo de rotación asociado, y $\bar{v}$ : vector director del eje de rotación.

Pero si:

$$\breve{q}=\frac{q}{|q|}\Rightarrow\breve{q}=\left(\cos\frac{\alpha}{2},\breve{v}\,.\,sen\frac{\alpha}{2}\right) \tag{15}$$

que es una expresión más sencilla para operar.

## Cálculo de rotaciones

Un cuaternión en notación trigonométrica trae asociados un ángulo y un vector. Entonces, realizar una rotación de un punto *P(a,b,c)* un ángulo *a* en sentido anti-horario alrededor del vector $\bar{v}$ = (*x;y;z*):

$$P' = q\,.\,P\,.\,q^* \tag{16}$$

Para la implementación de la ecuación (16), el punto se expresa como cuaternión con parte real nula. Así también, su resultado es un punto en el espacio con notación de cuaternión, con componente real nula.

**Ejemplo 15:** Calcular la rotación del punto $P(6;2;5)$ alrededor del vector $\bar{v}$ = $(1;2;1)$ un ángulo de 30° en sentido antihorario.

*Nota: A los fines de presentar sencillez de cálculo, los resultados parciales y números irracionales se presentan redondeados a tres decimales.*

Así:

cos(15°) = 0,966; sen(15°) = 0,259

$$\breve{v}=(0{,}408\,;0{,}816\,;0{,}408)$$

Usando la expresión obtenida en (16) se obtienen los cuaterniones unitarios:

*q* = (0,966 ; 0,106 ; 0,211 ; 0,106)

*q**= (0,966 ; -0,106 ; -0,211 ; -0,106)

La expresión (16) es entonces:

$P'=q.P.q^*$= (0,966 ; 0,106 ; 0,211 ; 0,106).(0 ; 6 ; 2 ; 5).(0,966 ; -0,106 ; -0,211 ; -0,106)

$$P' = (0{,}000\,;\, 7{,}164\,;\, 2{,}606\,;\, 2{,}624)$$

La ilustración 16 muestra una imagen en 3D del punto $P$, al vector $\bar{v}$ y al punto $P'$: resultado de rotar al punto $P$ un ángulo a alrededor de $\bar{v}$.

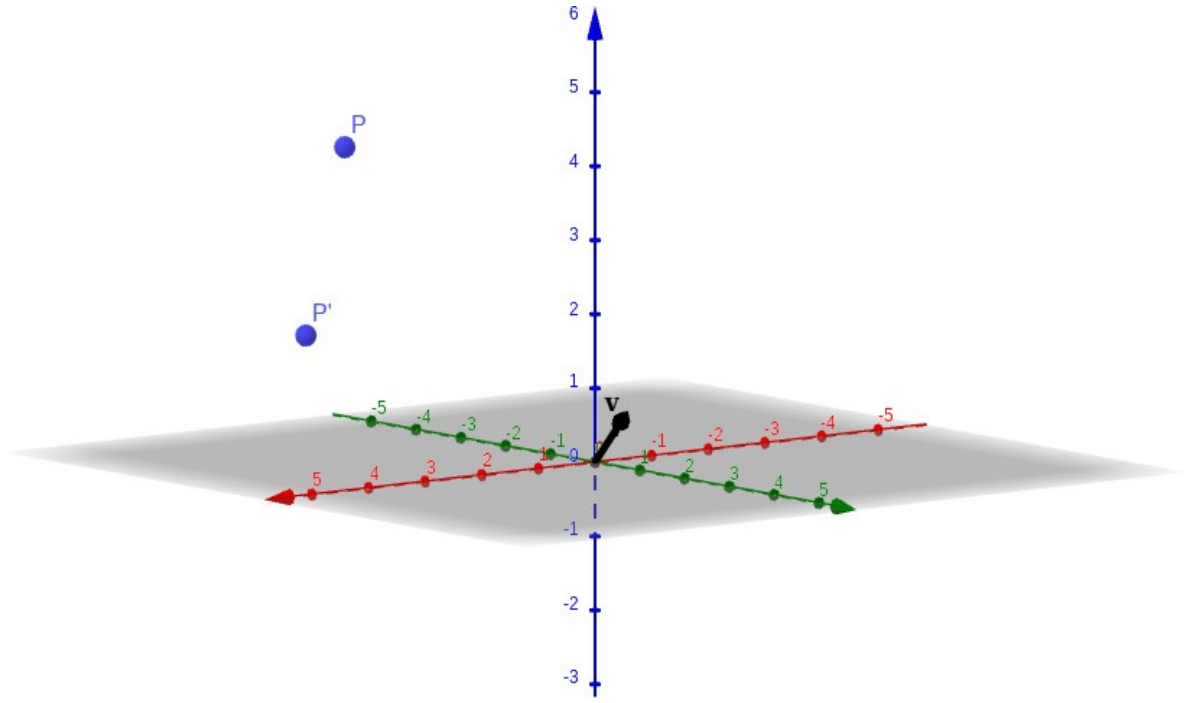



*Ilustración 16: Vista 3D de la rotación del punto P alrededor de un vector, un ángulo.*

*Nota: Las ilustraciones 5 a 16 fueron realizadas utilizando el software Geogebra*[4]

## 3.4. Reconocimiento de Objetos en Imágenes

En esta sección se presentan dos enfoques distintos comúnmente usados cuyo objetivo es el mismo: obtener información de la imagen digital para poder reconocer objetos dentro de las mismas. Un enfoque basado en la Topología Digital y otro basado en el uso de técnicas de Inteligencia Artificial (IA).

El enfoque de la topología digital permite extraer propiedades topológicas de un objeto. Es posible determinar sus bordes, los cuales pueden analizarse tanto topológicamente como geométricamente. La algoritmia es muy sencilla y de baja complejidad computacional.

4 Sitio Oficial de Geogebra: https://www.geogebra.org/

El enfoque de IA hace uso de diferentes técnicas. En especial, se enfoca en el uso de redes neuronales. Su uso en el reconocimiento de objetos imágenes está ampliamente difundido y muy aceptado.

Dado que este trabajo se basa en el primero de los enfoques mencionados, se lo presenta con mayor profundidad.

## 3.4.1. Enfoque Basado en Topología Digital

### Introducción

Este apartado es una introducción a los conceptos de Topología Digital, los cuales se aplican luego de realizada la segmentación de la imagen y permiten atribuirle a los objetos características topológicas. Se basa principalmente en conceptos del trabajo Rosenfeld (1979). Más específicamente, en este trabajo presentan conceptos y propiedades de la Topología Digital y su aplicación, entre los que se destaca el algoritmo BF4/8 para la obtención de bordes presentado por Rosenfeld (1979).

Usando el concepto de Topología Digital se permite tratar a ciertos píxeles de una imagen como un conjunto conexo, con borde o esqueleto, con o sin agujeros, y demás características topológicas. Se pretende aplicar propiedades topológicas del plano real a un subconjunto del plano digital: la imagen digital y con estos algoritmos ayudar al desarrollo de aplicaciones de visión artificial.

### La Topología de una imagen digital

**El Plano digital:**

El plano digital $Z^2$ es el conjunto formado por los pares ordenados de números enteros.

**Vecindades:**

Dado un punto $P(x,y)$ del plano digital $Z^2$, pueden definirse a sus vecinos como:

$$4-vecinos=\{(x\pm 1,y),(x,y\pm 1)\} \quad (17a)$$

$$8-vecinos=\{(x\pm 1,y),(x,y\pm 1),(x+1,y\pm 1),(x-1,y\pm 1)\} \quad (17b)$$

Entonces, la *4-vecindad* es el conjunto de 4-vecinos y la *8-vecindad* es el conjunto de 8-vecinos.

**Topología del plano digital:**

Se define:

$$U_{(p)}=\begin{cases}\{P\}\ si\ x+y\ es\ impar\\ \{Q:Q\ es\ 4-vecino\ de\ P\}\ si\ x+y\ es\ par\end{cases} \quad (18)$$

Sea $B=\{U_{(P)}: \in Z^2\}$, $B$ es base para una topología en $Z^2$ (Kamlofsky, 2011; Rosenfeld, 1979). Así, al plano digital se le asigna una topología $\tau$ a partir de definida la base $B$. La topología digital $\tau$ de $Z^2$ es la topología generada por la base $B$.

**Topología de una imagen digital:** Una imagen digital $\pi$ de $M\ x\ N$ píxeles es un subconjunto finito de $Z^2$ tal que:

$$\pi=\{(x,y)\in Z^2 / 0\le x\le N-1\ \wedge\ 0\le y\le M-1\} \quad (19)$$

El borde o frontera de $\pi$ es:

$$Fr(\pi) = \{(x,y) \in \pi / \; x = 0 \vee \; x = N-1 \vee \; y = 0 \vee \; y = M-1\} \tag{20}$$

Sea:

$$B_\pi = \{U_{(P)} \cap \pi : P \in \mathbb{Z}^2\} \tag{21}$$

$B_\pi$ es base. La topología $\tau_\pi$ de la imagen digital $\pi$ es aquella generada por la base $B_\pi$.

## Los objetos de una imagen digital

La segmentación (o binarización) divide a la imagen digital en dos: el subconjunto de los objetos y el subconjunto del fondo de la imagen. Una vez que la imagen ha sido segmentada en subconjuntos, el paso siguiente consiste en establecer sus propiedades y relaciones entre ellos.

**Ejemplo 16:** la ilustración 17 muestra una imagen de un texto: luego de separar las letras del fondo, interesa identificar a cada letra. Algunas propiedades pueden obtenerse fácilmente como el área del subconjunto objeto. Pero como otras propiedades son netamente topológicas, se necesitan conceptos topológicos. Estas propiedades son útiles ya que, una vez que el subconjunto fue identificado (texto), se necesita separarlo en regiones conexas que se correspondan con cada letra.

Se considera que una imagen digital $\pi$ está segmentada en dos conjuntos: el conjunto $S$ que contiene a los objetos de la imagen en sí, y su complemento $S^C$ que representa el fondo. Se asume que $S \cap Fr(\pi) = \emptyset$. La frontera de $S$ es el conjunto de puntos que tienen 4-vecinos en $S$. Con $S = \{S_1, S_2, \ldots, S_j\}$, $j \in$ N, cada $S_i$ es un objeto dentro de la imagen p con $i \in [1, 2, \ldots, j]$.

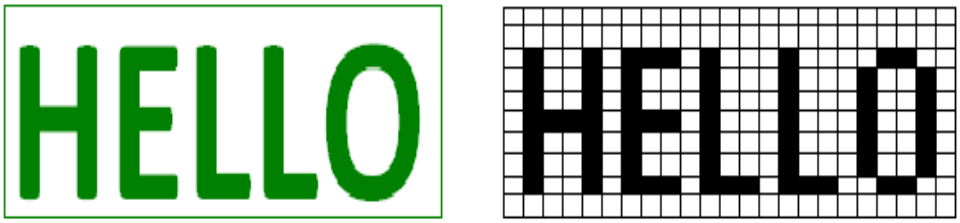



*Ilustración 17: Imagen analógica y una imagen digitalizada y segmentada de una palabra.*

## Conectividad

**Caminos:** Sean $P$ y $Q$ dos puntos de la imagen $\pi$. Un camino de $P$ a Q es una sucesión finita de puntos $P = P_0, P_1, \ldots, P_n, = Q$ pertenecientes a $\pi$ : $P_i$ *es vecino de* $P_{i-1}$ para $1 \leq i \leq n$. Podemos notar que esta definición contiene dos definiciones en una, ya sea si tratamos 4-vecinos u 8-vecinos tendremos 4-caminos u 8-caminos.

**Conjunto conexo**: Un conjunto $S_i$ es conexo, si para todo par de puntos de $S_i$ existe un camino que los una conformado por puntos dentro de $S_i$. Cada $S_i$ es considerado como un objeto en la imagen.

Se trata de trasladar adecuadamente las propiedades topológicas del plano real al plano digital. Para que así funcione, si se considera 4-conectividad para $S$ (con 4-caminos), debe considerarse 8-conectividad (con 8-caminos) para $S^C$. U 8-conectividad para $S$ y 4-conectividad para $S^C$ (Rosenfeld, 1979).

**Componente conexa:** A cada $S_i$ se la llama también "*componente conexa de S*" o simplemente "*componente*".

**Agujeros:** Sea $S \in \pi,\ S \neq \emptyset$ . El fondo de $\pi$ es la única componente conexa de $S^C$ que contiene a $Fr(\pi)$. Las demás componentes de $S^C$, si existen, reciben el nombre de "*agujeros de S*".

**Arcos:** $S \in \pi,\ S \neq \emptyset$ . S es un arco si es conexo y todos, salvo dos de sus puntos (sus extremos) tienen exactamente dos vecinos en $S$, mientras que los dos extremos tienen exactamente uno. Más precisamente, un arco $S$ es un camino $P_0, P_1, \ldots, P_n$ formado por puntos distintos y tal que $P_i$ es vecino de $P_j$ sí y solo sí $i = j \pm 1$. Puede considerarse a un arco como un camino abierto que no se autointersecta.

**Curvas:** $S \in \pi,\ S \neq \emptyset$ $S$ es una curva si es conexo y todos sus puntos tienen exactamente dos vecinos en $S$. Más precisamente, una curva $S$ es un camino $P_0, P_1, \ldots, P_n$ formado por puntos distintos tal que $P_i$ es vecino de $P_j$, y son iguales al inicio y al final. Es decir: $P_0 = P_n$. Puede considerarse a una curva como un camino abierto que se autointersecta al inicio y fin.

**El teorema de la curva de Jordan:** En la topología usual el teorema de la curva de Jordan afirma que una curva simple cerrada $C$ en el plano divide a éste en dos regiones: una acotada, llamada "el interior de $C$" y otra no acotada, llamada "el exterior de $C$", siendo $C$ la frontera común. Se pretende trasladar este resultado al plano digital. En 1979 Rosenfeld lo presentó en el siguiente teorema (Rosenfeld, 1979):

*Sea S una 4-curva en el plano digital . Entonces $S^C$ tiene exactamente dos componentes 8-conexas. Una de ellas es acotada, llamada "el interior de S" y la otra es no acotada, llamada "el exterior de S". Análogamente para 8-curvas.*

Este resultado se encuentra probado en Kong et al. (1991).

Una consecuencia del teorema es que una curva $S$ tiene exactamente un agujero. Otra consecuencia es que si $S$ es una 8-curva, entonces el agujero y el fondo son topológicamente conexos. Hay otro resultado obvio pero útil: Todo punto $P$ de una curva $S \in \pi$ es adyacente (en el sentido de conectividad de $S$) a las dos componentes de $S^C$.

## Extrayendo características de un objeto tras analizar arcos y curvas

Dada una componente, se desea simplificarla sin perder sus características básicas, de modo que sea posible describirla eficientemente.

Puede pensarse en dos enfoques: la obtención de sus bordes para su análisis, o la simplificación de la componente a su esqueleto. En ambos casos, se obtienen representaciones codificadas más simples que el objeto en sí, manteniendo invariantes sus propiedades topológicas. Obtener una codificación compacta de una componente es muy útil para identificar objetos dentro de imágenes digitales.

**Seguimiento de bordes de una componente:** Los bordes pueden presentar características que diferencian ciertos objetos de otros. Dado que los patrones de borde son finitos y pocos (Gross &

Latecki, 1995), más aún, si para almacenar objetos es suficiente con codificar sólo los puntos del borde (Eckhardt & Latecki, 1994), la tarea de identificación de objetos a través de reconocimiento de patrones de borde se ve simplificada.

**Borde de una componente respecto a una componente de su complemento:** Sea $C$ una componente conexa de $S$ y $D$ una componente conexa de $S^C$. El borde de $C$ respecto a $D$ es el conjunto: $C_D = \{P \in C \,/\, P$ tiene al menos un 4-vecino en $D\}$.

**Afinamiento de una componente:** Consiste en reducir una componente a un arco o a una curva mediante la eliminación sucesiva de puntos simples (Rosenfeld, 1979). Si la componente posee un agujero, el proceso de afinamiento simplifica la componente a una curva. Si no posee agujeros, la componente se reduce a un arco.

### 3.4.2. Uso de Técnicas de Inteligencia Artificial con imágenes.

#### Un breve resumen acerca de Inteligencia Artificial

En la industria es común el uso indistinto de los términos IA (Inteligencia Artificial) y ML (iniciales en inglés de Machine Learning). IA se define como la teoría y el desarrollo de sistemas informáticos que pueden realizar tareas que normalmente requieren inteligencia humana, como la percepción visual, el reconocimiento del habla o la toma de decisiones (entre otras). El aprendizaje automático (ML) es una aplicación de IA que proporciona a los sistemas la capacidad de aprender y mejorar automáticamente a partir de la experiencia, sin ser explícitamente programados para ello (Gil y Liska, 2019). En este trabajo se habla genéricamente de IA, pero con estas definiciones, se supone ya, que se logra IA mediante ML.

En general, una solución de IA pretende resolver en tiempo de ejecución, problemas que normalmente no podrían resolverse en corto plazo. Pueden distinguirse, entonces, dos instancias: entrenamiento del modelo y utilización del modelo.

En la etapa de entrenamiento, se ingresa a un algoritmo de IA un conjunto de datos (para ello deben estar en el formato adecuado). Con estos datos, se “entrena el modelo”.

Normalmente, durante este proceso, se divide al conjunto de datos en dos subconjuntos de datos: el subconjunto de datos de entrenamiento y el de prueba. Usualmente, la relación entre entre datos de entrenamiento y prueba suele ser: 80% – 20%. Entonces, al entrenarse los datos, el algoritmo obtiene un modelo, el cual intenta aplicar al subconjunto de prueba, y compara los resultados obtenidos con los presentes en el mismo. Así, es posible elaborar indicadores de calidad de las predicciones del modelo.

En algunos casos (dependiendo del algoritmo utilizado), se presenta también, una planilla, un gráfico ilustrativo, una fórmula u otro tipo de salida que permita describir mejor al fenómeno analizado.

La salida del algoritmo es un archivo binario o caja negra (el modelo) que contiene información obtenida del aprendizaje.

En la instancia de utilización del modelo, se le ingresa al sistema un dato o un conjunto de datos de los cuales se desconoce un atributo, y el algoritmo usa la experiencia adquirida en el momento de aprendizaje para predecir o asignar inteligentemente el valor o atributo desconocido. Para ello, el

algoritmo carga el archivo modelo generado durante la etapa de entrenamiento. De ese modo, utiliza la experiencia aprendida.

## Algunas técnicas y algoritmos de IA

Los algoritmos de ML más usados para lograr soluciones de IA son (Velogig, 2020; Hernandez Orallo et al, 2004):

**Reducción de Dimensionalidad:** dado un conjunto de datos con muchas variables, se destacan aquellas que aportan fuertemente a la variable objetivo o target. Enfoques: Análisis de componentes principales o PCA (según sus siglas en inglés) y entropia de la información de la teoría de la información de Shannon (Shannon, 1948)

**Cluster Analysis:** Segmenta elementos que son similares en algún sentido. Se aplica ampliamente en Marketing y Empresas, que desean segmentar el comportamiento de sus clientes y productos en el mercado y realizar marketing directo y personalizado. Son modelos descriptivos. Ejemplo: K-Means

**Regresión:** Consiste en aprender una función lineal que asigna a cada instancia un valor real. El objetivo en este caso es minimizar el error entre el valor predicho y el valor real. Son modelos predictivos. Ejemplo: Regresión lineal.

**Reglas de asociación:** Son métodos para descubrir nuevas relaciones no explícitas entre variables de una gran base de datos. Son muy utilizadas en supermercados y en publicidad. Son modelos descriptivos. Ejemplo: Apriori.

**Árboles de Decisión:** Se construyen diagramas lógicos en formas de árbol, leyéndose de arriba hacia abajo, hasta llegar a una decisión (hojas).

**Redes Neuronales:** Son modelos que imitan el funcionamiento del cerebro humano. Permite modelar problemas complejos en los que puede haber interacciones no lineal entre las variables. Son modelos predictivos. La Ilustración 18 muestra un modelo de red neuronal.

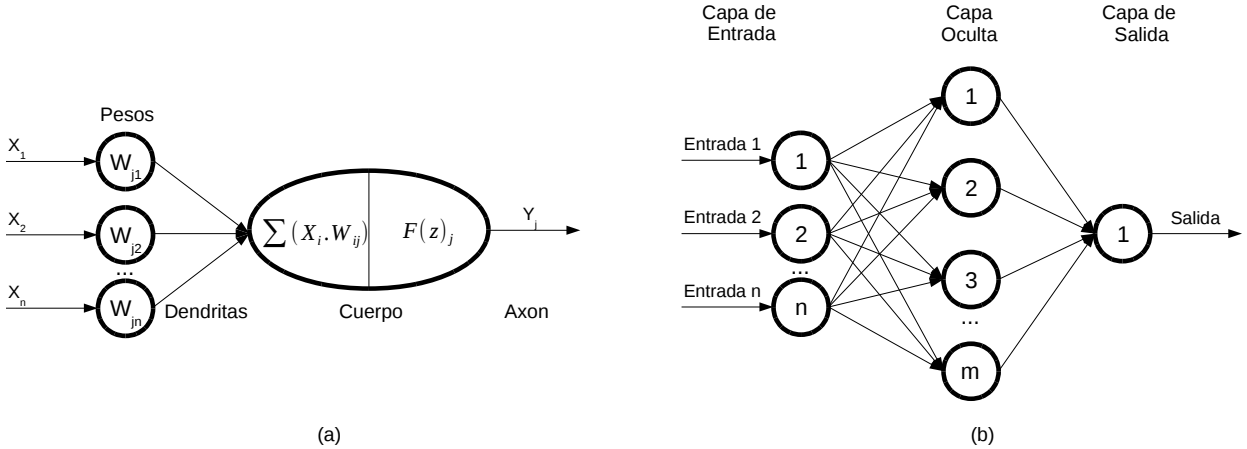



*Ilustración 18: (a) Neurona Artificial o Perceptron. (b) Red Neuronal Artificial.*

## Algunos uso de algoritmos de IA en imágenes

Las técnicas de reconocimiento de patrones estadísticos han logrado, desde el renacimiento de las redes neuronales, un uso generalizado en el procesamiento de imágenes digitales (Egmont-Petersen et al., 2002).

A mediados de los años ochenta, el grupo PDP (Rumelhart et al., 1986), introdujo el algoritmo de aprendizaje de propagación hacia atrás para redes neuronales. Este algoritmo por primera vez hizo posible entrenar una red neuronal no lineal equipada con capas de nodos ocultos.

Desde entonces, las redes neuronales con una o más capas ocultas pueden en teoría ser entrenadas para realizar virtualmente cualquier tarea de regresión o discriminación. Además, no se hacen suposiciones con respecto al tipo de distribución (paramétrica) subyacente de las variables de entrada, que pueden ser nominales, ordinales, reales o cualquier combinación de las mismas.

En 1993 Pal & Pal (1993) habían predicho que las redes neuronales serían ampliamente aplicadas en el procesamiento de imágenes.

## Aplicaciones de redes neuronales en el procesamiento de imágenes

En este apartado, se mencionan algunas instancias de aplicación de redes neuronales, desarrolladas para resolver diferentes problemas del procesamiento de imágenes (Egmont-Petersen et al., 2002):

**Restauración de imagen:** En general, se quiere restaurar una imagen distorsionada. El sistema de adquisición puede introducir ruido, desenfoque de movimiento, distorsión causada por una baja resolución, etc.

**Mejoramiento de imagen:** El objetivo de la mejora de la imagen es amplificar las características perceptivas.

**Detección de bordes:** Se presentan diversas aplicaciones de redes neuronales, entrenadas para comportarse como detectores de bordes.

