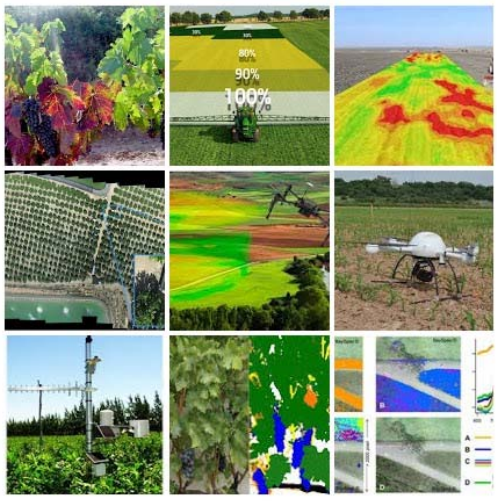
COLEGIO DE POSTGRADUADOS

Programa de Estudios

VISIÓN ARTIFICIAL APLICADA A LOS RECURSOS NATURALES: AGUA, SUELO Y VEGETACIÓN.

Manual de prácticas



JUNIO DE 2020

DRA. ANTONIA MACEDO-CRUZ
Postgrado en Hidrociencias

Contenido

PRESENTACIÓN

TITULO DEL CURSO:	Visión artificial aplicada a los recursos naturales: Agua, Suelo y Vegetación.
Tipo de curso:	Teórico-Práctico
PROFESOR TITULAR:	Antonia Macedo Cruz
CLAVE DE PROF.:	X03692
CORREO ELECTRÓNICO:	macedoan@colpos.mx

PRE-REQUISITOS

Los interesados en cursan esta asignatura deben tener conocimientos de programación, ya que para aprender sobre tratamiento matemático de imágenes digitales y poderlo aplicar a su vida profesional es muy importante realizar las prácticas correspondientes incluidas en el presente curso. Por ello el temario está diseñado para comenzar desde los fundamentos del tratamiento de imágenes e ir desarrollando los contenidos partiendo de lo aprendido en anteriores sesiones hasta llegar al reconocimiento de patrones (Visión Artificial de nivel medio). Para resolver las prácticas del presente curso se recomienda trabajar en MATLAB, lo cual le garantiza obtener asesoría por parte del profesor titular del curso sobre el uso de comandos del mismo. Esto no significa que no puedan usar otras herramientas (lenguajes de programación), sin embargo, no estará garantizada la asesoría en estos casos.

INTRODUCCIÓN

El procesamiento digital de imágenes es un campo de investigación abierto. El constante progreso en esta área no ha sido por sí mismo, sino en conjunto con otras áreas con las cuales está relacionada como la computación, las matemáticas y el conocimiento cada vez mayor de ciertos órganos del cuerpo humano que intervienen en la percepción y en la manipulación de las imágenes. Aunado a esto, la inquietud del hombre por imitar y usar ciertas características del ser humano como apoyo en la solución de problemas. El avance del Procesamiento Digital de Imágenes se ve reflejado en la medicina, la astronomía, geología, microscopía, la teledetección, Información meteorológica, entre otras. La visión artificial permite la detección automática de estructuras y propiedades de un posible mundo dinámico en 3 dimensiones a partir de una o varias

imágenes bidimensionales. En el presente curso la visión artificial nos permitirá, simular, evaluar e implementar sistemas de procesamiento de imágenes digitales cuyas prácticas se enfocan al reconocimiento, descripción y cuantificación de los recursos naturales: Agua, Suelo y Vegetación.

La importancia de implementarlo en este sector se debe a que como institución de enseñanza e investigación en ciencias agrícolas, se debe estar a la vanguardia en la implementación de nuevas tecnologías como la visión artificial. Es cierto que su mayor aplicación se desarrolla en la robótica, la industria y en la medicina; pero tiene un alto potencial para aplicarlo al reconocimiento de patrones en los recursos naturales, en agua suelo y vegetación.

La visión artificial a través de imágenes digitales es una de las tecnologías innovadoras con capacidad para reconocimiento de surcos, canales y cárcavas; el reconocimiento especies vegetales por su forma tamaño y color de las hojas; en el sector agrícola en cultivos de invernadero, reconocimiento de frutos, evaluación de madurez, para determinar fechas de corte; reconocimiento de la salud de la planta, evaluar la nutrición, la evaluación de plagas y enfermedades para detección temprana, o incluso tomadas desde sensores remotos (drones, satélites, etc.) para cubrir grandes superficies agrícolas.; clasificación de suelo por su color, textura, granulometría, etc.

El reconocimiento de patrones, no se limita a imágenes digitales del espectro visible, ya que los modelos matemáticos permiten clasificar y reconocer los objetos por sus descriptores matemáticos, pudiéndose aplicar cualquier banda espectral, desde imágenes binarias, en escala de gris, en color, imágenes multiespectrales, imágenes de radar, e incluso imágenes hiperespectrales. El estudio del comportamiento espectral se basa en métodos estadísticos y modelos matemáticos para los procesos de extracción de información con base a la percepción visual de los seres vivos y el análisis de imagen.

El objetivo general del presente curso es: al finalizar el curso el participante será capaz de contrastar las técnicas computacionales más eficientes para el procesamiento digital de imágenes y visión artificial de nivel medio; diseñar los algoritmos que le permitirán modelar matemáticamente los procesos de extracción de información con base a la percepción visual de los seres vivos y generar programas (software) para la simulación de las capacidades visuales por computadora (el reconocimiento). Mediante la visión artificial el participante será capaz de detectar de manera automática las estructuras y propiedades de un posible mundo dinámico en 3 dimensiones a partir de una o varias imágenes bidimensionales, y en particular en el presente curso se diseñarán, simularán, evaluarán e implementar sistemas de procesamiento de imágenes digitales

y lo aplicarán al reconocimiento, descripción y cuantificación de los recursos naturales: Agua, Suelo y Vegetación.

El presente curso tiene la característica de ser teórico práctico ya que para aprender sobre tratamiento matemático de imágenes digitales y poderlo aplicar a su vida profesional es muy importante realizar las prácticas, en las cuales mediante programación se genera el banco de modelos que dependiendo del enfoque usado para el reconocimiento este puede variar desde modelos creados para reconocimiento mediante descriptores superficial del objeto hasta modelos de reconocimiento de rasgos abstractos.