**Compresión de imagen:** Se presentan diferentes enfoques para la compresión de imágenes.

**Extracción de características:** Se puede ver como un tipo especial de reducción de datos cuyo objetivo es encontrar un subconjunto de variables informativas basadas en datos de imagen. Dado que los datos de imagen son por naturaleza de gran volumen, la extracción de características es a menudo un paso necesario para que la segmentación o el reconocimiento de objetos tenga éxito. Hay un amplio rango de aplicaciones que usan redes neuronales para la extracción de características.

**Segmentación:** El propósito de la segmentación es asignar etiquetas a píxeles individuales. Al tratarse de un problema de clasificación, los enfoques basados en redes neuronales son apropiados.

**Comprensión de la imagen:** Es un área complicada en el procesamiento de imágenes. Combina técnicas de segmentación o reconocimiento de objetos con el conocimiento del contenido de imagen esperado.

**Reconocimiento de objetos:** Consiste en localizar las posiciones y posiblemente las orientaciones y escalas de instancias de objetos en una imagen. También puede ser asignar una etiqueta de clase a un objeto detectado. En la mayoría de las aplicaciones, las redes neuronales han sido entrenadas para localizar objetos individuales basados directamente en datos de píxeles.

## Deep Learning, aprendizaje profundo o reforzado.

Deep Learning hacen uso de la disponibilidad de grandes volúmenes de datos de múltiples orígenes (Big Data) para que los sistemas puedan realizar un aprendizaje mucho más profundo y detallado.

Redes sociales como Flickr[5], Facebook[6], Twitter[7] y diversos servicios digitales "gratuitos" como ser Google[8], Microsoft[9] Azure[10], Amazon[11], obtienen a diario gratuita y voluntariamente de usuarios de todo el mundo, millones de archivos conteniendo información textual o multimedial que les permite lograr modelos mucho más precisos. Por ejemplo: Flickr procesa diariamente unos 3.6TB de información (básicamente en imágenes), mientras que Google procesa globalmente unos 20.000TB de información a diario (Zhang et al., 2018).

Muchos de ellos construyeron enormes plataformas en la nube para desarrollo rápido de soluciones de alcance global. Se ofrecen esquemas de computación en la nube en los tres niveles definidos según NIST[12] (Mell y Grance, 2011): SaaS (software as a service), IaaS (infrastructure as a service) y PaaS (platform as a service),

## Algunos usos de Deep Learning en imágenes

En este apartado se presentan dos ejemplos de aplicaciones de Deep Learning en Imágenes. En particular, se eligió la plataforma de Amazon: AWS[13] (Amazon Web Services). Dentro de esta plataforma se presentan aquí, dos aplicaciones: Textract y Rekognition. La primera de ella extrae los textos presentes en una imagen, mientras que la segunda reconoce objetos en imágenes.

**AWS Textract[14]:**

Es un servicio que detecta y extrae automáticamente textos y datos de documentos escaneados. A diferencia de los simples OCR (siglas del inglés: Optical Character Recognition), Textract extrae el contenido rápida y eficientemente con muy alta confianza, identificando campos y ubicación. Los resultados se pueden presentar en archivos json lo cual permite su tratamiento posterior.

**Ejemplo 17:** La ilustración 19[15] presenta un ejemplo del uso de Textract.

5 Sitio de Flickr: https://www.flickr.com/photos/tags/flicker/
6 Sitio oficial de Facebook: https://about.fb.com/news/
7 Sitio de Twitter: https://twitter.com/
8 Sitio de Google: https://www.google.com/
9 Sitio de Microsoft: https://www.microsoft.com/
10 Sitio de Microsoft Azure: https://azure.microsoft.com/es-es/
11 Sitio de Amazon: https://www.amazon.com/
12 Sitio de NIST: https://www.nist.gov/
13 Sitio de AWS en español: https://aws.amazon.com/es/
14 AWS Textract en español: https://aws.amazon.com/es/textract/
15 Origen de la Imagen: https://spendmatters.com/2018/12/04/extinction-event-amazon-textract-has-just-killed-the-ocr-industry-whos-next-and-whos-safe/

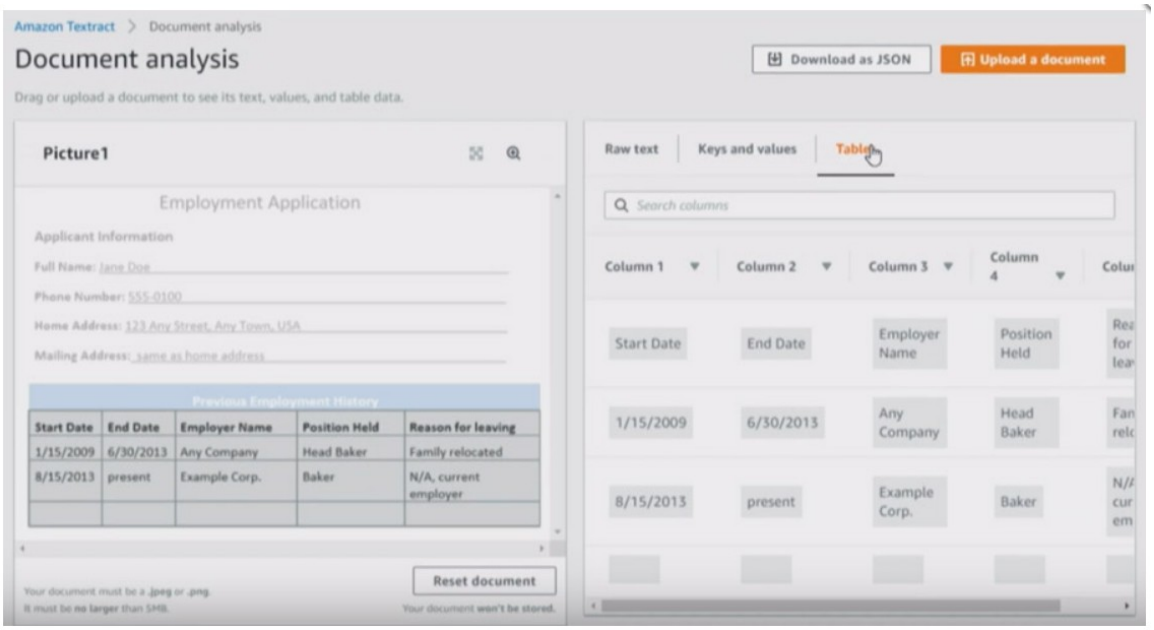



*Ilustración 19: Ejemplo de uso de AWS Textract sobre un formulario.*

**AWS Rekognition:**

Es una aplicación de Machine Learning de AWS que permite reconocer en una escena (presentada desde una foto) objetos y sus atributos y características en imágenes, rápidamente.

**Ejemplo 18:** La ilustración 20[16] presenta un ejemplo del uso de Rekognition.

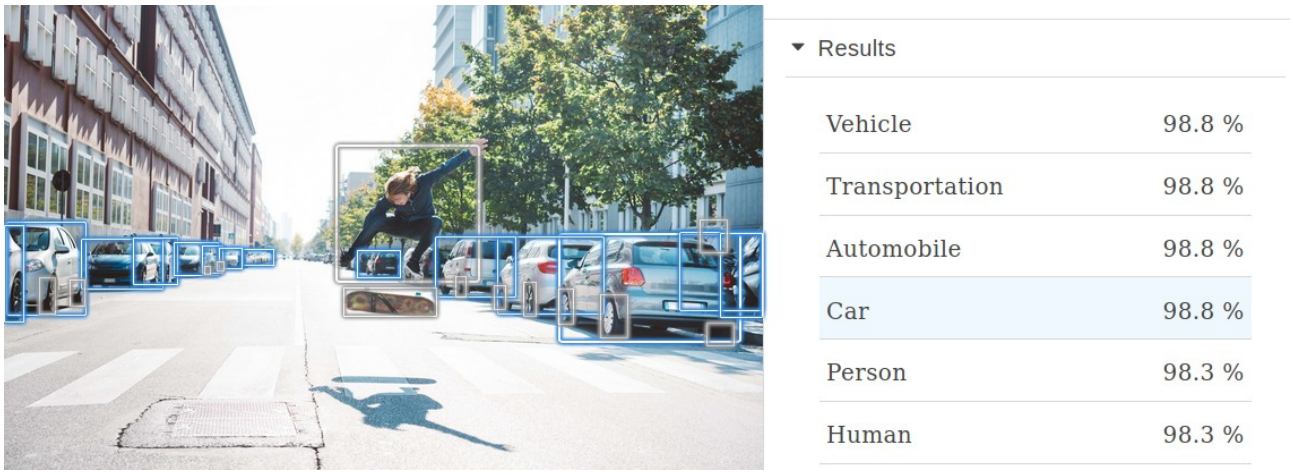



*Ilustración 20: Ejemplo de uso de AWS Rekognition*

16 AWS Rekognition: https://us-east-2.console.aws.amazon.com/rekognition/home?region=us-east-2#/label-detection

## 3.5. Modelado tridimensional

### 3.5.1. Calibración

Para realizar la reconstrucción tridimensional de una escena a partir de múltiples imágenes mediante estereoscopía, éstas deben calibrarse. Para ello, la matriz fundamental es un concepto clave, ya que con la información geométrica disponible, permite obtener la geometría epipolar de la escena a partir de imágenes no calibradas (Bergamini et al., 2016b).

Si se proyecta un punto $P$ de un sistema de referencia tridimensional en la imagen izquierda como $P_L$ y en la derecha como $P_R$, como se muestra en la ilustración 21, los vectores posición del punto de imagen en el mismo sistema referencial satisfacen la ecuación:

$$\overline{P}_L \,.\, \overline{b} \times \overline{P}_R = 0 \qquad (22)$$

donde $b$ es el vector de desplazamiento entre ambas cámaras: $b=(x_C\ ;\ y_C\ ;\ z_C)^{\mathrm{T}}$, o bien, las coordenadas de la cámara derecha en el sistema de referencia 3D, si el sistema de referencia está situado en la cámara izquierda. Los vectores $\overline{P}_L$, $\overline{P}_R$ y $\overline{b}$ son coplanares, el producto mixto es cero (ecuación 22). Usando la matriz anti-simétrica $B$:

$$B=\begin{pmatrix} 0 & -Z_c & Y_c \\ Z_c & 0 & -X_c \\ -Y_c & X_c & 0 \end{pmatrix} \qquad (23)$$

Con esta matriz, que reproduce el producto vectorial, la ecuación 22 puede expresarse: $P_L \,.\, B \,.\, P_R = 0$.

Sean $W_L$ y $W_R$ las coordenadas expresadas en píxeles en las imágenes de los puntos $P_L$ y $P_R$ expresados en unidades de longitud, relacionados mediante la matriz de calibración $C$ como sigue:

$$W=\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}=\begin{pmatrix} \alpha & 0 & u_0 \\ 0 & \beta & v_0 \\ 0 & 0 & 1 \end{pmatrix}\cdot\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}=C\,.\,V \qquad (24a)$$

o bien:

$$V=\begin{pmatrix} x \\ y \\ 1 \end{pmatrix}=\begin{pmatrix} \frac{1}{\alpha} & 0 & \frac{-u_0}{\alpha} \\ 0 & \frac{1}{\beta} & \frac{-v_0}{\beta} \\ 0 & 0 & 1 \end{pmatrix}\cdot\begin{pmatrix} u \\ v \\ 1 \end{pmatrix}=C^{-1}\,.\,W \qquad (24b)$$

donde $W = (u\ ;\ v\ ;\ 1)^{\mathrm{T}}$ presenta las coordenadas de los puntos en las imágenes, $W_0 = (u_0\ ;\ v_0\ ;\ 1)^{\mathrm{T}}$ el punto principal en la imagen, $\alpha$ y $\beta$ distancias focales, $V = (x\ ;\ y\ ;\ 1)^{\mathrm{T}}$ coordenadas del punto en unidades de longitud.

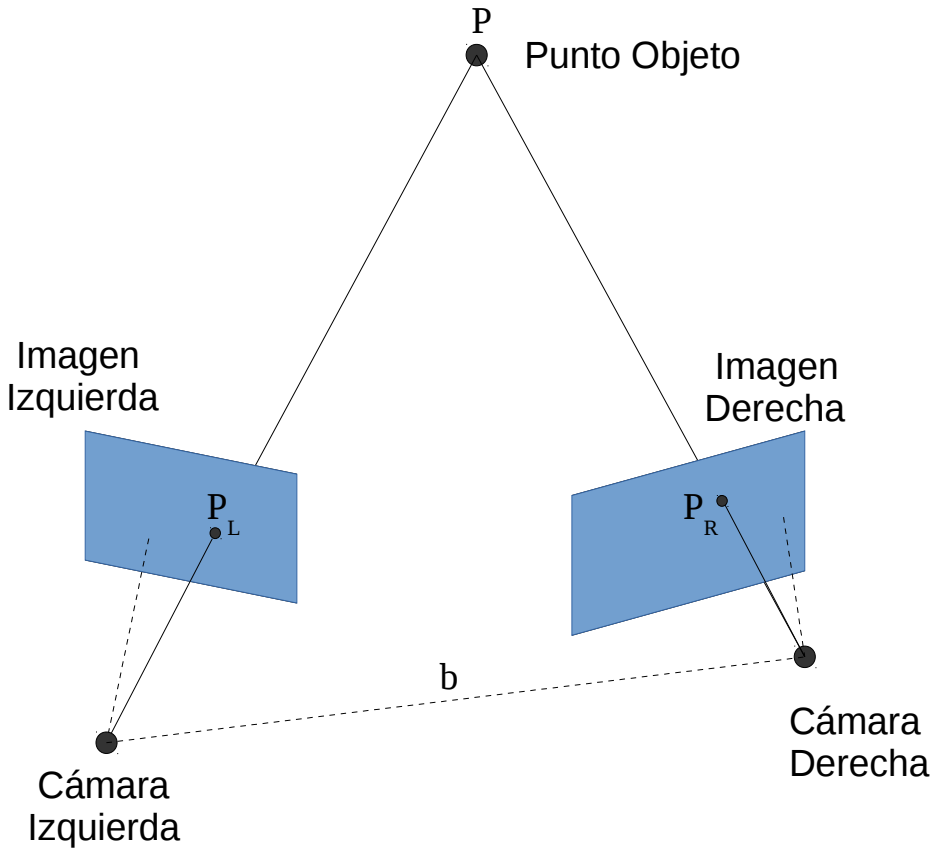



*Ilustración 21: Geometría Epipolar*

Como el sistema referencial se ubica en la posición de la cámara izquierda, se considera también entonces, una rotación de la cámara derecha.

Así:

$$W_L . (C^{-1})^T . B . R . C^{-1} . W_R = 0 \quad (25a)$$

con:

$$F = (C^{-1})^T . B . R . C^{-1} \quad (25b)$$

donde *F* es la matriz fundamental (Luong & Faugeras, 1993), que involucra 9 parámetros: 4 parámetros de calibración de cámaras, 3 de rotación y 2 para la base *b*. *F* es una matriz cuadrada de orden 3 y rango 2.

Cuando la calibración se conoce, solo los parámetros de $B$ y $R$ son incógnitas. En este caso, la matriz esencial (Horn & Berthold, 1990):

$$E = B . R \quad (26a)$$

Se obtiene sencillamente a partir de la matriz fundamental $F$ (Luong & Faugeras, 1997), ya que:

$$E = C^T . F . C \quad (26b)$$

Hay varios enfoques para obtener los 5 parámetros involucrados: Longuet-Higgins (1981), Heikkila (2000), Zhang (2000), Zhang (2004) y Stewenius et al. (2006). El más usado involucra la descomposición de la matriz esencial: Hartley & Zimmerman (2003).

Hay muchos métodos que han sido propuestos para la calibración de cámaras. En Bergamini et al. (2016b) se propuso un esquema de calibración sencillo que es fácil de implementar.

Se asume que es simple obtener las coordenadas $x$ e $y$ del punto de vista de la cámara en relación con el referencial 3D cuando el eje óptico de la cámara tiene una rotación pequeña respecto del eje $Z$. Se acepta que el error del punto de vista en la coordenada $z$ es menor que 1cm.

Mediante el uso de la matriz esencial (Kalantary & Jung, 2008), se obtiene la base $B$ y la rotación $R$ como solución del sistema de ecuaciones lineales (Bergamini et al., 2016b).

Los parámetros de calibración son necesarios para obtener con más precisión las coordenadas tridimensionales de los puntos de la escena que se obtienen mediante estereoscopía.

## 3.5.2. Obtención de puntos homólogos

### Presentación general

La identificación automática de puntos homólogos obtenida a partir de dos o más imágenes digitales es un problema esencial para lograr un modelo tridimensional de la escena mediante estereoscopía. Estos puntos corresponden a las proyecciones de dos o más imágenes de un mismo punto físico de la escena.

Al problema de la obtención automática de puntos homólogos se lo llama *apareo estéreo* (o simplemente *apareo*) y se lo llama también *puesta en correspondencia* (Donadio y Mendez, 1997). Es sin dudas, la tarea más compleja en el proceso de visión 3D.

### Algunas condiciones para lograr el apareo

**Epipolaridad:** Los puntos homólogos deben estar sobre la misma linea epipolar, es decir, los puntos principales de cada imagen deben estar en la misma línea. A esto se lo llama que las imágenes estén roto-rectificadas.

El cumplimiento de esta condición restringe notoriamente el espacio de búsqueda de puntos homólogos. Sin embargo, esta condición es de difícil cumplimiento, ya que requiere orientación y alineamiento perfecto entre las cámaras.

**Similaridad:** Este es el principio básico para resolver el apareo. Establece que los elementos homólogos deben poseer características similares en ambas imágenes. Es decir, ambas imágenes deben ser “parecidas”. Para ello, los ángulos entre los planos de las cámaras entre sí y la rotación alrededor de sus normales deben ser pequeños.

**Continuidad figural:** Los objetos de una figura se espera que aparezcan, sin oclusiones en ambas imágenes.

### Tipos de algoritmos de apareo

**Algoritmos basados en áreas:** Emplean como primitiva la intensidad de los píxeles y su estrategia de apareo se aplica en forma local sobre los píxeles de la imagen. Se supone que las propiedades fotométricas se mantienen invariantes en ambas imágenes. Soluciones basadas en correlación, por ejemplo, se utilizan áreas.

**Algoritmos basados en características:** Seleccionan un conjunto de primitivas abstractas y aplican el apareo sobre ese conjunto asumiendo que las propiedades geométricas de la escena se mantienen invariantes. Estrategias de análisis de bordes o contornos usan este tipo de algoritmos.

## 3.5.3. Estereoscopía

La fotogrametría o metrología con imágenes estereoscópicas o simplemente estereoscopía es un conjunto de técnicas que mediante una cámara fotográfica, permiten deducir una proyección cónica de la imagen. Es posible proveer con precisión conocida: dimensiones, orientación y ubicación de objetos en el espacio 3D utilizando medidas hechas en dos o más imágenes. Partiendo de dos imágenes de una misma escena, conociendo sus puntos homólogos, mediante el uso de la estereoscopía es posible realizar su reconstrucción tridimensional. Cuantos más puntos homólogos de la escena se conozcan, mejor resultará su modelo tridimensional. Por lo tanto, el cálculo de las coordenadas 3D de cada uno de estos puntos (denominados genéricamente aquí como $M_Q$) se obtiene fácilmente mediante contribuciones de información de ambas imágenes (Zelasco et al., 2000).

Las coordenadas espaciales del punto genérico $M_Q = (x\ ;\ y\ ;\ z)^T$ se obtienen utilizando la ecuación de reconstrucción de escena aplicada a cada imagen:

$$\begin{pmatrix} \breve{i}.R - U.\breve{k}.R \\ \breve{j}.R - V.\breve{k}.R \end{pmatrix}.M_q = \begin{pmatrix} \breve{i}.L - U.\breve{k}.L \\ \breve{j}.L - V.\breve{k}.L \end{pmatrix} \quad (27)$$

Donde $R$ es la matriz de rotación de la cámara, $L = R\ .\ S$ con $S$ la posición de la cámara en el sistema de referencia general, y $U$ y $V$ son coordenadas $x$ e $y$ en unidades de longitud de la proyección del punto $M_Q$ en cada imagen. Es importante tener en cuenta que en estas ecuaciones se usan algunos parámetros calculados en la instancia de calibración. Dado que $M_Q$ es desconocido, $M_Q$ puede llamarse como $X$, y la expresión (27) puede presentarse resumidamente como: $A\ .\ X = B$ donde $A$ es una matriz de 2 x 3, y $B$ es una matriz de 2 x 1.

Entonces, la contribución de la imagen de la izquierda es: $A_L\ .\ X = B_L$ y la contribución de la imagen de la derecha es: $A_R\ .\ X = B_R$. Uniendo las contribuciones de ambas imágenes:

$$\begin{pmatrix} A_L \\ A_R \end{pmatrix}.X = \begin{pmatrix} B_L \\ B_R \end{pmatrix} \quad (28)$$

Llamando $A = (A_L\ ;\ A_R)^T$ y $B = (B_L\ ;\ B_R)^T$ la ecuación 28 puede presentarse:

$$A\ .\ X = B \quad (29)$$

Donde $A$ es una matriz de *4 x 3*, $B$ es una matriz de *4 x 1*. Pre multiplicando por $A^{\mathrm{T}}$ en ambos miembros de (29) y luego también por $(A^{\mathrm{T}}.A)^{-1}$ las coordenadas 3D de la reconstrucción tridimensional pueden calcularse:

$$M_q = \begin{pmatrix} x \\ y \\ z \end{pmatrix} = \left(A^T . A\right)^{-1} . A^T . B \tag{30}$$

Una vez que se obtienen los puntos en las coordenadas espaciales, se puede lograr el modelo tridimensional de la escena.

# 4. Desarrollo técnico de la propuesta

## 4.1. Enfoque y abordaje del desarrollo técnico

El desarrollo técnico de esta propuesta presenta un proceso que permite obtener rápidamente los puntos homólogos de un objeto presente en dos imágenes de una misma escena. Las condiciones de epipolaridad, similaridad y continuidad figural (presentadas previamente en 3.5.2.2) son necesarias para lograr el apareo entre ambas imágenes.

El tipo de algoritmo desarrollado en esta propuesta es incremental, basado en características, las cuales se van hallando mientras se recorren las curvas de borde.

La ilustración 22 presenta un esquema que describe globalmente al proceso.