Los contenidos de las prácticas se dividen en capítulos, dispuestos de forma que sigan el proceso general de tratamiento de una imagen. Es decir, en primera instancia la imagen se coloca en la posición que facilite su tratamiento **(**giros, traslaciones, escalado**) denominado etapa de acondicionado o preprocesado;** el objetivo de esta etapa es permitir que etapas posteriores del análisis tengan mejores posibilidades de éxito**.** A continuación se debe tratar el **ruido si lo hubiera (filtrado).** Una vez que la calidad de la información es buena y tenemos el objeto de interés en una orientación adecuada se procede a la **extracción de características,** el extractor de características aplica operadores sobre una imagen segmentada o no segmentada, permitiendo identificar posiciones de rasgos que ayudan en la formación de hipótesis sobre la presencia de un objeto dado en la escena correspondiente**.** Posteriormente cuando ya se analizan dichas características y se extraen las principales que permiten distinguir diferencias entre cada objeto y reconocerlos por sus características principales, es decir se generan hipótesis para asignar certidumbres a los objetos presentes en la escena correspondiente, aplicando los modelos de clasificación o reconocimiento de objetos. Finalmente se evalúa la precisión del clasificador para cada objeto reconocido de la escena.

PLANTEAMIENTO GENERAL DE LAS SESIONES PRÁCTICAS

En las sesiones prácticas, a diferencia de las sesiones de teoría, los alumnos disponen de ordenadores y es en ellas donde se realizarán las prácticas utilizando como lenguaje de programación Matlab. Una vez que los temas se han tratado en las clases teóricas, la sesión de práctica se desarrollará bajo la siguiente metodología:

1) El profesor entrega al estudiante el material necesario para la práctica (Archivos digitales de entrada) y la propuesta metodológica, para que el participante dedique tiempo al trabajo de una manera autónoma para resolver los ejercicios prácticos planteados (este preferentemente debe ser en el laboratorio de cómputo en el cual se encuentra instalado el Software).

2) La dinámica de las sesiones es esencialmente práctica, los ejercicios están orientados para que los alumnos entrenen los métodos y conceptos explicados en la sesión. Se han planteado de este modo para facilitar el aprendizaje, de este modo a los alumnos se les explicará un método, su definición, para qué sirve, cual es la función del método y al final de la clase implementarán el método para resolver la práctica.
3) La distribución del tiempo no será igual a lo largo del curso, en las primeras sesiones el tiempo dedicado a trabajo autónomo será menor que al final, ya que las prácticas finales tienen mayor dificultad y engloban todos los conocimientos adquiridos en sesiones anteriores. Por tanto la planificación tanto del temario por sesión como del tiempo dedicado será gestionado por el profesor de la asignatura.

De manera general los contenidos que componen el temario de las sesiones de laboratorio se expone a continuación.

P1: Transformaciones Geométricas.

P2: Filtrado de Imágenes Digitales.

P3: Morfología.

P4: Extracción de Características.

P5: Diferentes técnicas de Segmentación.

P6: Procesamiento de Imágenes de Color

P7: Reconocimiento /Clasificación y verificación de resultados

PRÁCTICAS EN LABORATORIO DE CÓMPUTO

En este apartado se presentarán las prácticas de laboratorio, que en general se dividen en:

a) Ejercicios de prácticas a realizar por los alumnos después de cada sesión teórica.
b) Proyecto de fin de curso para resolver de manera individual (extra clase).

Para cada tema general se presentan los objetivos que se pretenden que los alumnos adquieran con ellos.

Los ejercicios de aprendizaje están diseñados para poner en práctica los conceptos aprendidos en la teoría de las sesiones, incrementando de forma gradual la complejidad. Se desarrollarán en la

misma sesión, después de la exposición del profesor o en la sesión siguiente cuando sea necesario, por lo tanto, las clases se deben impartir en el laboratorio de cómputo.

El proyecto de fin de curso puede ser seleccionado por el estudiante de acuerdo a su trabajo de investigación de tesis, o el participante puede seleccionar uno de varios temas propuestos por el profesor.

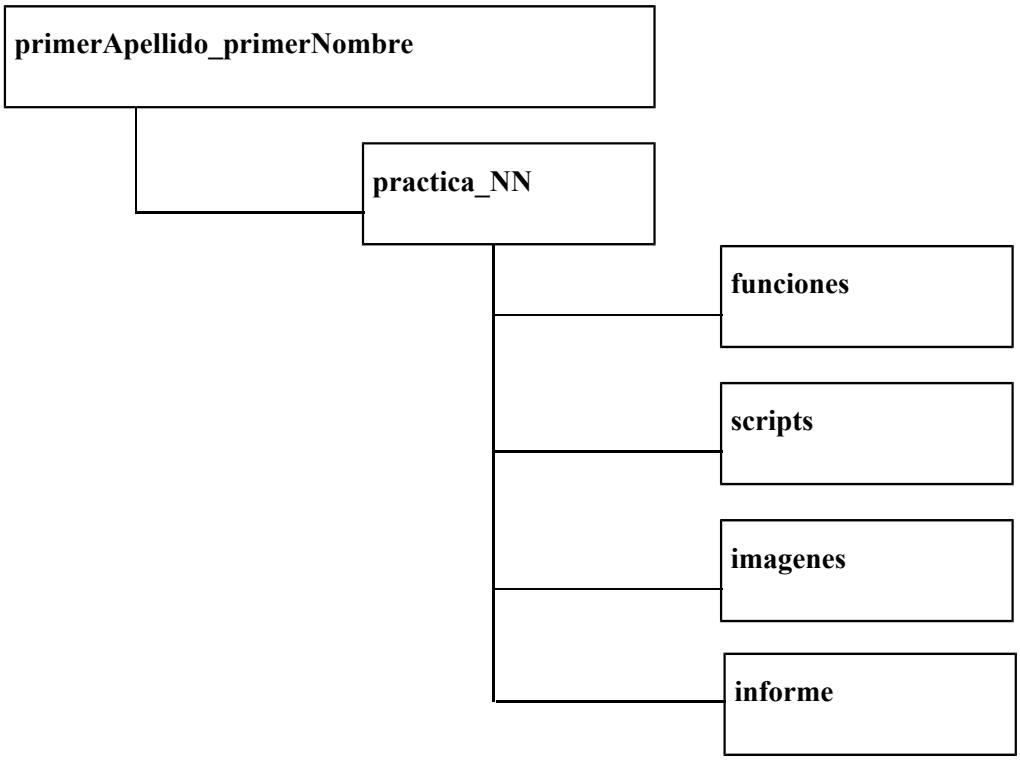
El documento de prácticas contiene algunas imágenes diseñadas por el autor y otras que han sido obtenidas de internet cuyo acceso es libre.