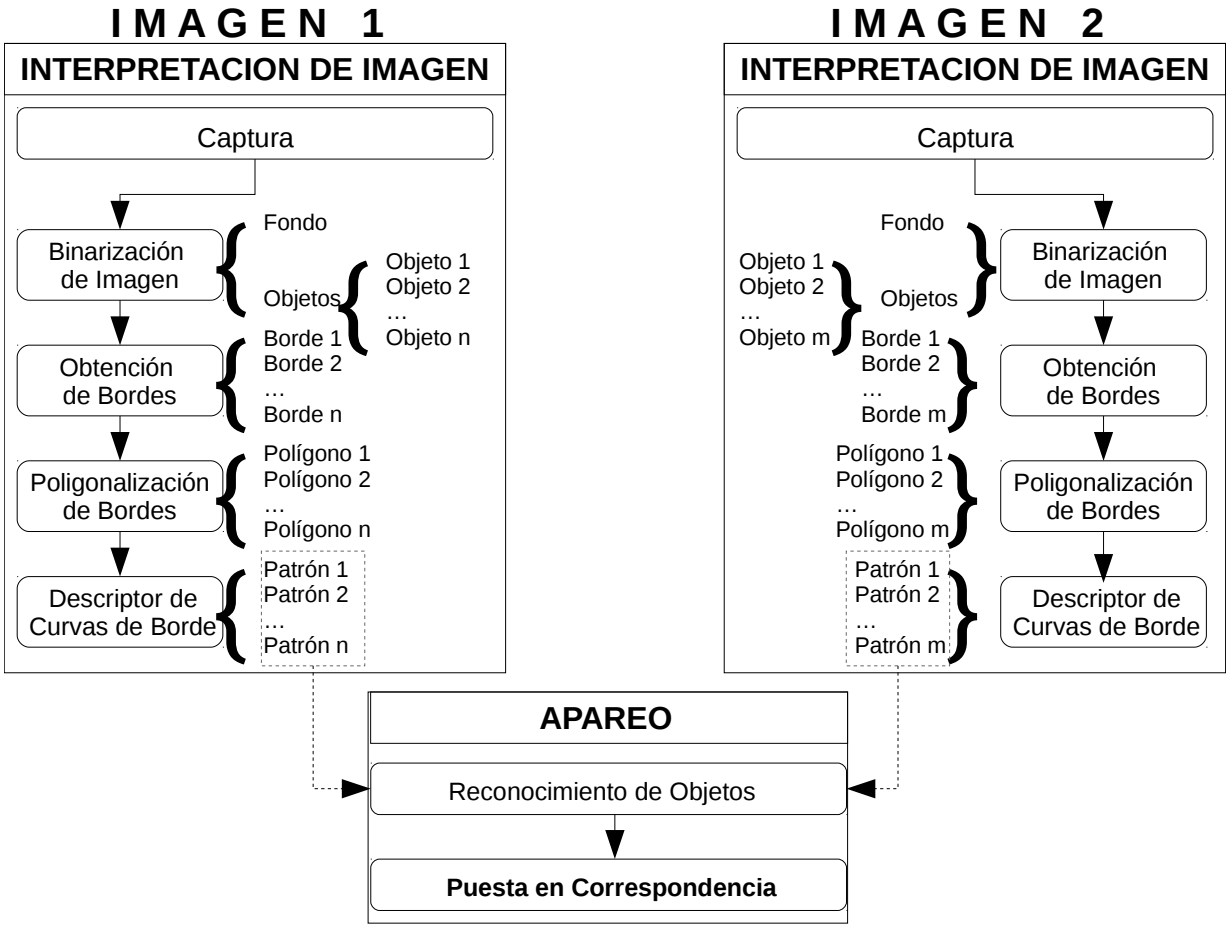



*Ilustración 22: Esquema del proceso de obtención de puntos homólogos*

Puede observarse que los sub-procesos de interpretación de imágenes pueden ser hechos en hilos separados, en paralelo, de modo de lograr así resultados más eficientes. Este enfoque redunda en

una mejora importante en los tiempos de procesamiento respecto de lo presentado en Kamlofsky et al. (2018). Se analiza en el Capítulo 5.

Generalmente, *búsqueda de puntos homólogos*, *apareo* y *puesta en correspondencia* se denomina al mismo proceso que entrega los pares de correspondencia de puntos de un objeto entre las imágenes. Como en este trabajo se analiza esto en detalle, al proceso global aquí se lo llama *búsqueda de puntos homólogos*, al sub-proceso que parte de las imágenes analizadas y finaliza del mismo modo que el proceso global se lo denomina *apareo*, mientras que la tarea que solo relaciona la correspondencia entre puntos de las imágenes de objetos ya reconocidos, se la denomina *puesta en correspondencia*.

En este capítulo, se presenta en detalle las tareas de cada sub-proceso según lo esquematizado en la ilustración 22. Es decir, se analizará en detalle:

- El sub-proceso de interpretación de imagen
- El sub-proceso de Apareo

## 4.2. El Sub-proceso de interpretación de imagen

### 4.2.1. Captura, pre-procesamiento y binarización de la imagen

El sub-proceso denominado *interpretación de imagen* se inicia con la captura de la imagen y finaliza con la presentación de una colección de patrones descriptores de los objetos hallados en la misma. El tipo de cámara adecuado depende del problema. El desarrollo de esta propuesta es independiente del dispositivo de adquisición. Por lo tanto, no se presentan más detalles. A los fines de la presentación de esta propuesta, se supone que las imágenes se reciben sin necesidad de ser mejoradas o pre-procesadas, y en condiciones para realizar la binarización.

La binarización o segmentación se realiza mediante umbralizado simple, tal como fue definido en Rosenfeld (1979). Para lograrlo, se debe recorrer cada píxel de la imagen. Se reemplaza los niveles de gris por 0 si el valor del píxel es menor que un umbral definido, y por 255 en otro caso. Luego, los píxeles que tienen valor 255 se los agrupa conformando el fondo, mientras que los píxeles con valor 0 se agrupan conformando al conjunto de objetos de interés.

El recorrido en una imagen digital se puede realizar con dos ciclos *for* anidados. El primero para las filas (componente $y$): desde 0 hasta el valor del alto de la imagen −1, de arriba hacia abajo. El segundo ciclo es para las columnas (componente $x$): desde 0 hasta el valor del ancho de la imagen −1 (de izquierda a derecha). Así, la imagen se recorrerá: de izquierda a derecha, y de arriba hacia abajo.

### 4.2.2. Obtención de puntos de borde: el algoritmo BF4/8

#### Presentación

Obtener los bordes de un objeto permite extraer muchas de sus características con una cantidad reducida de puntos sin perder muchos atributos. Con el algoritmo BF4/8 se obtienen los puntos de borde de los objetos en una imagen binarizada.

Sea $C$ una componente conexa de $S$ (es decir, un objeto) y $D$ una componente conexa de $S^C$ (es decir, una componente del fondo), BF4/8 permite obtener el borde de $C$ respecto de $D$. El algoritmo fue presentado originalmente por Rosenfeld (1979).

Correctamente, BF4/8 son dos algoritmos: BF4 considera 4-conectividad para $C$ y 8-conectividad para $D$, mientras que BF8 considera 8-conectividad para $C$ y 4-conectividad para $D$. BF4 presenta bordes más gruesos, compuestos por más puntos que BF8.

## El Código Cadena

El algoritmo BF4/8 posee una importante ventaja: la codificación compacta de los puntos de la curva de borde.

Cuando se almacena un punto a la colección de puntos del borde, se agrega un par ordenado que contiene las coordenadas del píxel.

**Ejemplo 19:** En una imagen de de 1024 píxeles de ancho y 1024 píxeles de alto (1Mp), cada punto de la imagen se almacenará en un par $(x,y)$, donde tanto $x$ como $y$ se almacenan con 10 bits, por lo que cada punto de borde se almacenará en 20 bits.

Al obtener los puntos de borde de una curva mediante el algoritmo BF4/8, a partir del punto de inicio, cada uno de los sucesivos puntos obtenidos, son 4-vecinos u 8-vecinos del anterior. Esto es: cada $P_i$ de la curva tendrá un vecino $P_{i+1}$, que se ubicará a un ángulo de $k.(p/4)$.

Entonces, en BF4: $0 \leq x \leq 3$ (2 bits) y en BF8: $0 \leq x \leq 7$ (3 bits). Entonces, en el caso de la imagen anteriormente ejemplificada, si la curva de borde está compuesta por 100 puntos, su almacenamiento mediante el código cadena ocupará: $100\ x\ 20bits = 2000bits$. Si en cambio, se utilizara el código cadena mediante BF8 ocupará: $1\ x\ 20bits + 99\ x\ 3bits = 317bits$. Su código cadena mediante BF4 ocupará: $1\ x\ 20bits + 99\ x\ 2bits = 218bits$.

## Algoritmo BF4

**Resumen:** Dado un par de puntos $(P_i\ ;\ Q_i)$ con $P_i \in C$ y $Q_i \in D$ con $P_i$ 4-vecino de $Q_i$, el algoritmo determina un nuevo par de puntos $(P_{i+1},Q_{i+1})$ con $P_{i+1} \in C$ y $Q_{i+1} \in D$, $0 < i < n$ con $P_{i+1}$ 4–vecino de $Q_{i+1}$ y $P_{i+1}$ 4–vecino de $P_i$. Así se visitarán todos los puntos de $C_D$.

Inicia con el par de puntos $(P_0\ ;\ Q_0)$. Sea $R_0$ la 8-vecindad de $P_0$ ordenada en sentido horario, partiendo desde $Q_0$. $R_0 = Q_0; R_{02}; R_{03}; \ldots; R_{08}$. Sea $R_{0j}$ el primer elemento de $R_0$ perteneciente a $C$, y 4–vecino de $P_0$. Si $R_{0j-1} \in D$, $(P_1\ ;\ Q_1) = (R_{0j}\ ;\ R_{0j-1})$. Sino: $(P_1\ ;\ Q_1) = (R_{0j-1}\ ;\ R_{0j-2})$.

El proceso se repite sobre los restantes puntos. Finaliza cuando $(P_i\ ;\ Q_i)=(P_0\ ;\ Q_0)$.

**Ejemplo 20:** En la ilustración 23 se muestra una imagen donde se implementa el algoritmo BF4. A continuación se presenta cada uno de los pasos.

- **Inicio (a):** Se inicia con los puntos $(P_0\ ;\ Q_0) = ((2\ ;\ 3)\ ,\ (1\ ;\ 3))$
- **Paso 1 (b):** De la primer vecindad se obtiene: $(P_1\ ;\ Q_1) = ((2\ ;\ 2)\ ,\ (1\ ;\ 2))$ ya que $(2\ ;\ 2) \in C$, $(2\ ;\ 2)$ es 4-vecino de $(2\ ;\ 3)$ y $(1\ ;\ 2) \in D$.

- **Paso 2 (c):** En el siguiente paso se obtiene: $(P_2 ; Q_2) = ((2 ; 3) , (3 ; 2))$ ya que $(2 ; 3) \in C$, $(2 ; 3)$ es 4-vecino de $(2 ; 2)$ y $(3 ; 2) \in D$. Puede observarse que: $P_2 = P_0$, y además, el punto $(1 ; 1)$ no es tenido en cuenta por el algoritmo BF4, lo cual es correcto.

- **Paso 3 (d):** En el tercer paso se obtiene: $(P_3 ; Q_3) = ((2 ; 4) , (3 ; 4))$ ya que $(2 ; 4) \in C$, $(2 ; 4)$ es 4–vecino de $(2 ; 3)$ y $(3 ; 4) \in D$.

- **Paso 4 (e):** En el cuarto paso se obtiene: $(P_4 ; Q_4) = ((1 ; 4) , (1 ; 5))$ ya que $(1 ; 4) \in C$, $(1 ; 4)$ es 4–vecino de $(2 ; 2)$ y $(1 ; 5) \in D$.

- **Paso 5 (f):** En el quinto paso se tiene que: $(P_5 ; Q_5) = ((2 ; 3) , (1 ; 3))$. Como $(P_5 ; Q_5) = (P_0 ; Q_0) = ((2 ; 3) , (1 ; 3))$, el algoritmo finaliza.

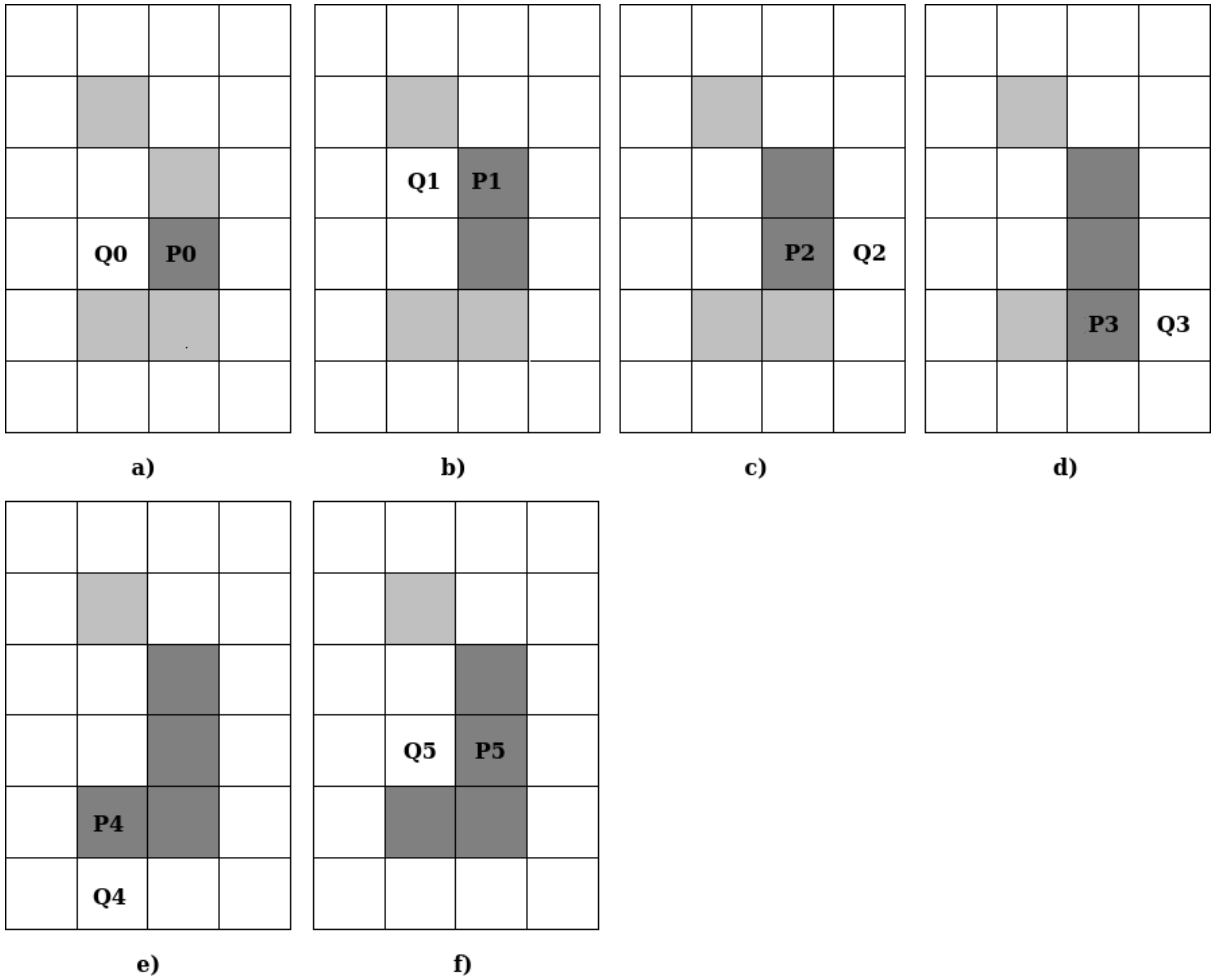



*Ilustración 23: Ejemplo de implementación del algoritmo BF4.*

## Algoritmo BF8

**Resumen:** Dado un par de puntos $(P_i ; Q_i)$ con $P_i \in C$ y $Q_i \in D$ con $P_i$ 4-vecino de $Q_i$, el algoritmo determina un nuevo par de puntos $(P_{i+1} ; Q_{i+1})$ con $P_{i+1} \in C$ y $Q_{i+1} \in D$ con $P_{i+1}$ 4–vecino de $Q_{i+1}$ y $P_{i+1}$ 8–vecino de $P_i$. Así se visitarán todos los puntos de $C_D$.

Inicia con el par de puntos ($P_0$ ; $Q_0$). Sea $R_0$ la 8-vecindad de $P_0$ ordenada en sentido horario, partiendo desde $Q_0$. $R_0 = Q_0$ ; $R_{02}$ ; $R_{03}$ ; … ; $R_{08}$ . Sea $R_{0j}$ el primer elemento de $R_0$ perteneciente a $C$, y 8–vecino de $P_0$. Como $R_{0j-1} \in D$, ($P_1$ ; $Q_1$) = ($R_{0j}$ ; $R_{0j-1}$).

El proceso se repite sobre los restantes puntos. Finaliza cuando ($P_i$ ; $Q_i$) = ($P_0$ ; $Q_0$).

Ejemplos y una explicación más amplia puede hallarse en el trabajo de Rosenfeld (1979).

Una particularidad de este algoritmo es que si las vecindades se recorren en sentido horario, así como los bordes del objeto, se estará obteniendo los bordes externos de un objeto. Sin embargo, si las vecindades se recorren en sentido horario, y los bordes se obtienen en sentido anti-horario, se estará obteniendo el borde interno o un agujero.

## 4.2.3. Simplificación de las Curvas de Borde

### Introducción

El análisis de bordes de un objeto permite estudiar eficientemente la forma de un objeto en una imagen digital: sólo se analiza una pequeña cantidad de puntos del mismo, de modo que ese conjunto de puntos obtenidos mantiene ciertas propiedades topológicas respecto del objeto original.

La curvas de bordes de los objetos pueden simplificarse mediante una transformación de dichas curvas a polígonos aproximantes: se logran aproximaciones adecuadas de las curvas que nos aseguran robustez algorítmica y baja complejidad computacional (De Berg, 1997) y además se permite ignorar efectos de ruido causados por la digitalización. El grado de aproximación se controla con un parámetro $\varepsilon$, que permite manejar el balance entre el error de la aproximación, y la complejidad del polígono resultante.

En esta sub-sección se propone un método para aproximar el borde de un objeto por una sucesión de segmentos rectos. La aproximación se hace teniendo en cuenta una tolerancia del error $\varepsilon$ cometido al remplazar puntos del borde por segmentos rectos. Como consecuencia de la aproximación, el borde se suaviza, eliminándose la incidencia del ruido en la forma de la imagen digital. Otra ventaja de este enfoque es que el borde se reduce ahora a un conjunto de puntos que son los vértices del polígono aproximante, lo que brinda una enorme ganancia en capacidad de procesamiento, cuestión elemental para lograr seguimiento de objetos en tiempo real. Este enfoque fue presentado en Kamlofsky y Bergamini (2012). Al final de esta sección se presentan otros enfoques estudiados y propuestos.

### Pre-poligonalización

La pre-poligonalización es una simplificación de la curva de borde, eliminando puntos que están alineados. Equivale a una aproximación con tolerancia $\varepsilon = 0$. Mediante esta transformación se pretende obtener disminución del costo computacional en la etapa principal del algoritmo, sin perder información relevante acerca de las características del borde.

**Funcionamiento del algoritmo:** Este procedimiento recorre el código cadena de la curva, y si encuentra repeticiones consecutivas de código $k$, las mismas corresponden a segmentos de recta de pendiente $k.\pi/2$ (en 4–curvas) o $k.\pi/4$ (en 8–curvas). Entonces, calcula el puntos extremos de tal segmento, lo almacena en la lista de vértices del pre-polígono, y elimina las repeticiones.

**Ejemplo 21:** En la ilustración 24 se presenta la Pre-poligonalización de una forma en una imagen sintética correspondiente al dibujo de un automóvil.

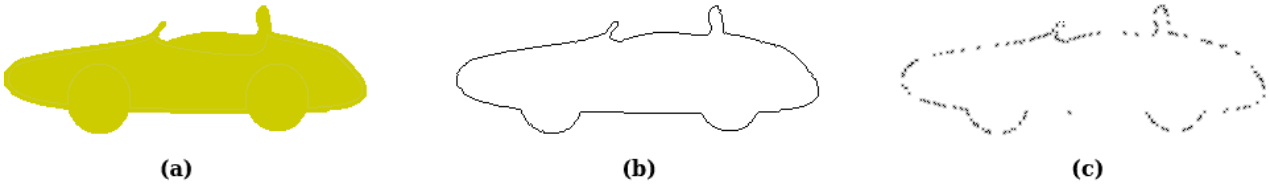



*Ilustración 24: Pre-poligonalización de una forma. (a) Forma tratada. (b) Sus bordes obtenidos mediante BF8. (c) Puntos restantes tras el procedimiento.*

En la ilustración 24 (a) se presenta el objeto a analizar, el dibujo de un automóvil con 14971 píxeles. En (b) se muestra el borde obtenido mediante BF8. La cantidad de puntos del borde es de 648 píxeles. En (c) se muestra el conjunto de vértices luego de la pre-poligonalización. La cantidad total de puntos resultantes es de 212 píxeles, que representa el 32% de los puntos del borde del objeto. Es decir, luego de este procedimiento muy sencillo se ingresa al procedimiento principal de simplificación, una cantidad de puntos que representa una reducción del 68% del total de bordes.

## Poligonalización

Se recibe un conjunto de puntos ordenados que corresponden a un borde de una forma que fue pre-poligonalizado. Dado un coeficiente de tolerancia, se realizará la poligonalización del borde en función de ese coeficiente. Resultará en un nuevo conjunto de puntos mucho más reducido que el anterior, que corresponderá con el conjunto de vértices del polígono aproximante.

**Funcionamiento del algoritmo:** Se inicia con el primer punto $P_0$. Se van agregando puntos al lado mientras el ancho de la cáscara convexa del lado sea menor que una tolerancia e: método admitirPuntoEnLado. Si para un punto $P_i$, el ancho de la cáscara convexa de todos los puntos del lado es mayor a e, el punto $P_0$ y el punto $P_{i-1}$, son los extremos del primer lado y se eliminan los puntos restantes. El método admitirPuntoEnLado se repite a partir de $P_{i-1}$, y finaliza cuando el último extremo sea $P_0$.

**La cáscara convexa:** Dado un conjunto de puntos en el plano $X$, la cáscara convexa o envoltura convexa de $X$ se la define como la intersección de todos los conjuntos convexos que contienen a $X$. Una buena analogía para entender este concepto se presenta en De Berg (1997): imagine que tiene un conjunto de clavos clavados en una tabla plana, y luego encierra los clavos con una banda elástica. Ese polígono que encierra a todos los clavos mediante la banda elástica es la cáscara convexa.

**Ejemplo 22:** En la ilustración 25 se muestra un ejemplo del proceso denominado admitirPuntoEnLado utilizando el concepto de la cáscara convexa.

En la ilustración 25 se muestra el resultado de aplicar el algoritmo de poligonalización a un arco: un tramo de una curva. En (a), se observa el conjunto de puntos obtenido al aplicar el procedimiento de pre-poligonalización, resultando en un conjunto de nueve puntos. En (b) se muestra la envoltura convexa de los primeros 5 puntos. El ancho de la misma es $w_1$, que es menor que la tolerancia $\varepsilon$. La envoltura convexa que incluye los puntos de $P_0$ a $P_5$ no verifica que su ancho sea menor que la tolerancia. El conjunto de puntos entre $P_4$ y $P_8$ tiene ancho $w_2$, también menor que la

tolerancia. En (c) se muestran los segmentos resultantes de aproximar $P_0$ – $P_4$ y $P_4$ – $P_8$, identificando tres vértices de la poligonal que aproxima la curva original, como se muestra en (d).

**Ejemplo 23:** En la ilustración 26 se presenta un ejemplo de poligonalización de una curva cerrada. En este ejemplo, se muestra el resultado de aplicar el algoritmo a la imagen original mostrada en la ilustración 24, con tolerancias $\varepsilon_1 = 2$ y $\varepsilon_2 = 5$ (en píxeles).

En el ejemplo 21 (graficado en la ilustración 24) puede destacarse que la aproximación de curvas con poligonales usando el algoritmo propuesto con tolerancia $\varepsilon = 2$ píxeles se obtuvo una frontera simplificada consistente de 48 píxeles (a), y la forma del objeto mantiene las características relevantes de la figura original (b). Elevando la tolerancia a $\varepsilon = 5$ píxeles, el borde se reduce a 22 píxeles (c). Obviamente, la calidad de la aproximación disminuye (d).

Es importante notar que mediante el procedimiento de poligonalización de las curvas de borde de una forma, se logra una simplificación notable de la misma. En el ejemplo 22, con 22 píxeles se logra una representación simplificada de un objeto de 14971 píxeles. Lo que implica que con aproximadamente 0,15% de los píxeles se logra una representación aproximada de un objeto a un costo computacional muy bajo.