CRITERIOS PARA LA ENTREGA DE EJERCICIOS

- El trabajo es individual

Para cada práctica se entregará:

- Informe impreso con los resultados y conclusiones de la práctica.
- Archivo comprimido con el informe, los fuentes y las instrucciones para generar los resultados del informe. El archivo comprimido será en formato ".zip" y tendrá la siguiente estructura:



- En el directorio **funciones** se incluirán aquellas funciones pedidas en la práctica.
- En el directorio **scripts** se incluirán los programas principales que usan las funciones creadas y generan los resultados, gráficas, etc.. En este directorio se incluirá también un archivo de texto con las instrucciones para ejecutar los programas.
- En el directorio **scripts** se incluirá a modo de resumen un script (de nombre **main.m**) que muestre los resultados de la práctica.
- En el directorio **imagenes** se colocarán **solamente** las imágenes que se usen en la práctica y que no pertenezcan a la base de test (se entiende por base de test el conjunto de imágenes suministradas para la práctica del curso).
- En el directorio **imagenes** no se incluirán las imágenes resultado de la práctica. Éstas aparecerán en el informe y deberán poder generarse corriendo el script **main.m**.
- El acceso entre archivos de los directorios funciones, scripts, imagenes y base_test se hará mediante caminos relativos o suponiendo que los directorios pertenecen al path de Matlab.
- Se trabajará en Matlab (preferentemente). Salvo aclaración específica las funciones pedidas en las prácticas se programarán sin recurrir a bibliotecas existentes.

EJERCICIOS PRÁCTICOS

En este apartado se exponen los ejercicios correspondientes a las sesiones de laboratorio que le permitirán, por un lado, reafirmar el conjunto de conceptos vistos en la sesión a la que corresponden o en anteriores. Por el otro lado, le permitirán aprender nuevos conceptos y comenzar a poner en práctica sus habilidades de programación al manejar imágenes.

Ejercicios P1 Transformaciones Geométricas.

Objetivo de la práctica uno:

Al finalizar el tema el participante será capaz de programar y aplicar las estructuras de datos de las imágenes digitales, así como algunas propiedades de textura y compresión de imágenes.

Ejercicio 1.1 Dada la imagen digital binaria mostrada en la figura ***1.01***, diseñar un programa en algún lenguaje de preferencia que permita:

1) Barrer dicha imagen y contabilizar el número de píxeles con valor ***1***. Note que estos píxeles corresponden a la región conectada en la imagen.

Nombre función: *busca1.m*

Entradas:

- *I:* Imagen (matriz binaria)
- *Val:* Valor buscado

Salidas: *Cuantos1:* Número de píxeles con valor ***1***

2) Al usuario seleccionar un píxel dado, con valor ***1*** y determinar el número de píxeles con valor 1 dentro de su vecindad ***8***.

Nombre función: *vecindad8.m*

Entradas:

- *I:* Imagen (matriz binaria).
- *fila:* número de fina en que se encuentra el pixel seleccionado.
- *columna:* número de columna en que se encuentra el pixel seleccionado

Salida: *Cuantos1:* Número de píxeles con valor ***1*** dentro de la vecindad 8 del pixel.

3) Barrer dicha imagen y marcar los píxeles del contorno (píxeles externos) de la región compuesta mostrada en la imagen.

Nombre función: *contorno.m*

Entradas:

- *I:* Imagen (matriz binaria).

Salida:

- *fila:* número de fina en que se encuentra el pixel de contorno.
- *columna:* número de columna en que se encuentra el pixel de contorno.
- *Imagen2:* imagen de salida donde los pixeles de contorno se les asigna el valor 3.

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	0	0	1	1	0	0
0	0	1	1	0	0	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	1	0	0
0	0	0	0	0	0	0	0	0	0

Figura 1.01. Imagen muestra para resolver el ejercicio 1.1

Ejercicio 1.2. Dadas las imágenes digitales binarias de la figura ***1.02***, diseñar un programa en algún lenguaje de preferencia que permita:

1) Barrer dichas imágenes y decir si estas contienen solo regiones simplemente conectadas o solo regiones múltiplemente conectadas o una combinación de ambas.

   Nombre función: *tipoReg.m*

   Entradas:

   - *I:* Imagen (matriz binaria).

   Salida: *R*: es igual a 2 si contienen solo regiones simplemente conectadas.

   R es igual a 3 si contienen solo regiones múltiplemente conectadas.

   R es igual a 4 si contienen regiones múltiplemente conectadas y simplemente conectadas (una combinación de ambas).

2) Barrer dichas imágenes y marcar con un "2" los pixeles de las regiones simplemente conectadas y con un "3" las regiones múltiplemente conectadas.

Nombre función: *sustituir.m*

Entradas: *I:* Imagen (matriz binaria).

Salida: *Is:* imagen de salida regionalizada

3) Barrer dichas imágenes y para cada región múltiplemente conectada, contabilizar el número de contornos internos.

Nombre función: *contornosI.m*

Entradas: *I:* Imagen (matriz binaria).

Salida: *Ncont: numero de contornos internos*

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	0	0	0	1	1	0	0
0	0	0	0	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	0
0	0	1	0	1	1	1	1	1	0
0	1	1	0	1	1	1	1	1	0
0	1	1	0	0	1	1	1	1	0
0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	0	1	1	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	0	0	1	0
0	1	1	1	0	0	0	1	1	0
0	0	0	0	1	1	1	1	1	0
0	0	0	0	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0

Figura 1.02. Imágenes de muestra para resolver el ejercicio 1.2

Ejercicio 1.3. Dadas las imágenes digitales de la figura ***1.03***, diseñar un programa en algún lenguaje de preferencia que permita barrer dicha imagen, evaluar y mostrar el histograma correspondiente a cada imagen. En ambos, suponga que ***L=8***

Nombre función: histo.*m*

Entradas:

- *I:* Imagen (matriz binaria).