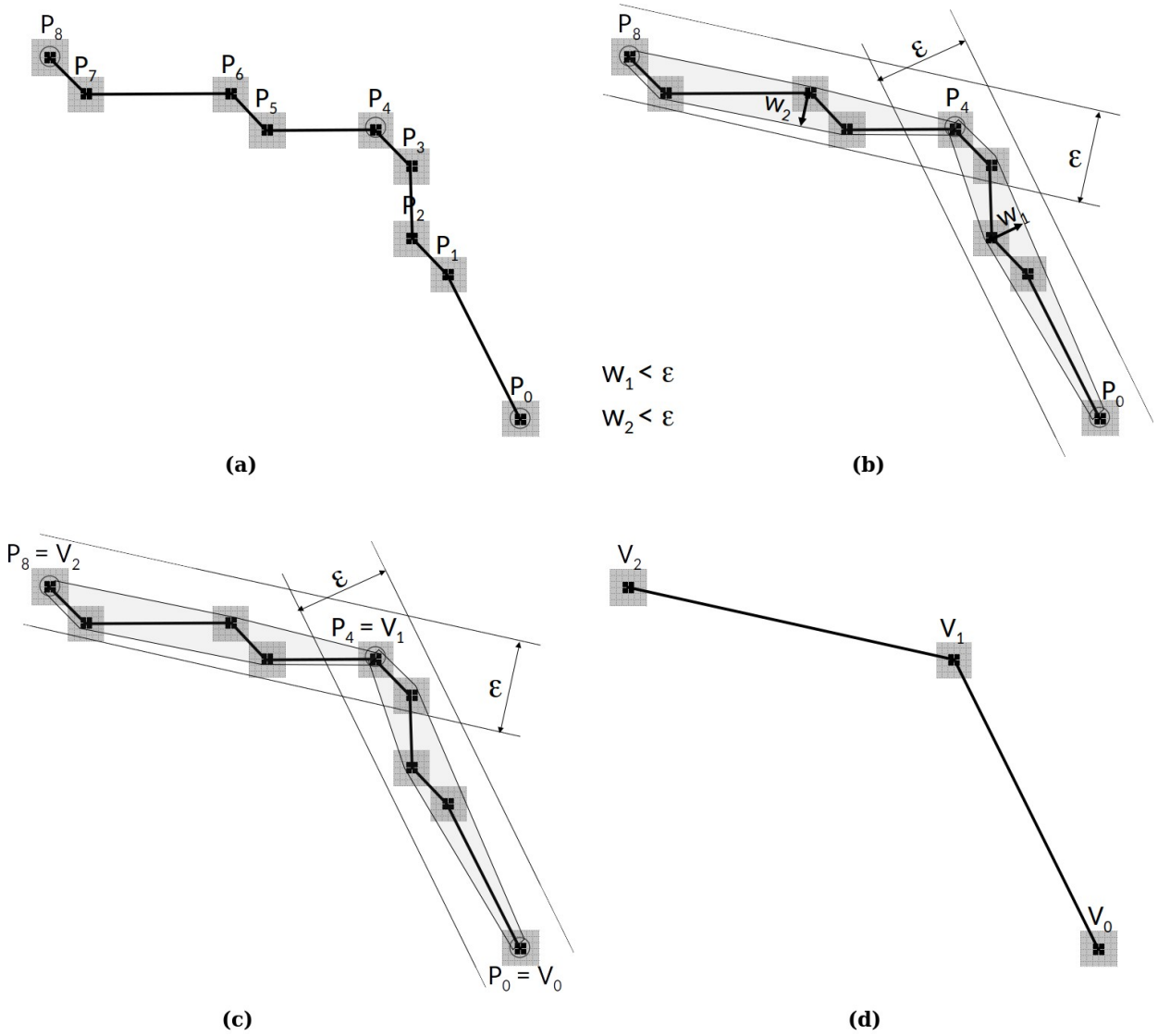



*Ilustración 25: El método admitirPuntoEnLado mediante el uso de la cáscara convexa.*

**Conclusiones acerca del método de simplificación:** El método propuesto para la simplificación del borde de un objeto digital permite representar al objeto con una cantidad reducida de puntos, manteniendo las características principales de su forma.

Además, mediante la poligonalización de curvas de borde se logra eliminar efectos adversos de la digitalización: el ruido.

El uso de envolturas convexas permite calcular el ancho de un conjunto de puntos de manera eficiente.

El control de la tolerancia queda en manos del usuario del algoritmo, quien podrá definirla según los requerimientos de la aplicación.

Así, con baja tolerancia pueden obtenerse resultados de alta calidad, mientras que con tolerancias mayores el usuario gana en velocidad de procesamiento. Esta herramienta es fundamental en el

proceso de reconocimiento de objetos en movimientos, donde los requerimientos de eficiencia y flexibilidad son de importancia relevante.

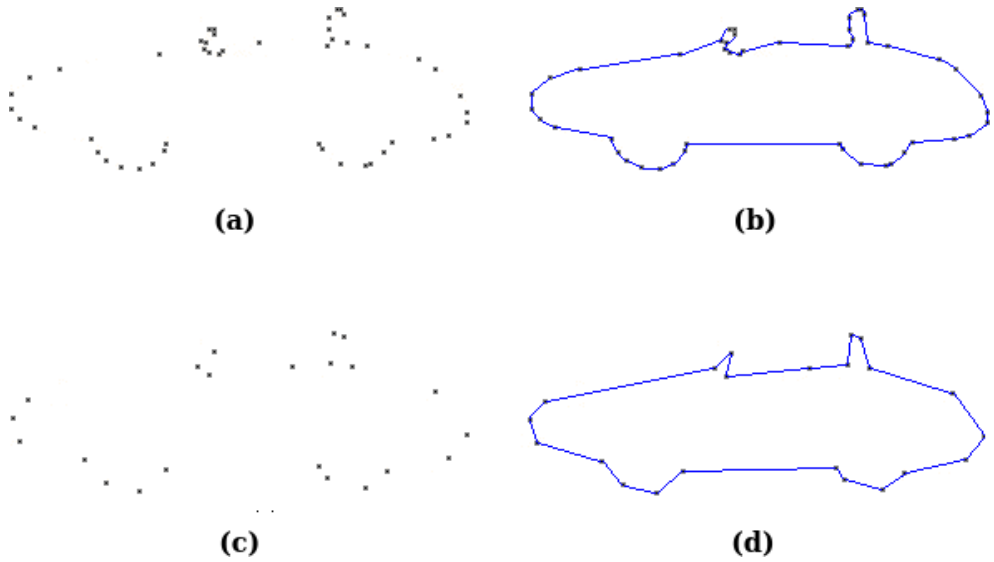



*Ilustración 26: (a, b) Vértices y polígono aproximante del objeto con tolerancia de 2 píxeles. (c, d) Vértices y polígono aproximante del mismo objeto con tolerancia de 5 píxeles.*

## Otros enfoques estudiados

**Poligonalización mediante barras digitales:** Este enfoque fue presentado en Kamlofsky y Bergamini (2013b). Dado un arco de la curva de borde de un objeto, se asume que se puede aproximar por un segmento de recta si todos sus puntos están dentro de una barra digital de ancho perpendicular $2\varepsilon$.

Dada la lista de puntos de borde $\{p_1, p_2, \ldots, p_m\}$, el algoritmo propuesto en Kamlofsky y Bergamini (2013b) divide la lista en sublistas maximales de puntos consecutivos $\{p_s, p_{s+1}, \ldots, p_{s+j}\}$, que se sustituyen por un lado del polígono. Los puntos en cada sub-lista se consideran aproximadamente alineados, al verse comprendidos en una barra digital de ancho perpendicular $2\varepsilon$, con eje en el segmento de recta continuo que une $p_s$ con $p_{s+j}$. Se dice que la sublista es maximal, porque al incorporar el siguiente punto de la curva, $p_{s+j+1}$, no se verifican esas condiciones. Así, $p_s$ y $p_{s+j}$ son vértices del polígono aproximante.

**Aproximación de arcos casi rectos:** En Bergamini y Kamlofsky (2017) se propuso un algoritmo de reconocimiento de segmentos de rectas digitales, que tiene en cuenta el ruido proveniente de los métodos de captura de imágenes, o del proceso de segmentación por umbralización. En el trabajo se deduce una condición de *arcos casi rectos*, a partir de la definición de recta digital basada en ecuaciones diofánticas lineales, y saca provecho de la representación de curvas mediante el código cadena. Se propone también la obtención de una aproximación poligonal de curvas digitales y de la cobertura tangencial, que luego facilita una sencilla forma de calcular la curvatura en cada punto.

## 4.2.4. Patrón descriptor de las curvas de borde

### Introducción

A partir de disponer de la representación simplificada de la curva de borde mediante su poligonalización, se espera extraer sus características distintivas, para poder identificar al objeto.

Se presenta un patrón descriptor de curvas cerradas. El mismo consiste en una función lineal a trozos de la evolución de la curvatura en función de la evolución del perímetro del polígono. Cada trozo es lineal debido a que la curvatura en los lados de un polígono es cero. El ángulo en cada uno de sus vértices se considera un estimador de la curvatura. Este estimador permite identificar características geométricas del objeto, como ser concavidad, convexidad, puntos angulosos, etc, mediante el análisis de la evolución de dicho parámetro a lo largo de la curva. Así, se obtiene una representación simplificada de la forma que implica una cantidad reducida de puntos en el espacio longitud-curvatura, y es poco sensible al ruido.

### El Patrón de evolución del ángulo de giro

Dado el polígono obtenido como aproximación de una curva digital, se proponen una representación del mismo basado en el patrón de evolución de ángulo de giro, que se explica a continuación.

**Nociones de geometría diferencial:**

Sea $C$ una curva regular parametrizada en el plano $xy$ con: $\alpha(s) = (x(s) ; y(s))$ donde $s$ es el parámetro longitud de arco, $s \in [0 , L]$ con $L$: Longitud de la curva. La curvatura en el punto $\alpha(s)$ es $\kappa(s)$, el cambio instantáneo de la inclinación del vector tangente a $C$ en $\alpha(s)$.

Sea $l(s)$ la longitud de la curva $C$ desde $\alpha(0)$ hasta $\alpha(s)$. Sea: $\lambda(s) = l(s) / L$ la longitud normalizada de la curva $C$. Así, $\lambda(s) = 0$, $\lambda(s) = 1$.

La curvatura acumulada en $s$ es:

$$\kappa_{acum}(s) = \int_0^s \kappa(r)\, dr \tag{31}$$

Entonces, $\kappa_{acum}(0) = 0$ y $\kappa_{acum}(L) = 2\pi$ para curvas simples cerradas.

Sea $\beta(s)$ la curva en el plano longitud-curvatura $l\kappa$ que representa la curvatura acumulada en función de la longitud normalizada:

$$\beta(s) = (\lambda(s), \kappa_{acum}(s)) \tag{32}$$

La ecuación de esta curva puede escribirse más sencillamente como:

$$\beta(s) = \kappa_{acum}(\lambda(s)) \tag{33}$$

**Extensión a polígonos:** Se extiende estas definiciones para el caso en el que $C$ sea un polígono con vértices $v_1, v_2, \ldots, v_n$.

En un polígono solo tiene sentido definir la curvatura en sus vértices (ya que en los lados es cero). La curvatura en el vértice $v_i$ es el ángulo interno al polígono en dicho vértice (o ángulo de giro) formado por los lados definidos por los vértices adyacentes: $v_{i-1}$ y $v_{i+1}$ . La curvatura acumulada en el vértice $v_i$ es:

$$\kappa_{acum}(i)=\sum_{j=1}^{j=i}\kappa(j) \tag{34}$$

es decir, es el ángulo de giro acumulado desde el primer vértice hasta el vértice $v_i$. Claramente:

$$\kappa_{acum}(n)=2\pi \tag{35}$$

Sea $\lambda(i)$ la suma de las longitudes de los segmentos del polígono desde $v_1v_2$ hasta $v_iv_{i+1}$ (tomando los índices módulo n) y $L$ la longitud total del polígono. Entonces: $\lambda(i) = l(i) / L$.

La evolución de giro de un polígono con *n* vértices es la curva lineal por tramos en el plano $\lambda\kappa$:

$$\beta(i)=(\lambda(i),\kappa_{acum}(i)) \tag{36}$$

Que puede expresarse más sencillamente como:

$$\beta(i)=\kappa_{acum}(\lambda(i)) \tag{37}$$

A esta función se la llama el *patrón de evolución del ángulo de giro* (o más simplemente *patrón de giro*). Esta función caracteriza a la forma del polígono, independientemente de la posición o escala del mismo en el plano. Es decir, es invariante frente a traslaciones, rotaciones y escala. Es un patrón normalizado. Además, brinda información sobre las características geométricas de la curva. Los intervalos de $\lambda$ donde $\lambda(s)$ es creciente, determinan las partes convexas del polígono. Y viceversa: los intervalos donde $\lambda(s)$ es decreciente, determinan las partes cóncavas.

Este patrón fue presentado en Kamlofsky y Bergamini (2013b).

**Ejemplo 24:** Dado un dibujo de una letra *C*, se presenta una aproximación poligonal de la curva de borde del dibujo, junto con su patrón de giro conteniendo partes cóncavas y convexas. Se muestra en la ilustración 27.

En el Ejemplo 24 graficado en la ilustración 27, se inicia el recorrido a partir del punto pintado en negro, en sentido antihorario. Se observa que los cinco puntos siguientes (azules) corresponden a partes cóncavas. En el patrón, esto se refleja con cinco puntos luego del inicial donde la función es puramente decreciente. Le sigue un tramo convexo (en rojo) donde la función descriptora es creciente salvo un tramo donde se observa una pequeña concavidad. Finaliza con una parte cóncava (en azul) netamente descendiente.

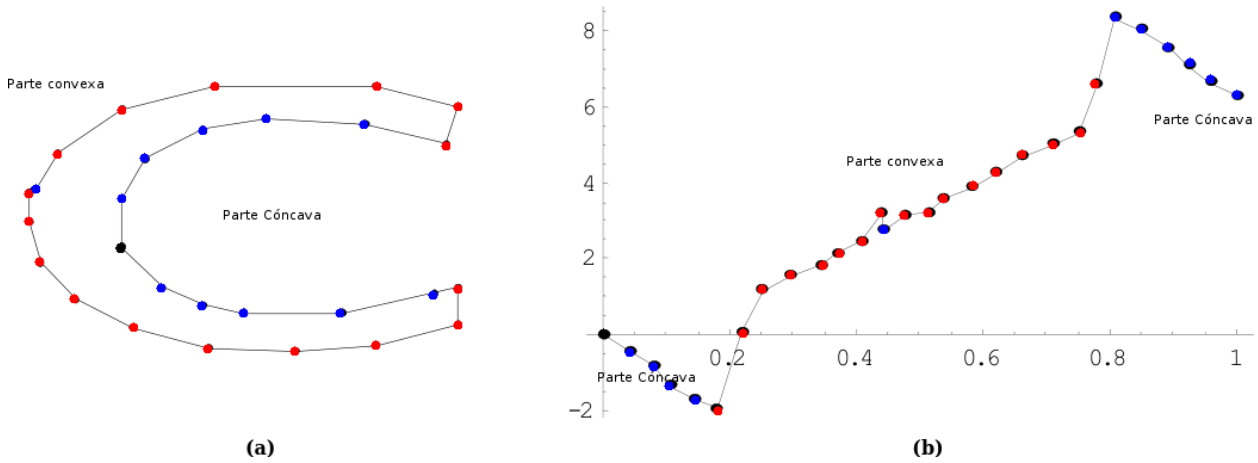



*Ilustración 27: (a) La poligonalización de una forma. (b) Su patrón de giro.*

El patrón de giro encierra toda la información de la silueta de la forma analizada. Las zonas crecientes del patrón representan las partes cóncavas de la curva, y las decrecientes las zonas convexas. De esta forma, los mínimos relativos del patrón representan puntos de inflexión. Estos puntos son puntos de referencia para segmentar la forma e identificar en el objeto analizado los tramos característicos.

Esto es de gran utilidad para llevar a cabo la clasificación de objetos de cierto tipo, que son en gran parte similares, y las disimilitudes se centran en una determinada zona.

## 4.3. El sub-proceso del apareo

### 4.3.1. Introducción

Se puede lograr la reconstrucción tridimensional de una escena mediante estereoscopía, a partir de dos imágenes obtenidas de la misma, desde dos puntos de origen diferentes, manteniendo la similaridad entre ellas.

Hallar los puntos homólogos en dos imágenes similares de una misma escena implica definir pares de correspondencia de puntos entre la primer imagen con la segunda.

En esta propuesta, para lograr ello, se analiza la primer imagen, se obtienen los objetos de interés junto con sus patrones descriptores, y los mismos se los busca en la segunda imagen. A esta etapa se la denomina *reconocimiento de objetos*.

Luego de reconocidos los objetos, se tratará de presentar una correspondencia entre los puntos de un objeto de la primer imagen con su reconocido en la otra imagen. A esta etapa se la denomina *puesta en correspondencia*.

### 4.3.2. Reconocimiento de objetos

En esta sub-sección se describe cómo puede usarse el patrón de evolución de ángulo de giro para hallar un objeto en una imagen.

## Distancia entre patrones de giro

Sean $k_1$ y $k_2$ dos patrones de de giro. La distancia entre los patrones $k_1$ y $k_2$ es el valor:

$$D(\kappa_1, \kappa_2) = \int_0^1 \left(\kappa_1(\lambda) - \kappa_2(\lambda)\right)^2 d\lambda \tag{38}$$

## Reconocimiento de objetos mediante la distancia entre sus patrones

Dos patrones obtenidos a partir de distintas instancias de la misma figura, donde una se obtiene por traslación y/o escalado de la otra en la imagen digital, tienen distancia cero entre sí. Pero si la transformación es una rotación esto no ocurre así.

La rotación es una transformación rígida en un espacio euclídeo, pero no en el plano digital. Al rotar una curva digital se exige a los puntos caer en puntos grilla, introduciendo así un error a causa de la discretización. Si se comparan instancias de una figura afectadas por una rotación, este error afectará el patrón de giro, haciendo que la distancia así definida no sea cero.

La clave para que el método de reconocimiento sea efectivo es que estas diferencias debido a la discretización sean mínimas frente a las diferencias debidas a cambios perceptibles en las formas (Kamlofsky y Bergamini, 2014a).

Dos representaciones $k_1$ y $k_2$ de la misma forma (o con gran similaridad), obtenidas de imágenes distintas, posiblemente a distinta escala y orientación, tienen distancia entre ellas pequeña. Análogamente, si dos representaciones son cercanas (en distancia), las curvas que las originan son similares (Kamlofsky y Bergamini, 2013a). Entonces, un objeto de una imagen es similar a otro objeto de otra imagen, independiente de su rotación, ubicación o escala si la distancia entre sus patrones descriptores es menor a un valor umbral predefinido por la experiencia.

Sea $k(\lambda)$ el patrón de giro de una forma $S$ que se quiere identificar. La identificación se realiza por comparación con patrones conocidos. Sean $\alpha_j(\lambda)$ patrones de formas candidatas a empatarse con la forma $S$. Entonces, se identifica a la forma $S$ con aquella cuyo patrón diste de $k$ en menos de un valor umbral $\mu$, es decir:

$$D(\kappa, \alpha_j) \leq \mu \tag{39}$$

En el marco de la visión 3D este enfoque resulta adecuado, debido a que las imágenes a partir de las cuales se desea hacer la reconstrucción tridimensional, necesariamente deben ser levemente distintas, por lo que las formas de sus objetos son entonces, similares y no iguales.

**Clasificación de objetos usando el patrón de giro:** En muchas ocasiones, objetos de una misma clase poseen patrones similares, debido a que sus formas son parecidas. A su vez, dentro de una misma clase, objetos de diferente sub-clase manifiestan diferencias específicas que pueden estudiarse con el patrón de giro.

La distancia definida en (39) también puede aplicarse a tramos parciales del patrón. La idea es analizar el tramo del borde comprendido entre dos puntos de referencia.

**Ejemplo 25:** Los patrones de borde de los vehículos vistos de perfil son similares entre sí. Autos tipo mono volumen, tricuerpo y furgones presentan muchas similitudes, pero las diferencias se

presentan principalmente en la cola (Kamlofsky y Bergamini, 2014a). La ilustración 28 presenta la comparación entre patrones de giro de este ejemplo.

En la ilustración 28(a) se muestran dos patrones diferentes. En (b) se presenta la comparación entre el patrón de un vehículo tricuerpo con un vehículo furgón. Se muestran dos patrones similares en casi toda la evolución donde se cumple la ecuación (39), pero que difieren marcadamente para $\lambda \in [0{,}15 \ ; \ 0{,}30]$. En (c) se muestran los patrones de dos vehículos tipo furgón. Se muestra que una distancia pequeña se distribuye uniformemente a lo largo de toda la evolución.

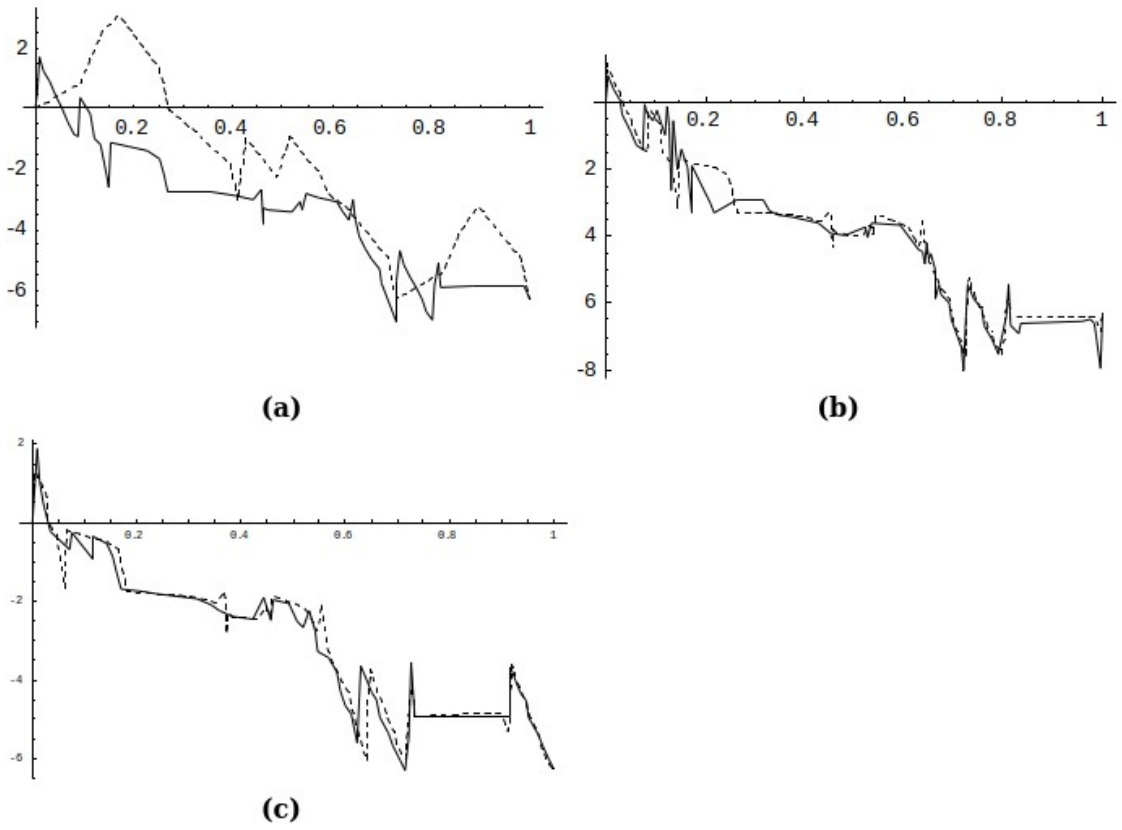



*Ilustración 28: Patrones de giro de distintos objetos. (a) Un vehículo (linea continua) y un objeto cualquiera. (b) Un vehículo tricuerpo (linea punteada) y un furgón (linea continua). (c) Dos furgones.*

## Dependencia del punto inicial

El algoritmo de obtención de bordes define al punto inicial como el que se encuentra más arriba y más a la izquierda en la imagen. Entonces, el punto inicial de la curva analizada, depende puramente de la orientación del objeto.