Salida:

tb = tabla de valores (clase, frecuencia)

his: histograma (formato imagen)

0	0	0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	1	1	0
0	1	2	2	2	2	2	2	1	0
0	1	2	3	3	3	3	3	1	0
0	1	2	3	6	5	4	3	1	0
0	1	2	3	6	5	4	3	1	0
0	1	2	3	3	3	3	3	1	0
0	1	2	2	2	2	2	2	1	0
0	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0	0
0	0	3	4	3	3	4	3	4	0
0	1	3	3	4	5	4	4	3	0
0	1	4	4	4	5	5	3	4	0
0	0	0	0	3	6	5	0	0	0
0	0	1	0	5	6	4	1	0	0
0	0	1	0	5	6	3	1	0	0
0	1	0	1	5	5	4	1	0	0
0	0	0	0	4	3	3	0	0	0
0	0	0	0	0	0	0	0	0	0

Figura 1.03. Imágenes muestra para resolver el ejercicio 1.3

Ejercicio 1.4. El número de pixeles con valor ***1*** en una de las filas (columnas) de una imagen binaria puede ser calculado al contabilizar el número de **1**s a lo largo de esa fila y a lo largo de esa columna. Así por ejemplo, el número de pixeles con valor ***1*** en la tercera fila de la imagen de la izquierda de la figura ***1.02*** es de cuatro, mientras que el número de pixeles con valor ***1*** en la cuarta columna de la misma imagen es de cinco. Dada la imagen digital binaria de la figura ***1.04***, diseñar un programa en algún lenguaje de preferencia que permita barrer dicha imagen y contabilizar el número de pixeles con valor ***1*** en cada una de las filas ***y*** columnas de la imagen. Para una imagen dada, el programa debe mostrar estos números acomodados como dos arreglos (uno para filas y otro para las columnas). Las dimensiones de estos arreglos son iguales, respectivamente a los números de filas (columnas***)*** de la imagen. Para el caso de la imagen de la figura ***1.04***, las dimensiones respectivas de los dos arreglos serán de ***10*** elementos.

A los arreglos lineales de esta forma se les conoce en la literatura, respectivamente como histograma de acumulación horizontal e histograma de acumulación vertical.

Nombre función: *HisAcum.m*

Entradas:

- *I:* Imagen (matriz binaria).

Salida: *taha:* tabla de acumulación horizonatal y vertical (10 x 2)

0	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0
0	1	1	1	1	1	0	0	0	0
0	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0	0
0	0	0	0	1	1	1	0	0	0
0	0	0	0	0	0	0	0	0	0

Figura 1.04. Imágenes muestra para resolver el ejercicio 1.4

Ejercicio 1.5. Dada la imagen digital X

1) Programar una función que realice la rotación de una imagen respecto del origen.

   En la interpolación se utilizará el método del "vecino más próximo".

   Nombre función: *rotar.m*

   Entradas:

   - *I: Imagen en escala de gris.*

   -*angulo:* ángulo que se desea rotar en grados.

   Salidas:-*R:* Imagen rotada

2) Ensayar el programa realizando una rotación de 15º (o el que se escriba) sobre las imágenes x (que se adjuntan)

Ejercicio 1.6. Dada una imagen Y

1) Programar una función que realice una traslación.

   Nombre función: *trasl.m.*

   Entradas:

   -*I:* Imagen a trasladar.

   -*distx*: píxeles de traslación en la dirección del eje x.

   -*disty:* píxeles de traslación en la dirección del eje y

   Salidas:

-*T:* Imagen trasladada.

2) Ensayar el programa realizado anteriormente sobre la imágen proporcionada desplazándola 100 pixeles a la izquierda sobre el eje X (x inicial -100).
3) Ensayar el programa realizado anteriormente sobre la imagen propuesta y desplazándola 100 pixeles arriba (y inicial -100) y 150 pixeles a la izquierda (x inicial -150).

Ejercicios P2: Filtrado de Imágenes Digitales

Objetivo de la práctica dos:

Al finalizar el tema el participante será capaz de comparar, diseñar, programar e implementar distintos modelos de filtro, para el suavizado de la imagen digital, la eliminación de ruido, realzar la imagen y detectar bordes.

Ejercicio 2.1. Dada la imagen X proporcionadas con la práctica, introducir los diferentes tipos de ruido que se piden a continuación, analizar su efecto sobre una imagen.

Guardar la imagen en el directorio de trabajo correspondiente a la práctica.

1) Ruido gaussiano. (media: 0, varianza: 0.01).

   Mostrar la imagen original y con ruido.

2) Ruido de Poisson.

   Mostrar la imagen original y con ruido.

   Guardar la imagen en el directorio de trabajo como 'florespoisson.jpg'.

3) Ruido sal y pimienta. (densidad: 0.5).

   Mostrar la imagen original y con ruido.

   Guardar la imagen en el directorio de trabajo carpeta de salida.

4) Ruido multiplicativo. (varianza: 0.01).

   Mostrar la imagen original y con ruido.

   Funciones de Matlab sugeridas:

   → 'imnoise'

Ejercicio 2.2. Sobre cada una de las imágenes proporcionadas para la práctica, ensayar los filtros espaciales lineales siguientes y mostrar por pantalla los resultados:

Analizar cómo los distintos filtros reducen el ruido y cuáles filtros tienen una mejor respuesta.

Filtro de la media. (Tamaño de la máscara 3x3) Filtro de Gaussiano.
(Tamaño de la máscara 3x3)

Funciones de Matlab sugeridas:

→ 'imfilter'

→ 'fspecial'

Ejercicio 2.3.Dadas las imágenes digitales, aplicar los filtros espaciales no lineales siguientes y mostrar por pantalla los resultados. Analizar los resultados de diferentes filtros sobre una imagen afectada con ruido:

Filtro de la mediana. (Tamaño de la máscara 3x3)

Filtro del máximo. (Tamaño de la máscara 3x3)

Filtro del mínimo. (Tamaño de la máscara 3x3)

Funciones de Matlab sugeridas:

→ 'imfilter'

→ 'medfilt2'

→ 'nfilter'

Ejercicio 2.4. Dadas las imágenes digitales (tres por lo menos), aplicar sobre las tres imágenes un filtro de la media y de la mediana y mostrar los resultados por pantalla.

Utilizar máscaras de dimensión 3x3.

Funciones de Matlab sugeridas:

→ 'imfilter'

→ 'medfilt2'

→'fspecial'

Ejercicio 2.5. Dada la imagen X,

1) Aplicar sobre la imagen un filtrado de la primera derivada. Las máscaras a emplear son las siguientes:

$\frac{\partial f(x,y)}{\partial x}$ | -1 | 1 |

$\frac{\partial f(x,y)}{\partial y}$ | -1 |
| 1 |

Mostrar los resultados por pantalla.

2) Realiza la suma de módulos de las dos imágenes de gradiente y comparar los resultados con la imagen original.