La curva  puede extenderse en forma continua y periódica al intervalo $\lambda = [1 \ ; \ 2]$ haciendo:

$$\kappa_{acum}(\lambda)=\kappa_{acum}(\lambda-1)+2\pi \quad (40)$$

Dadas dos representaciones mediante el patrón de giro del mismo objeto iniciadas desde distintos vértices, la distribución de la distancia entre ambos patrones será uniforme (es decir, sus representaciones son paralelas), pero no pequeña, impidiendo el cumplimiento de la ecuación (39), Y eso dependerá de la distancia entre los vértices de inicio: si los vértices de inicio coinciden, la distribución de la distancia será uniforme y pequeña (casi nula).

Si una de las poligonalizaciones posee *n* vértices, dado que los vértices de una poligonalización suele contener a los puntos relevantes del borde, y si además las formas son similares, se espera que tanto la cantidad de vértices como los ángulos en cada vértice de ambas sean similares. La distancia entre sus patrones será pequeña cuando coincidan los puntos de inicio. Entonces, si el punto de inicio es desconocido, son necesarias $n-1$ comparaciones hasta hallar una distancia pequeña entre patrones.

La determinación del punto de inicio puede lograrse fácilmente si ambos objetos poseen el mismo ángulo de rotación (condición de roto-rectificación). O bien, si los ángulos son relativamente distintos, detectada la orientación relativa, es simple obtener puntos de inicio correspondientes. Y además, lograr un punto homólogo del objeto en ambas imágenes.

La detección de la orientación o ángulo de inclinación se logra a partir de identificar al menor rectángulo que encierra a la curva. El ángulo que forman sus lados con el sistema de coordenadas cartesianas determina cuatro diferentes ángulos de inclinación u orientación:

$$\alpha+k.\frac{\pi}{2}, k=0,1,2,3. \quad (41)$$

Existen cuatro (o tres) puntos de cada curva que resultan de intersectarlas con el rectángulo minimal. Y esos puntos son mayormente correspondientes u homólogos en ambas curvas. El algoritmo iniciará el establecimiento del patrón descriptor en solo esos cuatro puntos (o a lo sumo tres puntos) posibles de la curva, independientemente de su orientación (Kamlofsky y Bergamini, 2013b). Esto significa que en lugar de necesitar realizar $n-1$ comparaciones de patrones, es suficiente con solo realizar cuatro (o tres) comparaciones.

Es importante aclarar que el rectángulo mínimo brindará información acerca de la orientación de la forma si la misma no se aproxima a una circunferencia.

En la ilustración 29 se muestra la detección de la orientación de una forma dentro de una imagen.

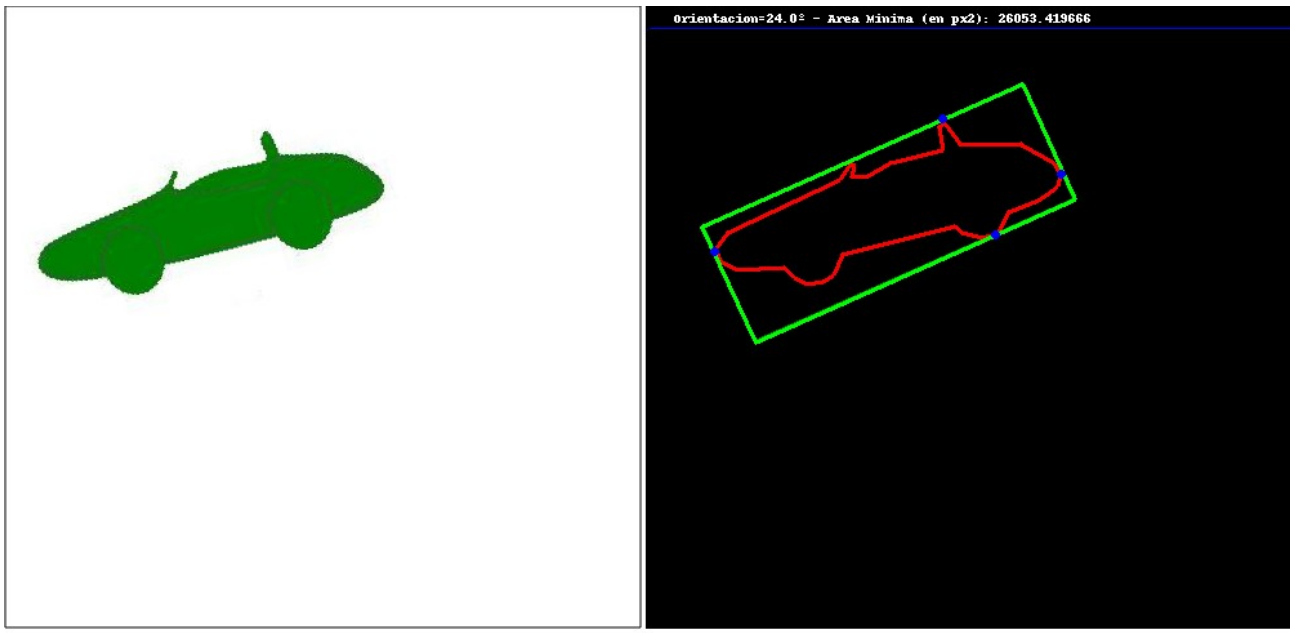



*Ilustración 29: (a) Objeto que se desea hallar. (b) Orientación del objeto candidato.*

En la ilustración 29(a) se muestra un objeto que se desea hallar en otra imagen. En 29(b) se muestra un objeto candidato encerrado por el rectángulo minimal. Puede observarse que el rectángulo interseca al borde del objeto en cuatro puntos: los cuatro puntos candidatos a punto de inicio. Además, el lado mayor indica la orientación de la forma tal como aquí se definió. En este caso, el objeto está rotado 24°.

El gráfico de la ilustración 29(b) se hizo utilizando la librería de gráficos matemáticos de Python Matplotlib[17].

## El reconocimiento

Luego que el rectángulo minimal haya permitido obtener la orientación de los objetos candidatos, se presentan (a lo sumo) cuatro puntos de inicio candidatos. Con ello, se establecen los patrones descriptores y se calcula las respectivas distancia.

**Ejemplo 26:** En la ilustración 30 se presentan dos imágenes (a) y (b). En la primera (a) se presenta el objeto que se desea buscar.

17 Sitio Oficial de Matplotlib: https://matplotlib.org/

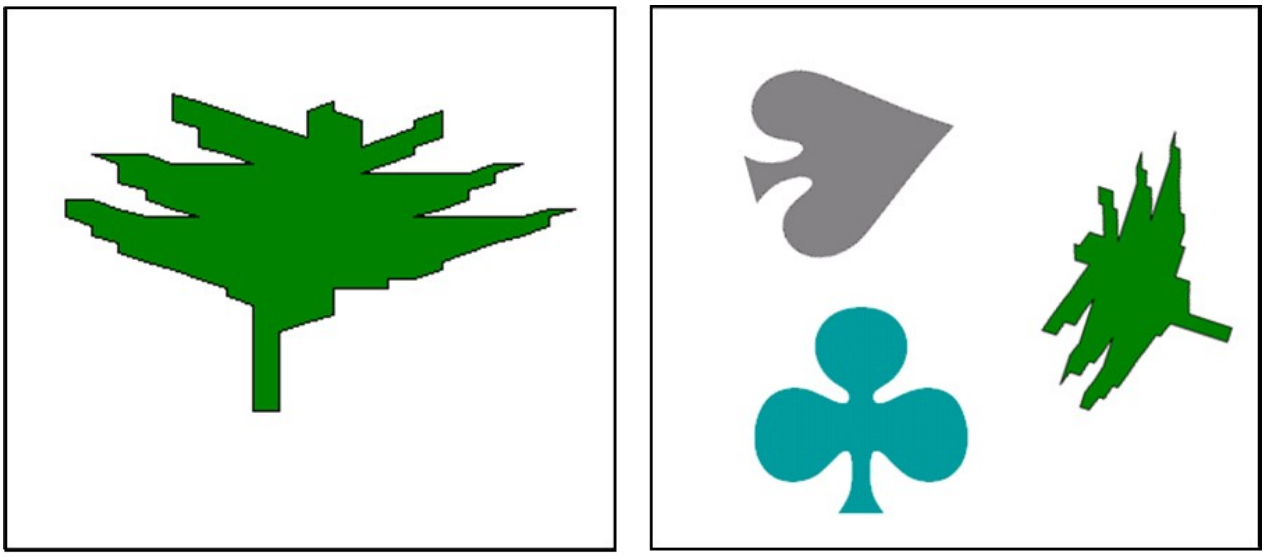



*Ilustración 30: (a) Imagen conteniendo el objeto que se desea buscar (Imagen Modelo). (b) Imagen que contiene varios objetos (Imagen de Búsqueda).*

Puede observarse que en la segunda imagen (b) se encuentra un dibujo similar al objeto de la primer imagen (a), con diferente orientación, ubicación y escala.

La Tabla 2 muestra los valores de distancia de los cuatro patrones de la forma que se desea buscar ilustración 30(a) con cada uno de los cuatro patrones de las tres formas (b). Se muestran las distancias entre los cuatro patrones del objeto modelo ubicado en la ilustración 30(a) y los patrones de los objetos ubicados en la imagen presentado en la ilustración 29(b). Se observa que el cuarto patrón del Objeto 2 (Hoja verde) presenta una distancia de 0,364 con el primer patrón del modelo, lo cual se encuentra destacado con un sombreado verde.

*Tabla 2: Distancias entre los diferentes patrones de objetos en una imagen con un patrón modelo.*

| ***Distancia entre patrones de Objetos*** | | ***Objeto en Imagen Modelo*** | | | |
|---|---|---|---|---|---|
| *Objeto de la Imagen de Búsqueda* | | *Patrón 1* | *Patrón 2* | *Patrón 3* | *Patrón 4* |
| *Objeto 1: Pica gris* | *Patrón 1* | *2,297* | *2,448* | *2,529* | *1,993* |
| | *Patrón 2* | *4,994* | *8,529* | *2,222* | *3,893* |
| | *Patrón 3* | *3,263* | *6,667* | *2,173* | *2,699* |
| | *Patrón 4* | *1,752* | *2,545* | *2,752* | *2,244* |
| *Objeto 2: Hoja verde* | *Patrón 1* | *2,654* | *0,695* | *5,485* | *3,485* |
| | *Patrón 2* | *2,081* | *5,143* | *1,959* | *2,29* |
| | *Patrón 3* | *2,641* | *3,957* | *2,63* | *0,798* |
| | *Patrón 4* | *0,364* | *2,684* | *2,3* | *2,318* |
| *Objeto 3: Trébol Turquesa* | *Patrón 1* | *4,149* | *7,297* | *1,698* | *3,718* |
| | *Patrón 2* | *3,402* | *6,155* | *1,482* | *3,361* |
| | *Patrón 3* | *1,724* | *3,107* | *3,034* | *1,547* |
| | *Patrón 4* | *1,912* | *3,587* | *2,855* | *1,693* |

Notar que en imágenes estéreo, donde hay similitud tanto entre los objetos como en las orientaciones de las cámaras, muchas veces no se requiere el cálculo y comparación de los cuatro patrones, ya que con el cálculo de uno solo de ellos es suficiente. Es decir, en estas condiciones, el primer patrón del objeto modelo obtendrá reconocimiento positivo con el primer patrón del objeto correspondiente.

### 4.3.3. Puesta en correspondencia

Se puede lograr la reconstrucción tridimensional de una escena mediante estereoscopía, a partir de dos imágenes similares obtenidas de la misma, desde dos puntos de origen diferentes, manteniendo la orientación de las cámaras. Para ello se requiere la puesta en correspondencia de los puntos de un objeto en ambas imágenes.

Lograr la puesta en correspondencia consiste en hallar pares de puntos donde el primer punto pertenece al borde poligonalizado de la forma modelo (o de la primer imagen), el segundo par corresponde a un punto de la segunda imagen (o imagen de búsqueda) que se corresponde con el primer punto. Es decir, dados ciertos puntos del objeto real, la puesta en correspondencia presenta los pares de puntos de ambas imágenes que muestran a los puntos reales del objeto.

Si las figuras en ambas imágenes están roto-rectificadas (condición de epipolaridad), es de esperar que los vértices que se ubiquen en la posición más elevada de la figura, sean los primeros vértices

que aparecen en las listas de vértices de ambas poligonalizaciones y que además sean correspondientes.

En cambio, si las imágenes no están roto-rectificadas, aún así, puede realizarse la puesta en correspondencia entre ambas figuras, si se mantienen las otras condiciones: similaridad y continuidad figural. Para ello, durante el reconocimiento de las formas, su roto-rectificación se logra luego de hallar la orientación. En este proceso, el rectángulo mínimo que encierra a la forma, se interseca con 4 (o en algún caso tres) puntos de ella, que son extremos que se corresponden en ambas imágenes.

Quedan determinados así, cuatro pares de puntos correspondientes (en algunas formas la cantidad de puntos puede ser tres). A partir de cualquiera de ellos, siguiendo el patrón en ambas figuras, es posible hallar el restos de los puntos homólogos, como se explica a continuación.

Gracias a que las curvas están parametrizadas por longitud de arco, y teniendo en cuenta que las imágenes son similares, pueden obtenerse más puntos homólogos: se recorren los vértices de una de las curvas, se busca su homólogo en la evolución del perímetro normalizado en la segunda curva, dentro de un radio pequeño de tolerancia $\rho$. Por ejemplo, si se usa una tolerancia $\rho$ = 5% para un vértice, y si un vértice $v_i$ tiene coordenadas en el patrón de la primer imagen ($\lambda_{1i}$ ; $\kappa_{1i}$), se buscará un vértice $v_j$ en el patrón de la segunda imagen en ($\lambda_{2j}$ ; $\kappa_{2j}$) de modo que: $D((\lambda_{1i} ; \kappa_{1i}) ; (\lambda_{2j} ; \kappa_{2j})) < 0{,}05 = \rho$, con $D(x ; y)$: distancia usual o euclídea, y se lo agregará al conjunto de puntos homólogos (si no ha sido ingresado previamente).

La necesidad de un intervalo de tolerancia se debe a que las imágenes obtenidas y las formas contenidas en las mismas son similares, pero no exactamente iguales. Esto tiene un significado análogo a lo que $\mu$ representa en la ecuación (39) para lograr el reconocimiento de una forma.

Finalmente, se obtiene la puesta en correspondencia entre un subconjunto de los polígonos aproximantes de las formas.

### 4.3.4. Mejoras en el Desempeño

En Kamlofsky y Bergamini (2017) y en Naidoo et al. (2017) se mostró que el proceso de obtención de puntos homólogos de un objeto en dos imágenes estéreo tomó 4,9 segundos.

Sin embargo, luego, en Kamlofsky et al. (2018) se mostró que agregando mejoras y contemplando ciertas restricciones, pueden obtenerse los mismos puntos homólogos, en 0,90 segundos, sin perder detalles importantes, lo cual se aproxima a los requerimientos de tiempo real.

La primer restricción se refiere a que la imagen se recibe ya binarizada (blanco y negro). Entonces, como en RGB el gris se representa con el promedio de las tres componentes de la imagen en color en cada una de las componentes, en el proceso de lectura, solo se lee una de las tres componentes RGB de la imagen binarizada.

La segunda restricción se refiere al tamaño mínimo de un objeto, definido en 10px de ancho y alto. Es decir, todo elemento de menos de 10px de ancho y 10px de alto no se tendrá en cuenta y será considerado ruido. De este modo, la recorrida de la imagen se realizará con dos *ciclos for* anidados a saltos de 10px tanto en ancho como en alto, reduciendo al 1% en promedio el proceso de lectura de los bits de la imagen.

La última restricción presentada en Kamlofsky et al. (2018) indica que en imágenes proveniente de videos, la velocidad del movimiento relativo del objeto es constante y conocida. Gracias a ello, en lugar de realizar la búsqueda del objeto en la totalidad de la imagen, se la realizará en solo una porción de la misma.

En este trabajo se propone una nueva mejora: tal como se indica en la ilustración 20, el procesamiento de lectura de imágenes, binarización, obtención de puntos de borde, poligonalización y obtención de los patrones de giro en cada imagen pueden realizarse en paralelo. Y dado que el equipamiento usado incluye cuatro procesadores, en este trabajo se realizaron las mencionadas tareas en distintos hilos, paralelamente, lo cual permitió mejorar aún más los tiempos del proceso.

## 4.4. La complejidad computacional del programa

### 4.4.1. Introducción

Presentar resultados experimentales que muestren tiempos de procesamiento de un programa en ciertas condiciones puede dar una noción acerca de la performance de la solución, pero es muy dificil realizar una comparación con otros algoritmos, ya que deberían presentarse exactamente bajo las mismas condiciones. El cálculo de la complejidad computacional de un algoritmo permite presentar un indicador que resulta adecuado para ello.

### 4.4.2. Nociones básicas

La complejidad algorítmica es una medida teórica que permite comprender el comportamiento de un algoritmo en función de recursos computacionales como ser tiempo (cuánto tiempo se tarda en resolver un problema usando el algoritmo) o bien en función de la memoria requerida (cuánta memoria consume el algoritmo para su solución). Así, puede compararse cómo un par de algoritmos se comportan frente a un problema, y en función de ello, decidir (Guillermo, 2018).

Se llama O(f(n)) al *Órden de la complejidad* de la función f(n). Se suele identificarse la complejidad de ciertas funciones de referencia. La tabla 3 muestra a las funciones de referencia con su correspondiente orden de complejidad.

*Tabla 3: Ordenes de complejidad de las funciones de referencia*

| Función | Orden de complejidad | Notación |
|---|---|---|
| $f(n)=1$ | O(f(n)): Orden Constante | $O(f(n))=O(1)$ |
| $f(n)=\log(n)$ | O(f(n)): Orden Logarítmico | $O(f(n))=O(\log n)$ |
| $f(n)=n$ | O(f(n)): Orden Lineal | $O(f(n))=O(n)$ |
| $f(n)=n.\log(n)$ | O(f(n)): Orden Cuasi-lineal | $O(f(n))=O(n.\log n)$ |
| $f(n)=n^2$ | O(f(n)): Orden Cuadrático | $O(f(n))=O(n^2)$ |
| $f(n)=n^a$ | O(f(n)): Orden Polinomial (a>2) | $O(f(n))=O(n^a)$ |
| $f(n)=a^n$ | O(f(n)): Orden Exponencial (a>1) | $O(f(n))=O(a^n)$ |
| $f(n)=n!$ | O(f(n)): Orden Fatorial | $O(f(n))=O(n!)$ |

En general, con conocer los órdenes de complejidad de estas funciones es suficiente, ya que son sencillas y además, éstas aparecen con gran frecuencia, y es muy poco frecuente que se necesiten otras funciones (Mañas, 2017).

### 4.4.2. Un enfoque sencillo para su cálculo

Puede evaluarse la calidad de un algoritmo realizando una comparación con otros algoritmos o realizando una comparación de la complejidad del problema con alguna cota de complejidad conocida (Schaeffer, 2020).

Con esta idea, se presenta un enfoque empírico simple que permite determinar la complejidad de un programa basado en el trabajo de Mañas (2017). Este enfoque consiste en calcular los tiempos de resolución de un problema con algoritmos cuyo órden de complejidad ya está definido, y compararlo con los tiempos de resolución de nuestro programa. Las comparaciones determinarán funciones de complejidad mayor o menor a la de nuestro programa. Luego, el órden de complejidad de nuestro programa estará determinada por el órden de complejidad que se encuentra entre una función de complejidad mayor y una función de órden menor (basados en la Tabla 3). A este enfoque se lo denomina Análisis asintótico (Schaeffer, 2020).

# 5. Aplicación experimental: implementación de la propuesta

En este capítulo se presenta la implementación experimental de la propuesta técnica. Contiene condiciones de la experiencia, datos experimentales y resultados.

## 5.1. Presentación de la aplicación experimental

### 5.1.1. El resumen

Se obtuvieron dos imágenes de una misma escena con dos cámaras separadas, con orientación similar. A partir de las imágenes obtenidas, en procesos paralelos, se segmentaron las imágenes con la intención de separar los objetos del fondo (binarización). A los objetos se les obtuvo su borde, luego sus patrones de giro. Con ello, se realizó el reconocimiento de un objeto de una imagen en la otra imagen. Luego se realizó la puesta en correspondencia.

La aplicación experimental se realizó siguiendo las tareas presentadas en la propuesta técnica (ver ilustración 22) y mencionadas anteriormente. Los datos experimentales consisten en: mediciones de acierto en el apareo en ambos objetos para distintos valores de $\rho$. Se realizaron también, repetidos cronometrados de cada tarea y la presentación de los resultados obtenidos en cada medición.

Finalmente se presentan mediciones que permiten clasificar a la solución según su complejidad computacional.

## 5.2. Herramientas y equipamiento usado

### 5.2.1. Hardware

Para la adquisición de las imágenes se usaron dos cámaras USB marca Logitech modelo C170[18] separadas 230mm una de otra. Cada una de las imágenes obtenidas es de 640x480 píxeles.

El computador usado es una notebook HP Pavillion TS-15 con pantalla táctil de 15 pulgadas, 4 cores AMD A10-5745m, 12GB de memoria RAM.  La ilustración 30 muestra una foto del equipamiento usado.

18 Hoja de datos disponible en: http://www.cartimex.com/v2/pdf/960-000880.pdf Fecha de Consulta: 1/8/19

*Ilustración 31: Equipamiento usado para la experiencia.*

### 5.2.2. Software

El sistema operativo instalado en el computador es Ubuntu 20.04 LTS (long time support), una distribución con núcleo Linux de 2020.

Los algoritmos se programaron en Python[19] 2.7. Las principales librerías usadas son: *sys* y *os* para manejo de archivos, *numpy* y *math* para formatos numéricos y funciones matemáticas, *PIL* y *OpenCV* para el manejo de imágenes y *matplotlib* para la realización de gráficos.

### 5.2.3. Imágenes de trabajo y seteos

Se trabajó sobre dos pares de imágenes estéreo de 640 x 480 píxeles cada una tal como se muestra en la ilustración 32 (martillo) y en la ilustración 33 (zapato). Ambas imágenes se tomaron sobre un fondo blanco para simplificar la segmentación ya que el análisis del problema de la segmentación no forma parte de esta solución.

19 Sitio Oficial de Python: https://www.python.org/

(a) (b)

*Ilustración 32: Imágenes estéreo de un martillo. (a) Imagen izquierda. (b) Imagen derecha.*

(a) (b)

*Ilustración 33: Imágenes estéreo de un zapato. (a) Imagen izquierda. (b) Imagen derecha.*

Los parámetros que se usaron son: ancho de la cáscara convexa $\varepsilon = 4$ (píxeles) y la tolerancia en el hallazgo de puntos homólogos $\rho$ se la seteó entre 1% y 5% a intervalos del 1%.

Se usan ventanas de búsqueda iguales de 320 x 240 píxeles en ambos pares de imágenes estéreo (o bien: “pares estéreo”).

## 5.3. Resultados obtenidos

Se presentan resultados de cada tarea a partir de los dos pares de imágenes estéreo mostradas en las ilustraciones 32 y 33 (martillo y zapato respectivamente), según las etapas mostradas en la ilustración 22.

*Nota: A los fines de simplificar la notación y visualización, los resultados numéricos que presentan muchos decimales (por ejemplo: los valores de cada punto de cada patrón de giro) se los redondea a tres decimales.*

## 5.3.1. Interpretación de imágenes

### Lectura y binarización

Cada una de las imágenes posee 307200 píxeles.

**Par estéreo del martillo (ilustración 32):** El tamaño de los objetos es: 8165 en la imagen izquierda y 8403 en la imagen derecha.

**Par estéreo del zapato (ilustración 33):** El tamaño de los objetos es: 16838 píxeles en la imagen izquierda y 17346 píxeles en la imagen derecha.

### Puntos de borde

**Par estéreo del martillo (ilustración 32):** El objeto de la imagen izquierda posee 792 puntos de borde mientras que el de la imagen derecha posee 790 píxeles.

**Par estéreo del zapato (ilustración 33):** El objeto de la imagen izquierda posee 636 puntos de borde mientras que el de la imagen derecha posee 648 píxeles.

### Poligonalización

**Par estéreo del martillo (ilustración 32):** El objeto de la imagen izquierda poligonalizado posee 16 vértices mientras que el de la imagen derecha posee 15 vértices.

Los vértices del polígono del objeto de la imagen izquierda son: $VI_1$ ={(300 ; 268) , (326 ; 278) , (315 ; 306) , (330 ; 324) , (513 ; 342) , (578 ; 348) , (578 ; 375) , (562 ; 377) , (420 ; 351) , (405 ; 344) , (324 ; 337) , (302 ; 351) , (309 ; 372) , (296 ; 366) , (287 ; 329) , (299 ; 289)}

Los vértices del polígono del objeto de la imagen derecha son: $VD_1$ ={(42 ; 227) , (70 ; 236) , (64 ; 264) , (78 ; 281) , (190 ; 283) , (331 ; 290) , (328 ; 319) , (242 ; 309) , (169 ; 299) , (73 ; 296) , (59 ; 302) , (69 ; 329) , (57 ; 334) , (35 ; 296) , (41 ; 263)}

**Par estéreo del zapato (ilustración 33):** El objeto de la imagen izquierda poligonalizado posee 14 vértices mientras que el de la imagen derecha posee 14 vértices.

Los vértices del polígono del objeto de la imagen izquierda son: $VI_2$ = {(390 ; 242) , (417 ; 248) , (458 ; 272) , (514 ; 273) , (511 ; 354) , (440 ; 347) , (436 ; 333) , (351 ; 329) , (274 ; 294) , (250 ; 265) , (253 ; 257) , (292 ; 274) , (341 ; 277) , (388 ; 244)}

Los vértices del polígono del objeto de la imagen derecha son: $VD_2$ = {(187 ; 191) , (212 ; 198) , (243 ; 216) , (310 ; 213) , (307 ; 295) , (233 ; 292) , (225 ; 277) , (146 ; 281) , (73 ; 256) , (46 ; 221) , (57 ; 214) , (94 ; 230) , (140 ; 227) , (181 ; 196)}

### Patrón de giro

**Par estéreo del martillo (ilustración 32):** El patrón de giro del objeto de la imagen izquierda es: $PGI_1$ = {(0 ; 0) , (0.045 ; -1.824) , (0.080 ; -3.402) , (0.119 ; -2.333) , (0.148 ; -1.555) , (0.382 ; -1.549) , (0.465 ; -3.028) , (0.499 ; -4.474) , (0.520 ; -4.800) , (0.704 ; -5.035) , (0.725 ; -4.685) , (0.828 ; -4.032) , (0.861 ; -2.755) , (0.894 ; -4.957) , (0.905 ; -5.775) , (0.955 ; -6.455) , (1 ; -6.283)}

El patrón de giro del objeto de la imagen derecha es: $PGD_1$ = {(0 ; 0) , (0.046 ; -1.854) , (0.083 ; -3.325) , (0.119 ; -2.425) , (0.147 ; -1.561) , (0.289 ; -1.593) , (0.468 ; -3.217) , (0.505 ; -4.800) , (0.615 ; -4.821) , (0.708 ; -4.712) , (0.823 ; -4.280) , (0.849 ; -2.759) , (0.885 ; -4.290) , (0.902 ; -5.731) , (0.958 ; -6.435) , (1 ; -6.283)}

**Par estéreo del zapato (ilustración 33):** El patrón de giro del objeto de la imagen izquierda es: $PGI_2$ ={(0 ; 0) , (0.004 ; -1.004) , (0.046 ; -1.315) , (0.117 ; -0.803) , (0.201 ; -2.393) , (0.323 ; -4.025) , (0.430 ; -5.219) , (0.451 ; -3.974) , (0.579 ; -4.354) , (0.707 ; -4.806) , (0.763 ; -5.856) , (0.776 ; -7.479) , (0.840 ; -7.130) , (0.914 ; -6.456) , (1 ; -6.283)}

El patrón de giro del objeto de la imagen derecha es: $PGD_2$ ={(0 ; 0) , (0.012 ; -0.968) , (0.051 ; -1.221) , (0.105 ; -0.650) , (0.207 ; -2.302) , (0.331 ; -3.877) , (0.443 ; -4.917) , (0.469 ; -3.786) , (0.588 ; -4.166) , (0.705 ; -4.750) , (0.772 ; -6.411) , (0.792 ; -7.386) , (0.853 ; -6.913) , (0.922 ; -6.330) , (1 ; -6.283)}

En la ilustración 34 se muestra una gráfica donde se presentan los dos patrones.

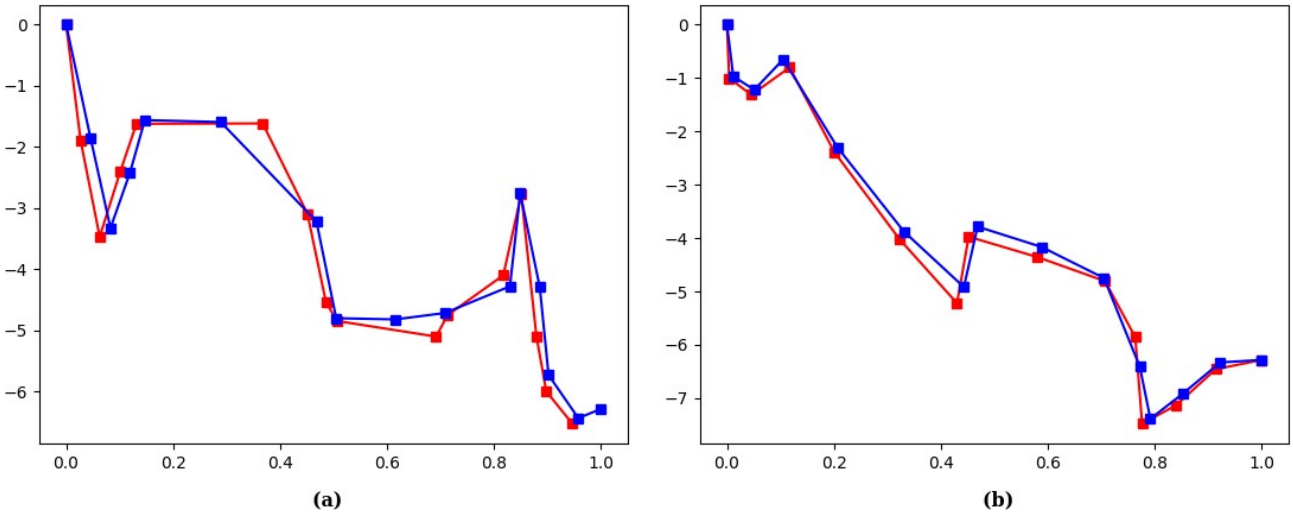



*Ilustración 34: Comparación entre los patrones de dos pares estéreo. (a) Martillo, (b) zapato.*

## 5.3.2. Apareo

### Reconocimiento de objetos

El reconocimiento de objetos se realiza tras aplicar la ecuación (39) a los patrones de giro de los objetos de la imagen izquierda comparándolos uno a uno con los patrones de giro de la imagen derecha. Se define el valor del umbral: $\mu$=0,1000, y se toman las medias entre los patrones presentados en el punto anterior.

**Reconocimiento de objetos en el par de imágenes estéreo del martillo (ilustración 32):**

Distancia entre los patrones: $D_1$ = 0,044, con $D_1 < \mu$. Ello significa que el objeto presente en la imagen de la derecha fue hallado en la imagen izquierda.

**Reconocimiento de objetos en el par de imágenes estéreo del zapato (ilustración 33):**

Distancia entre los patrones: $D_2 = 0{,}080$, con $D_2 < \mu$. Ello significa que el objeto presente en la imagen de la derecha fue hallado en la imagen izquierda.

## Puesta en correspondencia

El reconocimiento de un objeto en dos imágenes se logra tras hallar un par de patrones de giro de distintas imágenes que cumplan con la ecuación (39). Tras ello, el punto de inicio de ambos patrones queda definido. El resto de los puntos pueden hallarse recorriendo cada punto del perímetro de uno de los objetos, hallando la distancia entre los valores del patrón en ambas imágenes en ese punto, y comparándolo con un valor $\rho$.

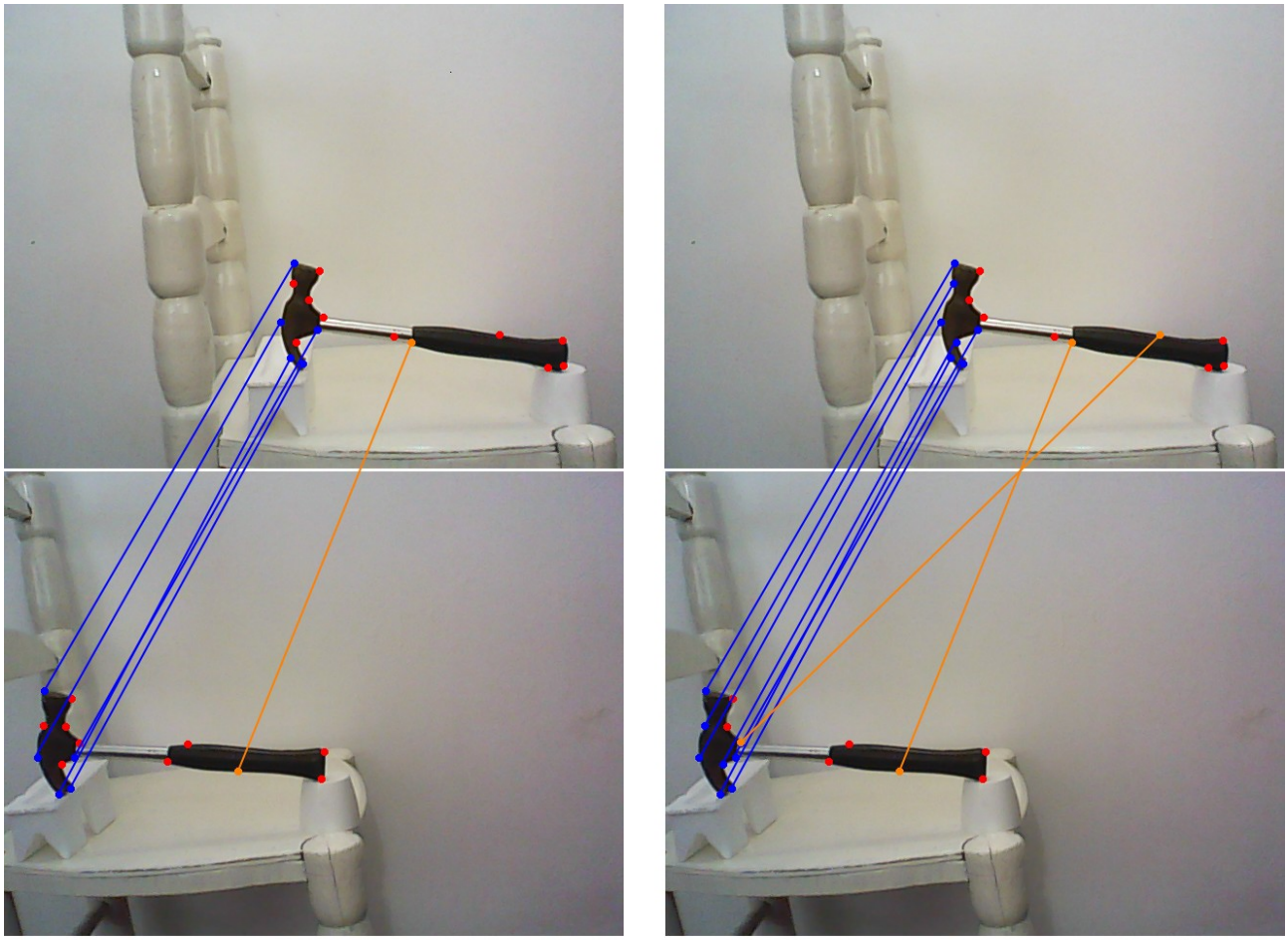
Se presenta a continuación los resultados de la puesta en correspondencia de ambos pares de imágenes estéreo (o pares estéreo) presentados en ilustraciones 32 y 33 y cinco valores de $\rho$. Las figuras 35 a 38 muestra los resultados hallados en los pares estéreo.

**Puesta en correspondencia del par estéreo del martillo para $\rho$=0,01:** El conjunto de puntos homólogos es: $PH_{m1}$ = {([300 ; 268] , [42 ; 227]) , ([420 ; 351] , [242 ; 309]) , ([324 ; 337] , [73 ; 296]) , ([309 ; 372] , [69 ; 329]) , ([296 ; 366] , [57 ; 334]) , ([287 ; 329] , [35 ; 296])}.

Cantidad de pares homólogos: 6.

**Puesta en correspondencia del par estéreo del martillo para $\rho$=0,02:** El conjunto de puntos homólogos es: $PH_{m2}$ = {([300 ; 268] , [42 ; 227]) , ([513 ; 342] , [78 ; 81]) , ([420 ; 351] , [242 ; 309]) , ([324 ; 337] , [73 ; 296]) , ([302 ; 351] , [59 ; 302]) , ([309 ; 372] , [69 ; 329]) , ([296 ; 366] , [57 ; 334]) , ([287 ; 329] , [35 ; 296]) , ([299 ; 289] , [41 ; 263])}.

Cantidad de pares homólogos: 9.



(a) (b)

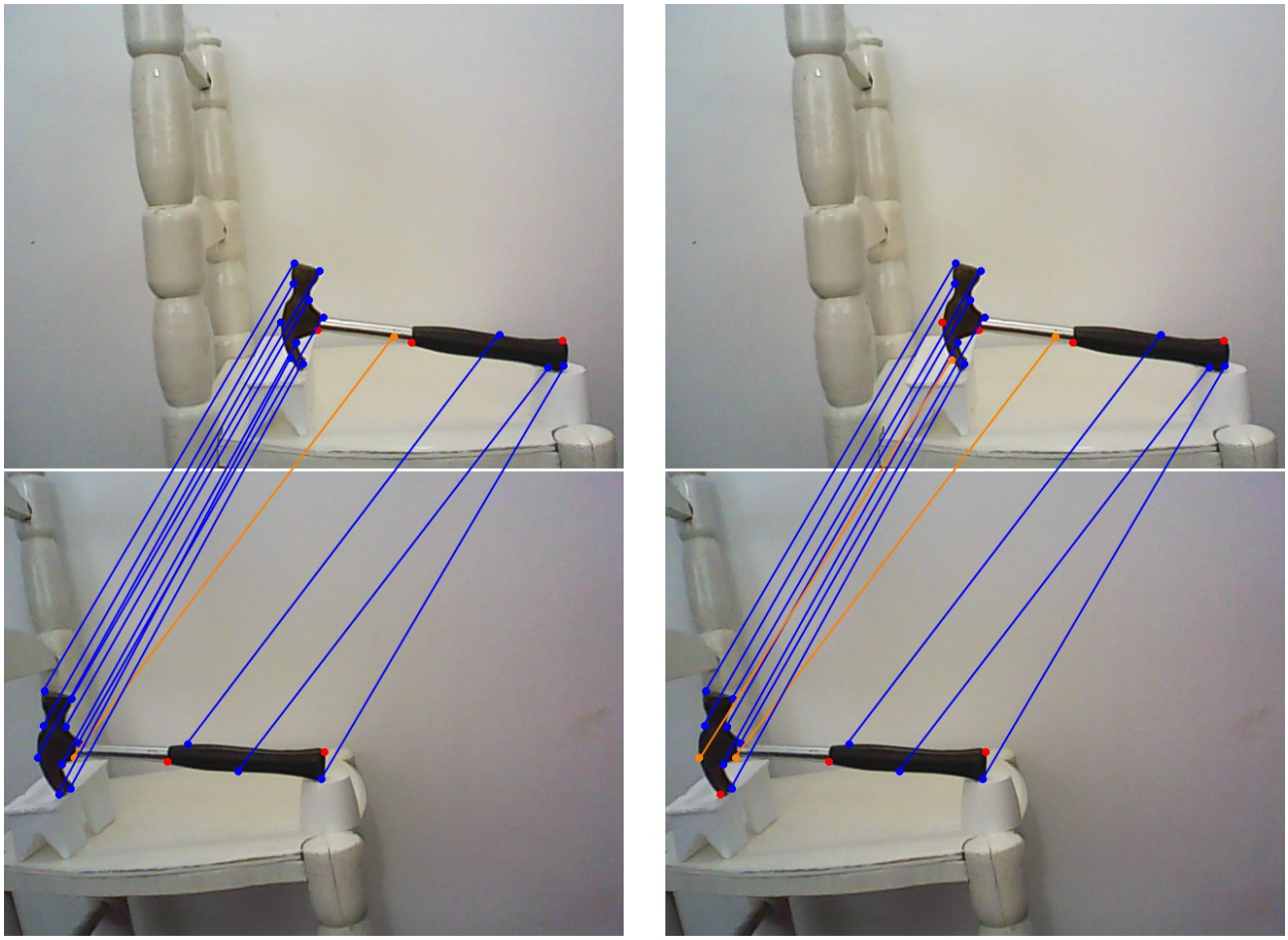
*Ilustración 35: Apareo del par estéreo del martillo. (a) Para ρ=0,01 (b) para ρ=0,02.*

**Puesta en correspondencia del par estéreo del martillo para ρ=0,03:** El conjunto de puntos homólogos es: $PH_{m3}$ = {([300 ; 268] , [42 ; 227]) , ([326 ; 278] , [70 ; 236]) , ([315 ; 306] , [64 ; 264]) , ([330 ; 324] , [78 ; 281]) , ([513 ; 342] , [190 ; 283]) , ([578 ; 375] , [328 ; 319]) , ([562 ; 377] , [242 ; 309]) , ([405 ; 344] , [73 ; 296]) , ([302 ; 351] , [59 ; 302]) , ([309 ; 372] , [69 ; 329]) , ([296 ; 366] , [57 ; 334]) , ([287 ; 329] , [35 ; 296]) , ([299 ; 289] , [41 ; 263])}.

Cantidad de pares homólogos: 13.

**Puesta en correspondencia del par estéreo del martillo para ρ=0,04:** El conjunto de puntos homólogos es: $PH_{m4}$ = {([300 ; 268] , [42 ; 227]) , ([326 ; 278] , [70 ; 236]) , ([315 ; 306] , [64 ; 264]) , ([330 ; 324] , [78 ; 281]) , ([513 ; 342] , [190 ; 283]) , ([578 ; 375] , [328 ; 319]) , ([562 ; 377] , [242 ; 309]) , ([405 ; 344] , [73 ; 296]) , ([302 ; 351] , [59 ; 302]) , ([309 ; 372] , [69 ; 329]), ([296 ; 366], [35 ; 296]), ([299 ; 289], [41 ; 263])}.

Cantidad de pares homólogos: 12.



(a) (b)

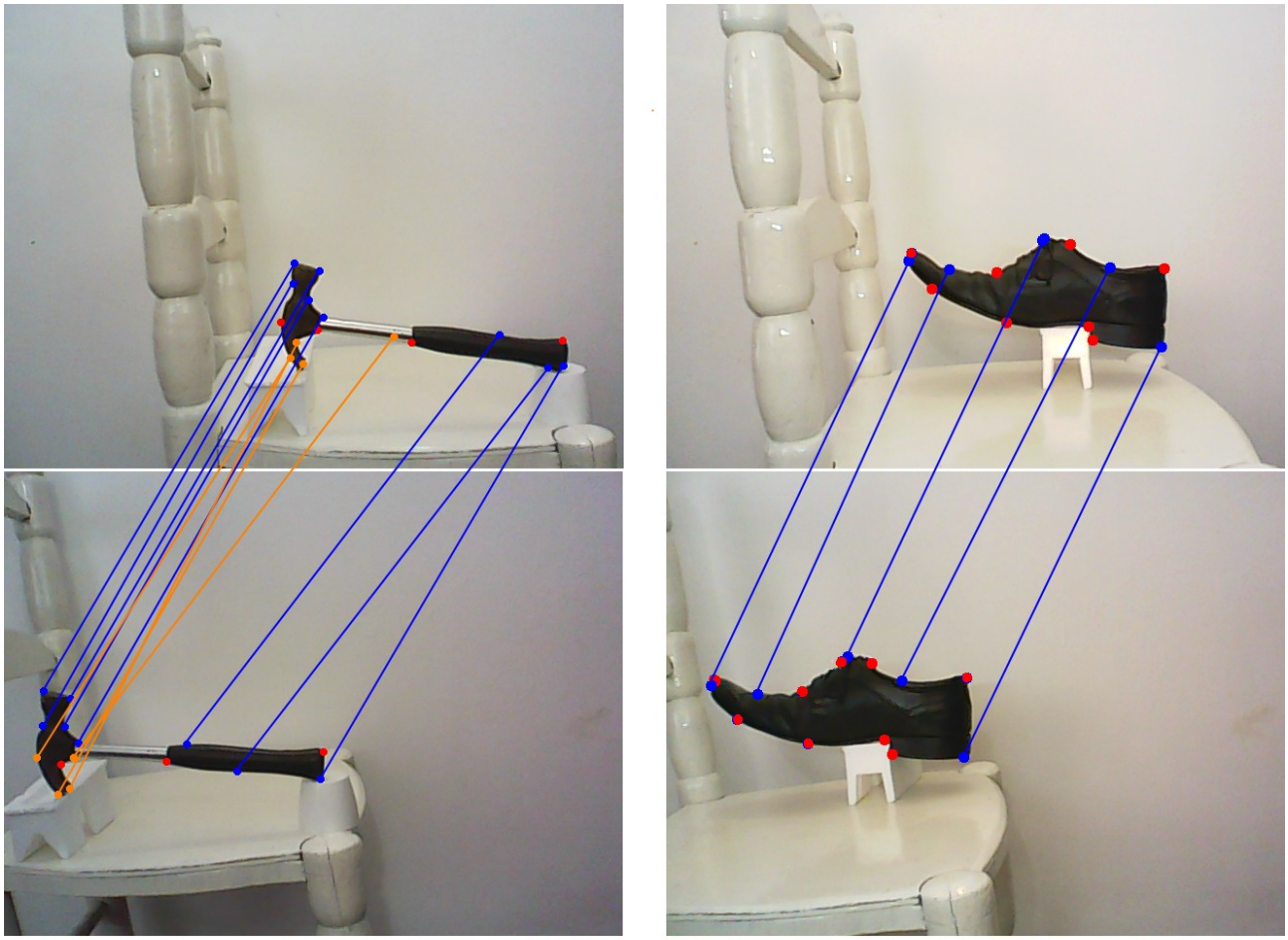
*Ilustración 36: Apareo del par estéreo del martillo. (a) Para ρ=0,03 (b) para ρ=0,04.*

**Puesta en correspondencia del par estéreo del martillo para ρ=0,05:** El conjunto de puntos homólogos es: $PH_{m5}$ = {([300 ; 268] , [42 ; 227]) , ([326 ; 278] , [70 ; 236]) , ([315 ; 306] , [64 ; 264]) , ([330 ; 324] , [78 ; 281]) , ([513 ; 342] , [190 ; 283]) , ([578 ; 375] , [328 ; 319]) , ([562 ; 377] , [242 ; 309]) , ([405 ; 344] , [73 ; 296]) , ([302 ; 351] , [69 ; 329]) , ([309 ; 372] , [57 ; 334]) , ([296 ; 366] , [35 ; 296]) , ([299 ; 289] , [41 ; 263])}.