Funciones de Matlab sugeridas:

→ 'filter2'

Ejercicio 2.6 Dada la imagen X

1) Sobre la imagen X realizar un filtrado de Sobel utilizando las siguientes máscaras:

$\left[\frac{\partial f}{\partial x}\right]$

-1	-2	-1
0	0	0
1	2	1

$\left[\frac{\partial f}{\partial y}\right]$

-1	0	1
-2	0	2
-1	0	1

Realizar la suma de módulos las dos direcciones y mostrar por pantalla los resultados.

2) Ver las opciones de la función 'edge' en la ayuda de matlab.
3) Sobre la misma imagen aplicar el filtro de Sobel mediante la función 'edge'.

Ensayar las diferentes opciones que permite la función, ('thinning', direction).

Funciones de Matlab sugeridas:

→ 'edge'

→ 'filter2'

Ejercicio 2.7. Con las imágenes digitales proporcionadas para la práctica aplicar los filtros extractores de contorno que se piden a continuación.

1) Sobre las distintas imágenes, aplicar los filtros de Sobel, Roberts y Prewitt. Mostrar con cada imagen los resultados y analizar las diferencias entre ellos sobre varias imágenes.

Funciones de Matlab sugeridas:

→ 'edge'

Ejercicio 2.8. Con el objetivo de analizar el comportamiento del filtro LoG ante dos tipos de ruido, y mediante las imágenes que se adjuntan con la práctica, realizar los siguientes ejercicios.

1) Extraer los contornos de la imagen mediante la Laplaciana gaussiana (LoG).
2) Introducir ruido tipo gaussiano (media: 0, varianza: 0.01) en la imagen y aplicar el método de la Laplaciana gaussiana de nuevo y observar los resultados.
3) Aplicar ahora un ruido tipo sal&pimienta (densidad: 0.01) y pasar el filtro LoG. Mostrar los resultados por pantalla.

Funciones de Matlab sugeridas:

→ 'edge'

Ejercicio 2.9. Utilizando la imagen correspondiente a la práctica y con el objetivo de encontrar las diferencias entre los filtros extractores de contornos antes varios tipos de ruido, realizar los ejercicios siguientes:

1) Sobre la imagen, extraer los contornos mediante los métodos: Sobel, Prewitt, Roberts, LoG y Canny. Mostrar los resultados.
2) Introducir en la imagen un ruido tipo sal&pimienta de densidad: 0.01. Ahora extraer los contornos utilizando el método de canny y LoG. Mostrar los resultados.
3) A la imagen con ruido aplicar: un filtro de Gauss 3x3 (sigma=0.5), un filtro de la media 3x3 y un filtro de la mediana 3x3. Mostrar los resultados.
4) Sobre la imagen filtrada mediante el filtro de la mediana y el filtro de la media, extraer los contornos mediante LoG y Canny. Mostrar los resultados.

Funciones de Matlab sugeridas:

→ 'edge'

→ 'imnoise'

→ 'fspecial'

→ 'imfilter'

Ejercicios P3: Morfología

Objetivo de la práctica tres:

Al finalizar el tema, el participante será capaz de analizar las características de interés de una imagen, seleccionar apropiadamente e implementar operaciones morfológicas de erosión, dilatación, cierre y apertura, en el contexto de un problema dado.

Ejercicio 3.1. Dadas las imágenes binarias (3 imagenes) que se adjuntan a la práctica**.** Programar una función que realice la inversión de una imagen binaria como se describe a continuación:

1) Crear una función que invierta los valores de una imagen binaria. Es decir, que los valores "0" (negros) pasen a valer "1" (blancos).

Nombre función: *invertir.m*

Entradas:

-*I*: Imagen binaria.

Salidas:

-*Ic*: Imagen inversa.

2) Usando la función sobre las imágenes que se adjuntan a la práctica y mostrar los resultados.

Ejercicio 3.2 Obtener el número de componentes conectadas mediante vecindad 4 y mediante vecindad 8 de la matriz que se muestra en la figura 3.01. Utilizar la función "bwconncomp".

0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0
0	1	1	1	0	1	1	0	0
0	1	1	1	0	1	1	0	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	1	0
0	0	1	1	0	0	1	1	0
0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	0

Figura 3.01. Imagen muestra para resolver el ejercicio 3.2

Puede usar la funcion de Matlab siguiente (si así lo prefiere):

→ 'bwconncomp'

Ejercicio 3.3. Dada la imagen que se adjunta a la práctica, desarrollar un programa que permita aplicar las operaciones morfológicas de erosión/dilatación mediante las funciones específicas de MatLab, como se pide a continuación:

1) Realizar una dilatación mediante un 'strel' de diamante con radio 2 y 5. Mostrar los resultados.
2) Realizar una erosión mediante un 'strel' de disco de tamaño 1. Mostrar los resultados.
3) Ensayar la dilatación mediante los siguientes 'strel':

- Línea, longitud 2 y ángulo 45º.
- Octágono, tamaño 3.
- Rectángulo, longitud 2 y altura 3.

- Arbitrario, de la siguiente forma:(en azul los 1, en blanco los 0, figura 3.02).

Mostrar los diferentes resultados por pantalla.

Figura 3.02

Funciones de Matlab sugeridas:

→ 'imerode'

→ 'imdilate'

Ejercicio 3.4. Dada una imagen en color (que presenta algún problema de rotación) se pide:

1) Binarizar la imagen y presentar la imagen de forma que se visualicen las figuras bien orientadas de forma vertical (rotar si es necesario).
2) Realizar una dilatación con los siguientes elementos estructuradores:

- Disco, radio 1.
- Diamante, radio 2.
- Línea, longitud 3 y altura 0.

Mostrar los resultados por pantalla.

Funciones de Matlab sugeridas:

→ 'imdilate'

Ejercicio 3.5. Dadas las imágenes correspondientes a la práctica:

1) Realizar una apertura sobre la imagen **x**. Utilizar un 'strel' de disco de tamaño 10. Finalmente aplicar una erosión con 'strel' de octagonal de tamaño 12. Mostrar los resultados.
2) Realizar un cierre sobre la imagen **y**. Con línea de 3 y 0º.

- Realizar una apertura. Con línea de 4 y 90º.

- Aplicar una erosión con rectángulo 6x6.

- Mostrar los resultados.

Funciones de Matlab sugeridas:

→ 'imopen'

→ 'imclose'

→ 'imerode'

Ejercicio 3.6. Dadas las imágenes correspondientes a la práctica, aplicar operaciones de erosión y dilatación pero sobre imágenes en escala de gris en vez de sobre imágenes binarias.