Cantidad de pares homólogos: 12.

**Puesta en correspondencia del par estéreo del zapato para r=0,01:** El conjunto de puntos homólogos es: $PH_{z1}$ ={([390 ; 242] , [187 ; 191]) , ([458 ; 272] , [243 ; 216]) , ([511 ; 354] , [307 ; 295]) , ([250 ; 265] , [46 ; 221]) , ([292 ; 274] , [94 ; 230])}.

Cantidad de pares homólogos: 5.



(a) (b)

*Ilustración 37: Apareo del par estéreo (a) del martillo para ρ=0,05 y (b) del zapato para ρ=0,01.*

**Puesta en correspondencia del par estéreo del zapato para ρ=0,02:** El conjunto de puntos homólogos es: $PH_{z2}$ ={([390 ; 242] , [212 ; 198]) , ([458 ; 272] , [243 ; 216]) , ([514 ; 273] , [310 ; 213]) , ([511 ; 354] , [307 ; 295]) , ([440 ; 347] , [233 ; 292]) , ([436 ; 333] , [225 ; 277]) , ([274 ; 294] , [73 ; 256]) , ([250 ; 265] , [46 ; 221]) , ([253 ; 257] , [49 ; 215]) , ([292 ; 274] , [94 ; 230]) , ([341 ; 277] , [140 ; 227]) , ([388 ; 244] , [181 ; 196])}.

Cantidad de pares homólogos: 12.

**Puesta en correspondencia del par estéreo del zapato para ρ=0,03:** El conjunto de puntos homólogos es: $PH_{z3}$ = {([390 ; 242] , [212 ; 198]) , ([458 ; 272] , [243 ; 216]) , ([514 ; 273] , [310 ; 213]) , ([511 ; 354] , [307 ; 295]) , ([440 ; 347] , [233 ; 292]) , ([436 ; 333] , [225 ; 277]) , ([351 ; 329] , [146 ; 281]) , ([274 ; 294] , [73 ; 256]) , ([250 ; 265] , [46 ; 221]) , ([253 ; 257] , [94 ; 230]) , ([341 ; 277] , [140 ; 227]), ([388 ; 244] , [181 ; 196])}.

Cantidad de pares homólogos: 12.

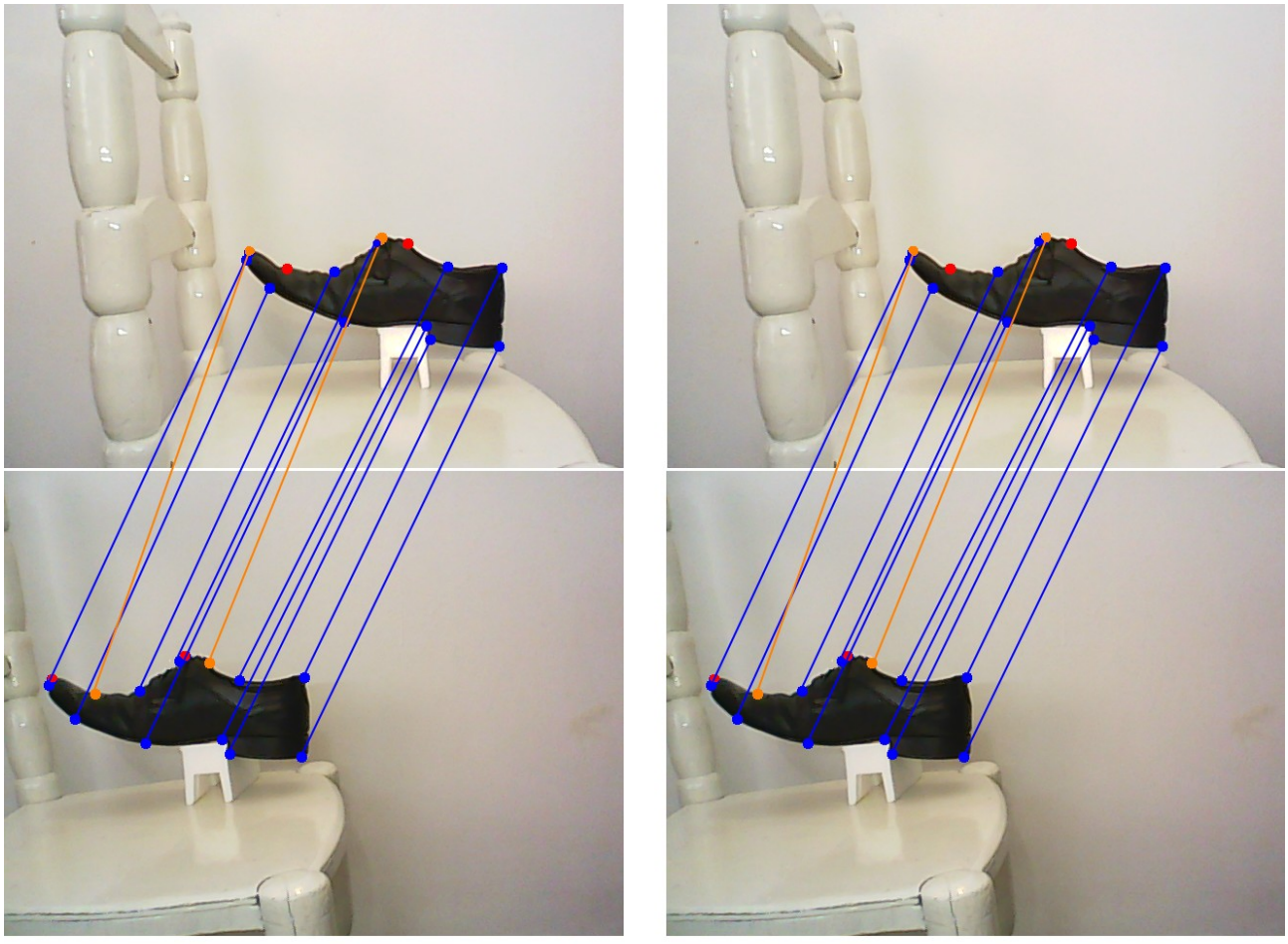
**Puesta en correspondencia del par estéreo del zapato para ρ=0,04:** El conjunto de puntos homólogos es: $PH_{z4}$ ={([390 ; 242] , [212 ; 198]) , ([458 ; 272] , [243 ; 216]) , ([514 ; 273] , [310 ; 213]) , ([511 ; 354] , [307 ; 295]) , ([440 ; 347] , [233 ; 292]) , ([436 ; 333] , [225 ; 277]) ,

([274 ; 294] , [73 ; 256]) , ([250 ; 265] , [46 ; 221]) , ([253 ; 257] , [49 ; 215]) , ([292 ; 274] , [94 ; 230]) , ([341 ; 277] , [140 ; 227]) , ([388 ; 244] , [181 ; 196])}.

Cantidad de pares homólogos: 12.

**Puesta en correspondencia del par estéreo del zapato para ρ=0,05:** El conjunto de puntos homólogos es: $PH_{z5}$ = {([390 ; 242] , [212 ; 198]) , ([458 ; 272] , [243 ; 216]) , ([514 ; 273] , [310 ; 213]) , ([511 ; 354] , [307 ; 295]) , ([440 ; 347] , [233 ; 292]) , ([436 ; 333] , [225 ; 277]) , ([274 ; 294] , [73 ; 256]) , ([250 ; 265] , [46 ; 221]) , ([253 ; 257] , [49 ; 215]) , ([292 ; 274] , [94 ; 230]) , ([341 ; 277] , [140,227]) , ([388 ; 244] , [181 ; 196])}.

Cantidad de pares homólogos: 12.



(a) (b)

*Ilustración 38: Apareo del par estéreo del zapato. (a) Para ρ=0,02 (b) para ρ=0,03.*

## 5.3.3. Análisis de confianza del proceso

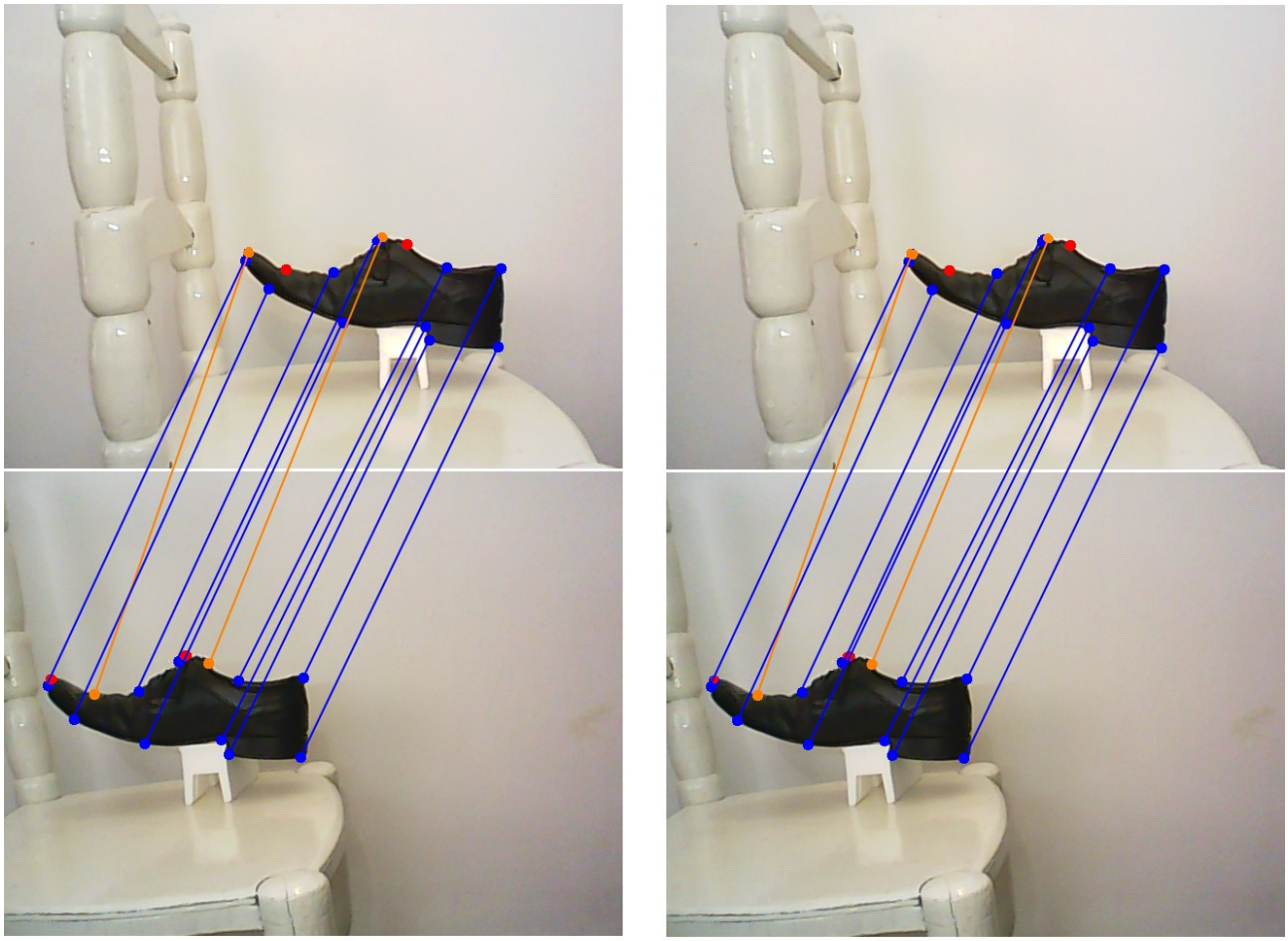
En las ilustraciones 35 a 38 se presenta la puesta en correspondencia entre pares estéreo de imágenes de dos objetos: un martillo (ilustración 32) y un zapato (ilustración 33).

Con un pequeño círculo de color se representa a cada uno de los vértices obtenidos en el proceso de poligonalización. En ambos pares se observa que casi la totalidad de los vértices del objeto de

una de las imágenes tiene su correspondiente en la otra imagen. En algunos casos puede observarse, además, que algún vértice de la poligonalización presenta su homólogo en la otra imagen en una ubicación que difiere levemente de la esperada.

En el caso del par estéreo de imágenes del martillo, el resultado de la poligonalización presenta 15 y 16 vértices respectivamente, por lo que el proceso de apareo podrá obtener a lo sumo 15 pares de puntos homólogos.

Para cada uno de los casos (para cada par estéreo de cada objeto y para los distintos valores de $\rho$), se presentaron anteriormente los resultados arrojados con el algoritmo, los cuales se contrastaron mediante un análisis visual.



(a) (b)

*Ilustración 39: Apareo del par estéreo del zapato. (a) Para $\rho$=0,04 (b) para $\rho$=0,05.*

En cada caso, en las mismas figuras, los vértices que fueron apareados correctamente se pintaron en azul y se unieron con una línea continua del mismo color. Los vértices que fueron apareados incorrectamente se pintaron de amarillo y se unieron con una línea continua del mismo color. Los vértices que no fueron apareados, se pintaron de rojo.

En base a lo anterior, se elaboró la tabla 4 que contiene datos experimentales que permiten analizar la confiabilidad del proceso de búsqueda de puntos homólogos.

Se presentan dos indicadores que pretenden describir la confiabilidad del proceso: el *Indice de apareos* y el índice de c*onfianza de los apareos*.

El i*ndice de apareos* ($I_A$) indica la proporción de los vértices del objeto poligonalizado que son apareados por el algoritmo.

$$I_A = \frac{cantidad\ de\ vértices\ apareados}{cantidad\ de\ vértices} \tag{41}$$

El índice de *confianza de los apareos* ($C_A$) se presenta como un indicador que refleja la bondad en el proceso de apareo. Del conjunto de vértices que han sido apareados, $C_A$ indica la proporción de ellos que resultaron correctos tras la comprobación visual.

$$C_A = \frac{cantidad\ de\ vertices\ apareados}{cantidad\ de\ vertices\ apareados\ correctamente} \tag{42}$$

*Tabla 4: Análisis de la confiabilidad del proceso de apareo.*

| Objeto | ρ | Vertices totales | Vertices apareados | Apareos Correctos | Indice de Apareos | Confianza en Apareos |
|---|---|---|---|---|---|---|
| Zapato | 0,01 | 14 | 5 | 5 | 0,36 | 1,00 |
| Martillo | 0,01 | 15 | 6 | 5 | 0,40 | 0,83 |
| Zapato | 0,02 | 14 | 12 | 10 | 0,86 | 0,83 |
| Martillo | 0,02 | 15 | 9 | 7 | 0,60 | 0,78 |
| Zapato | 0,03 | 14 | 12 | 10 | 0,86 | 0,83 |
| Martillo | 0,03 | 15 | 13 | 12 | 0,87 | 0,92 |
| Zapato | 0,04 | 14 | 12 | 10 | 0,86 | 0,83 |
| Martillo | 0,04 | 15 | 12 | 10 | 0,80 | 0,83 |
| Zapato | 0,05 | 14 | 12 | 10 | 0,86 | 0,83 |
| Martillo | 0,05 | 15 | 12 | 8 | 0,80 | 0,67 |
| **Promedio** | | | | | **0,73** | **0,84** |
| **Desvío Estándar** | | | | | **0,20** | **0,09** |
| **Coef. Var. de Pearson** | | | | | **0,27** | **0,10** |

En el total del conjunto de datos, el $I_A$ no presenta datos homogéneos, por lo que su promedio no es un indicador de tendencia central adecuado. Presenta valores bajos para $\rho$ pequeños. Pero a partir de $\rho$=0,03 sus valores mantienen homogeneidad (con coeficiente de variación de Pearson de 0,04) alrededor del promedio cuyo valor es del 84%. Esto es: a *partir de $\rho$=0,03 el algoritmo logra aparear (en promedio) el 84% de los vértices.*

La confianza en los apareos ($C_A$) en cambio, presenta datos homogéneos (con coeficiente de variación de Pearson de 0,10). Por ello, puede decirse que el promedio de la confianza en los apareos ($C_A$) es un indicador de tendencia central adecuado. Esto es: *El algoritmo logra aparear correctamente el 84% de los puntos que han sido apareados.*

*<u>Conclusión:</u> Para $\rho$>0,02 el algoritmo logra aparear correctamente (en promedio) aproximadamente el 70% de todos los vértices del polígono que simplifican a las curvas de bordes de los objetos presentes en ambas imágenes estéreo.*

## 5.3.4. Análisis del desempeño

En esta sub-sección se analizan tiempos de cómputo del algoritmo.

Se realizaron 5 mediciones por cada par de imágenes estéreo del tiempo para la obtención de puntos homólogos. Cada experimento se lo subdividió en las etapas y se cronometró cada tarea. La tabla 5 presenta el análisis de las mediciones tomadas.

Se tomó nota de los tiempos para la realización de cada una de las tareas presentadas en la ilustración 22: Segmentación de imagen #1, Obtención de puntos de Borde de Imagen 1#, Poligonalización de imagen #1, Cálculo de Patrón de Giro de imagen #1, Segmentación de imagen #2, Obtención de puntos de Borde de Imagen 2#, Poligonalización de imagen #2, Cálculo de Patrón de Giro de imagen #2, Reconocimiento, Puesta en Correspondencia.

Dada la paralelización de los procesos en ambas imágenes, para el cálculo del tiempo total, se toma en cuenta el tiempo de proceso mayor entre ambos.

En la tabla 5 pueden observarse los tiempos parciales por cada tarea.

*Tabla 5: Tiempos de cómputo del proceso.*

| ***Número de Prueba*** | ***Objeto*** | ***Tiempos de Cómputo (en segundos)*** | | | | | | | | | | ***Tiempo Total*** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | ***Segmenta_ción1*** | ***Obtención de Bordes1*** | ***Poligonali_zación1*** | ***Cálculo de Patrón1*** | ***Segmenta_ción2*** | ***Obtención de Bordes2*** | ***Poligonali_zación12*** | ***Cálculo de Patrón2*** | ***Reconoci_miento*** | ***Puesta en Correspond.*** | |
| *1* | *Martillo* | *0,191* | *0,074* | *0,085* | *0,134* | *0,169* | *0,069* | *0,051* | *0,153* | *0,047* | *0,039* | *0,570* |
| *2* | *Martillo* | *0,197* | *0,072* | *0,081* | *0,134* | *0,149* | *0,071* | *0,050* | *0,149* | *0,038* | *0,036* | *0,558* |
| *3* | *Martillo* | *0,183* | *0,070* | *0,085* | *0,125* | *0,157* | *0,071* | *0,050* | *0,150* | *0,039* | *0,031* | *0,533* |
| *4* | *Martillo* | *0,179* | *0,071* | *0,086* | *0,125* | *0,161* | *0,070* | *0,048* | *0,096* | *0,038* | *0,031* | *0,530* |
| *5* | *Martillo* | *0,187* | *0,076* | *0,086* | *0,126* | *0,151* | *0,069* | *0,050* | *0,150* | *0,042* | *0,033* | *0,550* |
| *6* | *Zapato* | *0,235* | *0,088* | *0,067* | *0,101* | *0,232* | *0,068* | *0,068* | *0,088* | *0,045* | *0,044* | *0,580* |
| *7* | *Zapato* | *0,268* | *0,069* | *0,065* | *0,103* | *0,217* | *0,079* | *0,071* | *0,082* | *0,043* | *0,045* | *0,593* |
| *8* | *Zapato* | *0,285* | *0,065* | *0,067* | *0,101* | *0,218* | *0,077* | *0,072* | *0,083* | *0,046* | *0,044* | *0,608* |
| *9* | *Zapato* | *0,143* | *0,076* | *0,069* | *0,104* | *0,221* | *0,071* | *0,069* | *0,086* | *0,046* | *0,045* | *0,538* |
| *10* | *Zapato* | *0,254* | *0,068* | *0,068* | *0,100* | *0,235* | *0,068* | *0,079* | *0,078* | *0,043* | *0,046* | *0,579* |
| ***Promedio:*** | | ***0,212*** | ***0,073*** | ***0,076*** | ***0,115*** | ***0,191*** | ***0,071*** | ***0,061*** | ***0,112*** | ***0,043*** | ***0,039*** | ***0,564*** |
| ***Desvío estándar:*** | | ***0,046*** | ***0,006*** | ***0,009*** | ***0,015*** | ***0,036*** | ***0,004*** | ***0,012*** | ***0,034*** | ***0,003*** | ***0,006*** | ***0,026*** |
| ***Coef. Var. Pearson*** | | ***0,215*** | ***0,087*** | ***0,123*** | ***0,127*** | ***0,190*** | ***0,052*** | ***0,197*** | ***0,304*** | ***0,080*** | ***0,157*** | ***0,047*** |

Los procesos: Obtención de bordes de imagen #1, obtención de bordes de imagen #2 y reconocimiento presentan homogeneidad (coeficiente de variación de Pearson CV $\leq$ 0,10).

Los procesos: segmentación de imagen #1, poligonalización de imagen #1, cálculo de patrón de giro de imagen #1, segmentación de imagen #2, poligonalización de imagen #2, cálculo de patrón de giro de imagen #2, puesta en correspondencia no presentan homogeneidad en sus datos (coeficiente de variación de Pearson > 0,10).

Sin embargo, el proceso global presenta homogeneidad en los datos (coeficiente de variación de Pearson CV = 0,047), por lo que puede considerarse a su promedio como un indicador de tendencia central adecuado.

*Conclusión: El algoritmo logra la obtención de puntos homólogos en un par de imágenes estéreo de un objeto en un promedio de 0,564 segundos en las condiciones presentadas.*

### 5.3.5. Limitaciones de la esta implementación

El análisis de desempeño o perfomance anteriormente presentado podría mostrar resultados notoriamente mejores sin alguna de las siguientes limitaciones:

- El lenguaje de programación elegido, Python, es un lenguaje que resulta indefectiblemente más lento que C, aunque con este lenguaje resulta fácil programar y ejecutar prototipos.
- La programación de los algoritmos no ha sido optimizada. La implementación de cualquier técnica de optimización mejoraría su tiempo de cómputo.
- Algunas de las tareas como ser la poligonalización o el reconocimiento podrían realizarse de una manera aún más eficiente.
- La puesta en correspondencia se realiza sobre puntos de borde, solamente.

## 5.4. Análisis empírico de la complejidad computacional de la solución

### 5.4.1. Presentación resumida de la experiencia

En el apartado 4.4 se presentó un enfoque empírico para el cálculo del órden de complejidad de la solución aquí presentada. Para ello se necesita calcular los tiempos de cómputo para funciones de referencia y compararlos con los tiempos de cómputo del programa que ejecuta la solución presentada.

En una consulta en el sitio Stack Overflow en Español (Abulafia, 2020) se presentan programas codificados en Python aquí llamados EM1 y EM2 con complejidad lineal y cuadrática respectivamente. Esto es: $O(EM1(n))=O(n)$ y $O(EM2(n))=O(n^2)$. Estos programas se seleccionaron para realizar el análisis asintótico de complejidad (Schaeffer, 2020). El programa que ejecuta la solución aquí presentada (desarrollado en Python) es llamado P. Entonces, el órden de complejidad de ese programa se denotará $O(P(n))$.

### 5.4.2. Datos experimentales

Respecto del programa P, su variable de análisis es n: cantidad de píxeles que tiene la imágen a analizar. Esta experiencia se realizó con la imagen “Zapato” usada previamente en este capítulo. A esta imagen se le realizó varios escalados uniformes de modo de presentar un conjunto de imágenes de donde extraer una serie a escala (cuasi) lineal de valores de n.

Para un análisis más profundo, se varía también, el valor de la tolerancia en la poligonalización $\varepsilon$: Para los valores de $\varepsilon=1$, $\varepsilon=3$, $\varepsilon=5$, llamaremos a las variantes del programa: PZ1(n), PZ3(n) y PZ5(n). Así, los órdenes de complejidad se denominan O(PZ1(n)), O(PZ3(n)) y O(PZ5(n)).

La Tabla 6 muestra los datos recopilados: el tiempo de ejecución de los programas definidos anteriormente para distintos valores de n.

*Tabla 6: Tiempos de ejecución de las diferentes funciones en función de n.*

| Tamaño Imagen | n (píxeles) | Tiempos de procesamiento (en s) | | | | |
|---|---|---|---|---|---|---|
| | | EM2(n) | EZ1(n) | EZ3(n) | EZ5(n) | EM1(n) |
| 106x80 | 8480 | 4,462 | 0,044 | 0,027 | 0,026 | 0,009 |
| 144x108 | 15552 | 14,406 | 0,085 | 0,042 | 0,041 | 0,019 |
| 176x132 | 23232 | 31,901 | 0,070 | 0,046 | 0,044 | 0,031 |
| 200x150 | 30000 | 53,911 | 0,066 | 0,051 | 0,057 | 0,033 |
| 228x171 | 38988 | --- | 0,065 | 0,062 | 0,059 | 0,039 |
| 250x187 | 46750 | --- | 0,083 | 0,071 | 0,069 | 0,048 |
| 266x202 | 53732 | --- | 0,082 | 0,075 | 0,074 | 0,059 |
| 288x216 | 62208 | --- | 0,088 | 0,087 | 0,087 | 0,062 |
| 304x228 | 69312 | --- | 0,094 | 0,088 | 0,900 | 0,072 |
| 320x240 | 76800 | --- | 0,125 | 0,100 | 0,103 | 0,079 |

*Nota: En las mediciones de EM2(n) aparecen leyendas "---". Al ser EM2(n) una función cuadrática, solo se presentan los primeros valores de "n" ya que muy pronto toma valores muy elevados. Por ello, su resultado, carece de interés y se omite intencionalmente.*

La Ilustración 40 presenta la comparación entre los tiempos de procesamiento de las funciones PZ1(n), PZ3(n) y PZ5(n) en comparación con funciones de referencia EM1(n) y EM2(n).

En la Ilustración 40 puede observarse que para todos los valores de n analizados, las funciones PZ1(n), PZ3(n) y PZ5(n) están acotadas por la funciónones EM1(n) y EM2(n).

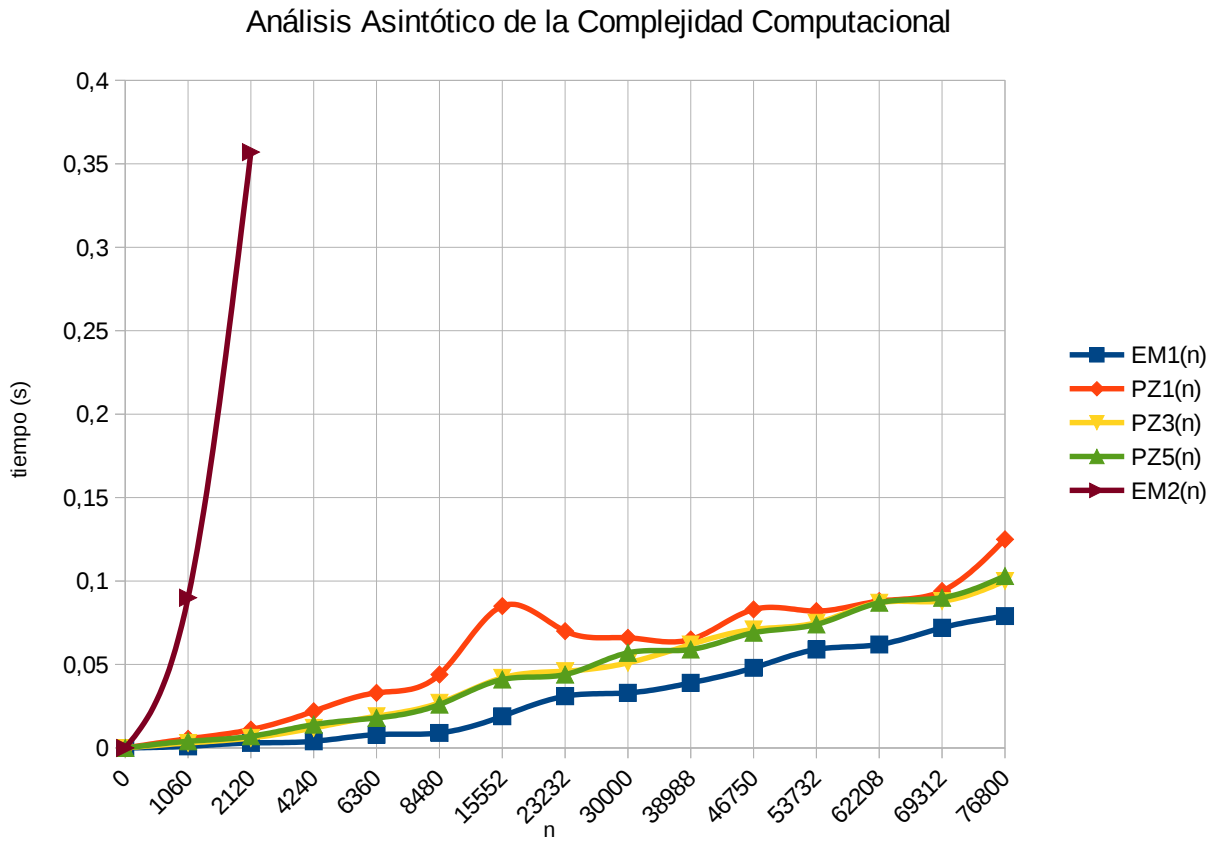



*Ilustración 40: Comparación de los tiempos de ejecución de las diferentes funciones.*

### 5.4.3. La complejidad computacional de la solución

Al analizar la Ilustración 10 y los datos de la tabla 6 puede observarse que:

$$\begin{cases} \forall n \in \mathbb{N}: EM2(n) > PZ1(n) > EM1(n) \Rightarrow O(PZ1(n)) = n \log n \\ \forall n \in \mathbb{N}: EM2(n) > PZ3(n) > EM1(n) \Rightarrow O(PZ3(n)) = n \log n \\ \forall n \in \mathbb{N}: EM2(n) > PZ3(n) > EM1(n) \Rightarrow O(PZ3(n)) = n \log n \end{cases}$$

| *Conclusión: la complejidad computacional de la solución presentada es cuasi-lineal.* |
|---|

*Nota: Si bien el análisis asintótico muestra que la complejidad de la solución propuesta es "cuasi-lineal", es fácil observar que las funciones se encuentran mucho más próximas a la función de referencia de complejidad lineal que a la cuadrática.*

## 5.5. Los resultados de la experiencia

Los resultados aquí presentados y en las condiciones de esta experiencia (lograr el apareo en 0,564 segundos) permitirían lograr el seguimiento de un objeto en tiempo real. Los resultados podrían mejorarse si se llegara a implementar alguna mejora que evite las limitaciones en el desempeño o performance presentadas.

Lo que aquí se logra es la rápida puesta en correspondencia de algunos pocos puntos del objeto, permitiendo la reconstrucción tridimensional de los bordes. Así se obtendrían (mediante métodos como mínimos cuadrados) coordenadas tridimensionales de figuras planas que aproximan a los objetos.

Y al poseer coordenadas tridimensionales (en unidades de longitud), es más sencillo lograr el reconocimiento de algunos objetos mediante abstracciones en el espacio en unidades espaciales. Por ejemplo, en una aplicación que busca personas vivas, se reconoce una forma humana erguida obtenida mediante el patrón de giro que posee 37°C (segmentado mediante una cámara térmica) y una altura de 1,70m (esto logrado tras la reconstrucción tridimensional en unidades de longitud): podría decirse que es un ser humano. Por el contrario, un hallazgo similar puede rechazarse rápidamente si su altura es de 4m, por más que la forma de los bordes se correspondan con las de una forma humana, ya que no existen personas de esa altura. Este tipo de abstracciones de nivel superior pueden lograrse muy rápidamente con un estudio profundo del dominio del problema y puede ser de gran utilidad en implementaciones como ser: robótica para búsqueda y rescate en áreas de catástrofe (Greer et al., 2002; Naidoo et al., 2015; Murphy & Stover, 2008; Kamlofsky et al., 2018).

# 6. Conclusiones

En este trabajo se presentó un algoritmo que logra la puesta en correspondencia de objetos en un par de imágenes estéreo de una misma escena rápidamente, lo cual puede ser compatible con requerimientos de tiempo real: se logra la obtención de puntos homólogos en aproximadamente medio segundo, en un prototipo sin optimizar en las condiciones de esta experiencia.

El cálculo de la complejidad computacional de esta solución permite suponer que es posible generalizar la afirmación anterior independizandola de las condiciones de esta experiencia.

Se presentó un adecuado detalle técnico del algoritmo, junto con datos experimentales e indicadores que permiten medir su alcance.

Una conclusión subyacente apunta a presentar al enfoque topológico de análisis de bordes como un enfoque simple, sencillo y muy eficiente, para la identificación de objetos en imágenes digitales y así usarse en soluciones de visión robótica.

# Trabajos Futuros

El esquema presentado en este trabajo posee aspectos muy novedosos en el marco de la Visión 3D, orientado a su procesamiento en tiempo real, el cual es un tema abierto en el estado del arte actual.

En el capítulo 5 se presentó la implementación de la solución, con sus ventajas y limitaciones. Entre estas últimas, se presentan limitaciones en la performance. Un primer trabajo futuro puede tratar acerca de solucionar las limitaciones aquí presentadas.

Otra limitación presentada (no relacionada con la performace) se refiere a que la obtención de puntos homólogos para modelización tridimensional se realiza sobre los puntos de borde de los objetos. En el trabajo presentado por Rosenfeld (1979), se presentó el algoritmo BF4/8 para obtener los bordes de un objeto. A partir de ello, aplicando los procesos presentados en este trabajo, se obtendrá el apareo de vértices del borde. Sin embargo, en el mismo trabajo, Rosenfeld también se presentó un algoritmo para afinamiento de formas a sus esqueletos. Ello también permitiría obtener la reconstrucción tridimensional aproximada de los puntos del esqueleto aplicando los mismos procesos presentados aquí. Otro trabajo futuro podría ser, entonces, obtener la rápida reconstrucción tridimensional de puntos de los esqueletos o de la combinación de bordes y esqueletos.

# Referencias


Abulafia, (2020). “Python – cuál es la complejidad de mi algoritmo”. Stack Overflow en Español. [En línea]: https: // es.stackoverflow.com / questions / 258283 / cu%C3%A1l-es-la-complejidad-algor%C3%Adtmica-de-este-programa. Consultado el 25/02/2021.

Bergamini María Lorena, Kamlofsky Jorge. “Aproximación Poligonal de Curvas Digitales”. Matemática Aplicada Computacional e Industrial Vol. 6 (2017): 545-548.

Bergamini, María L. & Jorge A. Kamlofsky. “Representación de Formas Digitales para Reconocimiento y Clasificación de Objetos.” Revista Colombiana de Computación 16 (2015): 28-47 .

Bergamini M., Ansaldo F., Bright G., & Zelasco J. Fundamental Matrix: Digital Camera calibration and Essential Matrix parameters. International Journal of Signal Processing, 1 (2016)(a): 120-126.

Bergamini M, Ansaldo F, Bright G & Zelasco J. “Fundamental Matrix: Digital camera calibration and Essential Matrix parameters”. 16th International Conference on Signal Processing, Computational Geometry and Artificial Vision 2016 proceedings (ISCGAV), (2016)(b).

Burgard et al.. "Experiences with an interactive museum tour-guide robot." *Artificial intelligence* 114.1-2 (1999): 3-55.

Coombs, Joseph, & R. Prabhu. "OpenCV on TI’s DSP+ ARM platforms: Mitigating the challenges of porting OpenCV to embedded platforms." *Texas Instruments* (2011).

Davis, Larry S. "A survey of edge detection techniques."*Computer graphics and image processing* 4.3 (1975): 248-270.

De Berg M et al.. “Computational Geometry”, *ISBN 3-540-61270-X Springer-Verlag Berlin Heidelberg New York* (1997).

Donadio, A., Mendez, D. “Vision 3D. Tratamiento de imágenes estereoscópicas. Restitución Automática. Evaluación de resultados. Universidad Nacional del Centro de la Provincia de Buenos Aires, Facultad de Ciencias Exactas, 1997

Eckhardt, Ulrich; Longin Latecki. “Digital topology”. *Inst. für Angewandte Mathematik* (1994).

Egmont-Petersen, Michael, Dick de Ridder, & Heinz Handels. "Image processing with neural networks—a review." *Pattern recognition* 35.10 (2002): 2279-2301.

García, Pedro Pablo García. "Reconocimiento de imágenes utilizando redes neuronales artificiales." *Facultad de Informatica*, *Universidad Complutense de Madrid* (2013).

Gil, Laurent and Liska, Allan. “Security with AI and Machine Learning.” *O’Reilly Media*, (2019).

Gonzalez, Rafael C.,Woods, Richard E.. "Digital image processing."*Addison-Wesley Publishing Company* (1993).

Greer D., McKerrow P., & Abrantes J., “Robots in Urban Search and Rescue Operations”, © ARAA, 2002 *Australasian Conference on Robotics and Automation*, November 2002.

Gross, Ari; Latecki, Longin. “Digital Geometric Invariance and Shape Representation”. *0-8186-7190-4/95*, *IEEE*, (1995): 121-126.

Guillermo, J. (2018) “¿Qué es la complejidad computacional y cómo se come?”. Sitio: Medium.com. [En línea]: https: // medium.com / @joseguillermo_ / qu%C3%A9-es-la-complejidad-algor%C3%Adtmica-y-con-qu%C3%A9-se-come-2638e7fd9e8c. Fecha de consulta: 24/2/2021

Hartley R & Zisserman A, “Multiple View Geometry in Computer Vision”, Cambridge University Press, 2003.

Heikkila J, “Geometric camera calibration using circular control points”, IEEE Transactions on Pattern Analysis and Machine Intelligence22(10), (2000): 1066-1077.

*Hernández, R., Fernández, C. y Baptista, P. "Metodología de la investigación (6ª ed.)". México: McGrwall Hill Education, (2014).*

Hernández Orallo, J; Ramírez Quintana, M, y Ferri Ramirez, C. "Introducción a la Minería de Datos". Editorial Pearson Educación SA, Madrid, (2004).

Hirschmüller, Heiko. "Accurate and efficient stereo processing by semi-global matching and mutual information.", IEEE, 2005.

Horn, Berthold KP. "Recovering baseline and orientation from essential matrix." *J. Opt. Soc. Am* 110 (1990).

Kalantary M. & Jung F, "Estimation Automatique de l'Orientation Relative en Imagerie Terrestre.", XYZ-AFT, 114, (2008): 27-31.

Kamlofsky, Jorge. "Topología digital, base para la visión artificial". Tesis de Licenciatura, Universidad Abierta Interamericana, Buenos Aires, 2011

Kamlofsky J., Bergamini, M.: Aproximación de Formas de Objetos Digitales por Polígonos. VII Jornadas Argentinas de Robótica ISBN: 978-950-658-316-3, (2012).

Kamlofsky, Jorge Alejandro & Bergamini, María Lorena. "Patrón de Evolución Discreta de Curvatura y Concavidad para Reconocimiento de Formas." *CONAIISI*, (2013)(b).

Kamlofsky, Jorge, & María Lorena Bergamini. "Patrón de Ángulo de Giro para Reconocimiento de Objetos en Imágenes Digitales." 14th Argentine Symposium on Technology, ISSN 1850-2806, (2013)(a): 115-127.

Kamlofsky, Jorge y Bergamini, María Lorena. "Herramientas de Análisis de Bordes para Reconocimiento y Clasificación de Objetos en Imágenes Digitales." II Congreso Nacional de Ingeniería en Sistemas de Información (Conaisi2014), 2014(a).

Kamlofsky, Jorge y Bergamini, María Lorena. "Un Modelo para Soporte de la Investigación Asistido por Técnicas de Visión Artificial." II Congreso Nacional de Ingeniería en Sistemas de Información (Conaisi2014), 2014(b).

Kamlofsky Jorge y Bergamini María Lorena. "Los Cuaterniones en Visión Robótica". Matemática Aplicada Computacional e Industrial Vol. 5 (2015): 517-520.

Kamlofsky. J y Bergamini, M. "Rápida Obtención de Puntos Homólogos para Vision 3D". *VI Congreso de Matemática Aplicada, Computacional e Industrial, Vol. 6 (2017)*: 549-552.

Kamlofsky J., Bergamini, M.: Aproximación de Formas de Objetos Digitales por Polígonos. VII Jornadas Argentinas de Robótica ISBN: 978-950-658-316-3, (2012).

Kamlofsky, J. Naidoo, N., Bright, G., Stopforth, R., Zelasco, J., Ansaldo, F., & Bergamini, M. "Semi-Autonomous Robot Control System with an improved 3D Vision Scheme for Search and Rescue Missions. A joint research collaboration between South Africa and Argentina." Advanced in Sciences, Technology and Engineering System Journal (Astes Journal) Vol **3** (6), 347-357 DOI: 10.25046/aj030643 (2018).

Kleiner K. "Better robots could help save disaster victims", 2006.

Kong, Yung; Kopperman, Ralph; Meyer, Paul. "A Topological Approach to Digital Topology". *The American Mathematical Monthly* Vol. 98, (1991): 901-917.

Lengyel, Eric. "Matemáticas para Videojuegos en 3D". Second Edition, Cengage Learning, (2011).

Longuet-Higgins H, "A Computer Algorithm for Reconstructing a Scene from Two Projections", Nature, 293 (10), (1981): pp 133-135.

Luong Q., Faugeras O., "Determining the Fundamental matrix with planes:Instability and new algorithms", *Proc. Conf. on Computer Vision and Pattern Recognition*, (1993): pp 489-494.

Luong Q. & Faugeras O.,"Self-Calibration of a moving camera from Point correspondences and fundamental matrices", *International Journal of Computer Vision*,22 (3), (1997): pp 261–289

Malpartida, Eddie Angel Sobrado. "Sistema de visión artificial para el reconocimiento y manipulación de objetos utilizando un brazo robot." *Pontificia universidad católica del Perú*, *Perú* (2003).

Mañas, José. “Análisis de algoritmos – Complejidad”. Universidad Politécnica de Madrid, Departamento de Ingeniería de Sistemas Telemáticos, (2017).

Mell P. and Grance Timothy. The NIST definition of cluod computing. “National Institute of Standards and Technology”. SP800-145, 2011

Monar, Monar, & Willan Leopoldo. *Aplicación de las Redes Neuronales al Reconocimiento de Objetos en Robots Manipuladores*. MS thesis. Quito, 2014.

Moravec, Hans P. "Robot spatial perceptionby stereoscopic vision and 3d evidence grids."*Perception* (1996).

Murphy R & Stover S. “Rescue Robots for Mudslides: A descriptive study of the 2005 La Conchita Mudslide Response”, International Journal of Field Robotics, Wiley, 2008.

Naidoo, N., Bright, G., Stopforth, R., Zelasco, J., Ansaldo, F., Bergamini, M., & Kamlofsky, J. (2017, November). “Semi-autonomous robot control system and with 3D vision scheme for search and rescue missions: A joint research collaboration between South Africa and Argentina.” In *Mechatronics and Machine Vision in Practice (M2VIP), 24th International Conference on* (pp. 1-6). IEEE, 2017.

Naidoo et al.. "Optimizing search and rescue missions through a cooperative mobile robot network." Pattern Recognition Association of South Africa and Robotics and Mechatronics International Conference (PRASA-RobMech), IEEE, 2015.

Nguyen T.P., Debled-Rennesson I.: Curvature Estimation in Noisy Curves. In: Kropatsch W., Kampel M., Hanbury A. (eds.) Computer Analysis of Images and Patterns, LNCS, vol. 4673, pp 474-481, Springer, Heidelberg (2007).

Open Source Computer Vision, (2018). OpenCV 4.0. [En Linea]: https: // docs.opencv.org / 4.0.0 / d1 / dfb / intro.html. Fecha de Consulta: 1/3/2019.

Pal N & Pal S. A review on image segmentation techniques, Pattern Recognition 26 (9) (1993) 1277–1294.

Peña-Cabrera, Mario, et al.. "Un Proceso de Aprendizaje para Reconocimiento de Objetos en Línea en Tareas Robotizadas." *3ª Conferencia Iberoamericana en Sistemas, Cibernética e Informática (CISCI 2004)*. 2004.

Rosenfeld, A. “Digital Topology”, *The American Mathematical Monthly*, 86(8), (1979) pp. 621-630.

Rosenfeld, A, & Kak, A. "Digital picture processing". Vol. 1. Elsevier, (2014).

Rowley, Henry A., Shumeet Baluja, & Takeo Kanade. "Neural network-based face detection."*IEEE Transactions on pattern analysis and machine intelligence* 20.1 (1998): 23-38.

Rudin, Leonid I., Stanley Osher, & Emad Fatemi. "Nonlinear total variation based noise removal algorithms."*Physica D: Nonlinear Phenomena* 60.1 (1992): 259-268.

Rumelhart D, Hinton G, Williams R. Learning internal representations by error propagation, in: D.E. Rumelhart, J.L. McClelland (Eds.), Parallel Distributed Processing: Explorations in the microstructure of Cognition, Vol. I, MIT Press, Cambridge, (1986) 319–362.

Sarría, Alfonso Fernández. Estudio de técnicas basadas en la transformada Wavelet y optimización de sus parámetros para la clasificación por texturas de imágenes digitales. Diss. Universitat Politècnica de València, (2007).

Schaeffer, E., (2020). “Complejidad computacional de problemas y el análisis y diseño de algoritmos”. [En línea]: *https://elisa.dyndns-web.com/teaching/aa/pdf/aa.pdf*. Consultado el 25/02/2021

Shannon, Claude Elwood. "A mathematical theory of communication." ACM SIGMOBILE, (1948).

Mobile Computing and Communications Review 5.1, pp. 3-55.Stewenius H, Engels C & Nister D. “Recent Developments on Direct Relative Orientation”, ISPRS Journal of Photogrammetry and Remote Sensing 60, (2006) 284-294.

Velogig. Machine Learning: Algoritmos más utilizados en marketing digital. En línea: *https://velogig.com/10-algoritmos-mas-usados-en-machine-learning-que-debes-conocer/* Consultado el 03/08/2020. “*Velogig*, *Marketing Científico*”, (2020)

Weaver, John B., et al.. "Filtering noise from images with wavelet transforms."*Magnetic Resonance in Medicine* 21.2 (1991): 288-295.

Zelasco, Jose F., et al.. "Computer vision in AUVs: automatic roto-rectification of stereo images."*OCEANS 2000 MTS/IEEE Conference and Exhibition*. Vol. 3. IEEE, (2000).

Zhang Z. “A flexible new technique for camera calibration”,IEEE Transactions on Pattern Analysis and Machine Intelligence22(11), (2000): 1330-1334.

Zhang Z. “Camera calibration with one-dimensional objects”IEEE Transactions on Pattern Analysis and Machine Intelligence26(7), (2004): 892-899.

Zhang D., Lu G. “Review of shape representation and description techniques”. *Patt. Recogn.*, *37* (1), (2004): 1-19.

Zhang, Q., Yang, L. T., Chen, Z., & Li, P. “A survey on deep learning for big data”. *Information Fusion*, *42*, (2018): 146-157.