1) Realizar una dilatación en escala de gris sobre la imagen x. Utilizar un elemento estructurador de disco y radio 5. Mostrar los resultados.
2) Aplicar sobre la imagen en escala de gris una erosión utilizando un elemento estructurador de línea con longitud 3 y 45º. Mostrar los resultados.

Funciones de Matlab sugeridas:

→ 'imdilate'

→ 'imerode'

Ejercicio 3.7. Dadas las imágenes correspondientes a la práctica, aplicar una operación morfológica que se piden, sobre una imagen en escala de gris

1) Realizar una apertura en escala de gris sobre la imagen. Utilizar un elemento estructurador de octágono 6. Mostrar los resultados.
2) Aplicar sobre la imagen en escala de gris un cierre utilizando un elemento estructurador de diamante de tamaño 4. Mostrar los resultados.
3) Realiza una binarización de ambos resultados. Mostrar los resultados.

Funciones de Matlab sugeridas:

→ 'imopen'

→ 'imclose'

Ejercicio 3.8. Dadas la imagen correspondiente a la práctica

Realizar las siguientes operaciones (para mejorar el posterior tratamiento) y mostrar por pantalla el resultado de cada operación:

1) Obtener la imagen binaria.
2) Eliminar todos los conjuntos blancos de menos de 10 píxeles y conectividad 8.
3) Eliminar todos los conjuntos blancos de menos de 20 píxeles y conectividad 4.
4) Realizar una apertura mediante 'bwmorph'.
5) Eliminar todos los conjuntos blancos de menos de 15 píxeles y conectividad 8.

Funciones de Matlab sugeridas:

→ 'bwareaopen'

→ 'bwmorph'

Ejercicio 3.9. Dadas la imagen correspondiente a la práctica

1) Obtener el perímetro de la imagen mediante erosión.
2) Obtener el perímetro mediante la función 'bwmorph'.
3) Obtenerlo el perímetro mediante la función 'bwperim'.
4) Mostrar por pantalla los diferentes resultados (para su comparación).

Funciones de Matlab sugeridas:

→ 'imerode'

→ 'bwmorph'

→ 'bwperim'

Ejercicios P4: Extracción de Características

Objetivo de la práctica cuatro:

Al finalizar el tema el participante será capaz de comparar, diseñar, simular e implementar los principales descriptores que caracterizan a un objeto, tal como: área, circularidad, rectangularidad,

alargamiento, centro de gravedad y momentos geométricos, los ejes principales de inercia que determinan la orientación, así como los descriptores de contornos.

Ejercicio 4.1 Dada una imagen binaria, programar una función que cuente los píxeles totales de un objeto. Se considerará como objeto aquellos píxeles que estén en blanco, y calcular el área del objeto.

Nombre función: calarea.m

Entradas: -I: Imagen de un objeto.

Salidas: -area: tamaño en píxeles del objeto.

Ejercicio 4.2. Dadas las 4 imágenes digitales cuyo contenido incluye un nombre escrito en cada una. Se pide:

1) Realizar una función que identifique el nombre escrito en la imagen. La función deberá devolver por pantalla el nombre escrito en la imagen.

Nombre función: nombre.m

Entradas: -I: Imagen con nombre escrito.

Salidas: -nombre: el nombre escrito en la imagen, muestra por pantalla el resultado.

Ejemplo de ejecución:

```
>>I=imread('texto2.jpg');
>>nombre(I);
El nombre escrito en la imagen es: Álvaro
```

Ejercicio 4.3. Realizar una función que extraiga las coordenadas del centro de gravedad de un objeto (utilizar el modelo matemático estudiado en la misma sesión). Consideraremos que el objeto

está representado mediante píxeles blancos. Correr el programa aplicándolo a las imágenes proporcionadas con la práctica.

Nombre función: centrogravedad.m

Entradas: I: Imagen binaria.

Salidas: [x, y]: vector con las coordenadas del CG.

Recordad que las coordenadas en Matlab están definidas de tal forma que la coordenada 0 para [***x,y***] se ubica en la esquina superior izquierda, como se muestra en la figura 4.03:

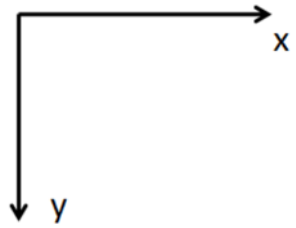


Figura 4.01. *Punto inicial de coordenadas de una imagen*

Ejercicio 4.4 Realizar una función que calcule el ángulo de los ejes principales de inercia. Y que además nos devuelva a su vez las coordenadas del centro de gravedad. Tomaremos como pieza los píxeles blancos. Se proporcionan dos imágenes para probar la ejecución del programa.

Nombre función: ejesinercia.m

Entradas: -I: Imagen binaria.

Salidas: -[angle, x, y]: vector con el ángulo de los ejes principales de inercia y las coordenadas del CG.

Ejercicio 4.5.

Descripción

Se cuenta con un invernadero tecnificado, en el cual se pretende realizar una plantación de alcatraces. El objetivo es colocar en el centro de cada maceta un esqueje de alcatraz mediante el brazo robot. Las macetas se colocan en una banda y en un extremo de la banda se dispone de un brazo robot, quien tomará cada esqueje y lo colocará en el centro de cada maceta.

*El brazo robótico necesita las coordenadas del CG de cada maceta para poder depositar el esqueje de alcatraz en el centro de la maceta.

*Tomamos imágenes de tramos de la banda transportadora en las que las macetas siempre aparecen completas y en posición "vertical".

Se pide:

1) Realizar un programa que calcule las coordenadas del centroide de cada una de las macetas de cada toma y que contabilice el número de alcatraces plantados. Ejecutar el programa con las imágenes proporcionadas.

Ejemplo de ejecución:

```
>>I=imread('maceta1.jpg');
>>procesatoma(I);
Coordenadas de los centroides:
T1(fila 103,columna 243)
T2(fila 403, columna 785)
T3(fila 818, columna 472)
T4(fila 984, columna 637)
Número de macetas: 4
```

Funciones de Matlab sugeridas:

→ 'bwlabel' → 'ejesinercia'

→ 'imdilate' → 'regionprops'

→ 'imerode' → 'bwareaopen'

Ejercicio 4.6 Para resolver la presente prácticas se deberán aplicar los métodos que se han proporcionado en temas anteriores hasta el presente. Finalmente ejecutar el programa con las imágenes proporcionadas con la práctica.

1) Realizar una función que identifique la ficha de dominó. Es decir, tendrá como entrada una imagen de una sola ficha y mostrará por pantalla el tipo de ficha que es. Las medidas de las fichas son: 50 x 110 píxeles.

Una posible solución es:

- Binarizar la imagen.
- Extraer el centro de gravedad de la ficha.
- Obtener la orientación y girar la ficha hasta una posición horizontal.
- Segmentar las dos zonas donde se encuentran los puntos que identifican la ficha.
- Una vez separados se realiza un etiquetado y se identifica la ficha.
- Por último se presentan los datos por pantalla.

Nombre función: fichadomino.m

Entradas: -I: Imagen RGB.

Salidas: -Muestra por pantalla la identificación de la ficha.

Ejemplo de ejecución:

```
>>I=imread('texto2.jpg');
>>fichadomino(I);
La ficha es: 3-2
```

Funciones de Matlab sugeridas:

→ 'bwlabel'

→ 'imrotate'

Ejercicio 4.7. Programar una función que realice la extracción del contorno de una imagen binaria mediante el método de la tortuga de Papert. El objeto está definido por píxeles blancos.

Nombre función: papert.m

Entradas: -I: Imagen binaria.

Salidas: -C: Imagen con el contorno.

-[f, c]: matriz con las coordenadas de los píxeles del contorno.

Ejercicio 4.8. Desarrollar un programa para **a**plicar la extracción del perímetro mediante el método de la tortuga y mediante la función 'bwperim'. Provar la ejecución del programa con las imágenes propuestas con la práctica y comparar los resultados con ambos métodos (el de la tortuga y bwperim').

Se sugiere extraer características de las imágenes y mostrar por pantalla todas las características:

- Longitud de la cadena del contorno.
- El área de la llave.
- Coordenadas del CG y ángulo de los ejes principales de inercia.
- El grado de circularidad.
- Número de agujeros.
- Número de Euler.
- La compacidad.

Funciones de Matlab sugeridas:

→ 'papert' → 'ejesinercia' → 'bwmoph'

→ 'bwperim' → 'regionprops' → 'bweuler'

→ 'calarea'

Ejercicio 4.9. Aplicación de Visión Artificial: En una fábrica de galletas circulan galletas terminadas por una cinta transportadora (figura 4.02) .

Una cámara toma imágenes grises de galletas para su inspección. Cada imagen contiene varias galletas. Nunca habrá una galleta que quede cortada por la imagen. Las galletas están completas. Las galletas defectuosas son aquellas que están rotas.

1. El objetivo de la práctica es realizar un programa que inspeccione de forma automática la correcta dimensión de las galletas. Tendremos como plantilla una imagen de una galleta correcta: "galletaok.jpg" y un conjunto de imágenes de galletas que frecuentemente toma la cámara (para la prueba del programa por lo menos dos imágenes de galletas).

   El programa contará el número de galletas defectuosas y calculará las coordenadas del centro de gravedad. Mostrará por pantalla el resultado de la inspección.

Se sugiere para resolverlo:

- Procesar la plantilla calculando su área. Mediante la comparación de áreas identificar las galletas defectuosas.
- Procesar la imagen con las galletas y etiquetar cada galleta.
- Con cada galleta obtener su área y si el área difiere un 5% tomarla como defectuosa.
- Obtener las coordenadas de los centros de gravedad de las galletas defectuosas.
- Por último presentar los resultados por pantalla.

Nombre función: inspeccion galletas.m

Entradas: -I: Imagen gris de una toma de la cinta transportadora.

Salidas: -Muestra por pantalla el número de galletas defectuosas y las coordenadas de los CG de las piezas defectuosas.

Ejemplo de ejecución:

```
>>I=imread('galleta1.jpg');
```

```
>>inspeccionGalletas(I);

Número de galletas defectuosas: 2

Coordenadas del centroide de las galletas defectuosas:
Defectuosa_1: fila 49, columna 120

Defectuosa_2: fila 175, columna 76
```

Funciones de Matlab sugeridas:

→ 'bwlabel' → 'centrogravedad' → 'bwmorph'

→ 'calarea' → 'bwareaopen'

Figura 4.02. Cinta transportadora de galletas

Ejercicios P5: Diferentes técnicas de segmentación

Objetivo de la práctica cinco:

Al finalizar el tema el participante será capaz de comparar, diseñar, programar e implementar diversas técnicas de segmentación, como la transformada de Hough para detectar cirulos y rectas, así como algunos detectores de esquinas y crecimiento de regiones.

Ejercicio 5.1. Encontrar mediante la transformada de Hough las líneas presentes en las imágenes. Dibujar las rectas encontradas mediante la función "plot". Ejecutar el programa y probarlo con las imágenes proporcionadas con la práctica.

Funciones de Matlab sugeridas:

→ 'hough'

→ 'houghpeaks'

→ 'houghlines'

→ 'plot'

Ejercicio 5.2. Con el objetivo de que los alumnos desarrollen sus capacidades para resolver una aplicación de visión artificial. En las posibles soluciones pueden emplear diferentes herramientas que conducirán a diferentes estrategias para llegar a la solución.

Se pide realizar una función que compruebe si dos llaves son iguales. Es decir, que la muesca de la cerradura en las dos llaves sea igual en las dos llaves. En la imagen solamente hay dos llaves. Las llaves tienen una orientación aleatoria.

Una posible solución podría ser:

- Pre procesar la imagen: convertir a gris, binarizar, extraer contornos y etiquetamos.
- Separamos cada llave en una imagen.
- Procesar cada llave calculando su centro de gravedad y su orientación. A continuación se trasladan al centro de la imagen y se rotan hasta una posición horizontal.
- Comparar las dos imágenes realizando una resta y se calcula el área de las diferencias obtenidas. Si el área de las diferencias supera el 1% del área total de la llave, las llaves son diferentes.
- Finalmente se muestran por pantalla los resultados.

Nombre función: llaves.m

Entradas: -I: Imagen RGB de dos llaves.

Salidas: -Informa por pantalla si las llaves son iguales.

Ejemplo de ejecución:

```
>>I=imread('parllaves1.jpg');
>>llaves(I)
Las llaves son iguales
```

Funciones de Matlab sugeridas:

→ 'bwlabel'

→ 'edge'

→ 'imfill'

→ 'calarea'

→ 'imrotate'

→ 'trasl'

Ejercicio 5.3. Con el objetivo de que los alumnos practiquen la función de hough para la detección de circunferencias., se pide: buscar todas las circunferencias de radio entre 50 y 100 píxeles presentes en la imagen adjunta (por ejemplo "pelotas.jpg").

Dibujar las circunferencias obtenidas sobre la imagen gris, dibujar también las coordenadas de los centros.

Sugerencia:

- Utilizando la función "**houghcircle.m**", programada como material de ayuda y suministrada por el profesor o crear su propia función y utilizar la función **hough** incluida en Matlab,
- Otras funciones utiles:

  → 'plot'

  → 'rectangle'

Ejercicio 5.4. Utilizando una imagen circulara como de un aro, corona, etc, se pide:

- Ajustar una circunferencia a una serie de puntos.
- Encontrar la circunferencia que mejor se ajusta a la imagen.
- Obtener las coordenadas del centro y el radio de la circunferencia.
- Dibujar el centro y la circunferencia sobre la imagen original.

Funciones de Matlab sugeridas

→ 'papert'

→ 'rectangle'

→ 'plot'

Entre otras

Ejercicio 5.5. Dada una imagen de un reloj de manecillas como los que se adjuntan con la práctica; realizar una función que determine la hora que marca el reloj; ejecutarlo para validarlo con las imágenes que se adjuntan a la práctica.

Nombre función: quehoraes.m

Entradas: -I: Imagen del reloj.

Salidas: - Devuelve por pantalla la hora que marca el reloj.

Ejemplo de ejecución:

```
>>I=imread('reloj2.jpg');
>>quehoraes(I)
Son las 11:12.
```

Funciones de Matlab sugeridas:

→ 'minimos'

→ 'bwlabel'

→ 'bwlabel'

→ 'houghcircles'

→ 'regionprops'

→ 'papert'

→ 'ejesinercia'

→ 'bwlabel'

→ 'centrogravedad'

Una posible solución para el ejercicio se resume en el siguiente esquema de operaciones:

- Pre procesar: convertir a gris, binarizar, extraer contornos.
- Calcular centro de gravedad que será el centro del reloj y eje de las agujas.
- Crear dos plantillas para eliminar una parte del reloj. La primera plantilla para eliminar el centro del reloj y la segunda para eliminar el exterior (los números romanos).
- Una vez eliminados sólo queda en la imagen las dos agujas separadas.
- Etiquetar las agujas e identificarlas mediante la diferencia de áreas.
- Se obtiene la orientación de cada aguja.
- Deducir la hora que indican a través del ángulo de orientación y el cuadrante que ocupan.
- Finalmente mostrar los resultados por pantalla.

Ejercicios P6: Procesamiento de Imágenes de Color

Objetivo de la práctica seis:

Al finalizar el tema el participante será capaz de categorizar los principales modelos de color de acuerdo a sus aplicaciones y utilidad.

Comparar, diseñar, simular e implementar los algoritmos de transformación de los modelos de color con base a las necesidades de segmentación.

Ejercicio 6.1. A partir del tetraedro de color asignar los valores correspondientes a cada pixel para obtener los siguientes colores:

1) Rojo
2) Verde
3) Azul
4) Amarillo
5) Cyan
6) Magenta
7) Gris
8) Blanco
9)

0.0	0.0	1.0

Ejercicio 6.2. Dada la figura 6.1, expresada en modelo de color RGB:

0.0	1.0	1.0
0.0	0.2	0.5

expresar su transformación al modelo CMY.

1.0	0.0	0.0
1.0	0.0	1.0
0.2	0.0	0.5

(a)

0.0	1.0	0.0
1.0	1.0	0.0
0.2	0.2	0.5

(b)

(c)

Figura 6.1. Tres bandas espectrales del modelo de color RGB donde (a) Banda R, (b) Banda G (c) banda B

Ejercicio 6.3. Dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB:

Realizar un programa en que permita convertir a los siguientes formatos:

De RGB a CIE L*a*b*

De RGB a HLS

De RGB a CMY

De CIE L*a*b* a RGB

De HSV a RGB

De CMY a RGB

Ejercicios P7: Reconocimiento /Clasificación y verificación de resultados

Objetivos de la práctica siete:

Al finalizar el tema el participante será capaz de comparar, diseñar, programar, implementar y validar los principales modelos de clasificación supervisada y no supervisada.

Ejercicio 7.1. Analice el artículo adjunto (avena2011.pdf) y dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB programar y analizar lo siguiente:

1. Convertir las imágenes de modelo de color RGB a L*a*b.

2. A cada banda espectral aplicar el método de segmentación mediante el algoritmo ISODATA (propuestos por Ridler y Calvard (1978)).
3. Aplicar el método de clasificación natural al combinar las tres bandas espectrales, y el etiquetando cada una de las clases (propuesto por Macedo *et al.* 2011).
4. Aplicar el modelo de clasificación y mezcla (propuesto en Macedo *et al*. 2011), para la clasificación de las imágenes.
5. Determinar las unidades de muestréo para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
6. Determinar el número de muestras para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
7. Definir un esquema de muestreo y aplicar el muestreo digital en las imágenes originales y en las clasificadas.
8. Calcular la matriz de error determinista y Fuzzy (Anderson et al. (1976), Congalton y Green, 2009, Macedo *et al*. 2011)
9. Determinar la precisión del clasificador para cada clase.

Escribir el reporte de prácticas señalado en los criterios para la entrega de ejercicios

Ejercicio 7.2. Analice el artículo adjunto (avena2011.pdf) y dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB programar y analizar lo siguiente:

1. Convertir las imágenes de modelo de color RGB a L*a*b.
2. A cada banda espectral aplicar el método de segmentación propuesto por Otsu (Otsu, 1979).
3. Aplicar el método de clasificación natural al combinar las tres bandas espectrales, y el etiquetando cada una de las clases (propuesto por Macedo *et al.* 2011).
4. Aplicar el modelo de clasificación y mezcla (propuesto en Macedo *et al*. 2011), para la clasificación de las imágenes.
5. Determinar las unidades de muestréo para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
6. Determinar el número de muestras para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
7. Definir un esquema de muestreo y aplicar el muestreo digital en las imágenes originales y en las clasificadas.

8. Calcular la matriz de error determinista y Fuzzy (Anderson et al. (1976), Congalton y Green, 2009, Macedo *et al*. 2011)
9. Determinar la precisión del clasificador para cada clase.

Escribir el reporte de prácticas señalado en los criterios para la entrega de ejercicios

Ejercicio 7.3. Analice el artículo adjunto (avena2011.pdf) y dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB programar y analizar lo siguiente:

1. Convertir las imágenes de modelo de color RGB a L*a*b.
2. A cada banda espectral aplicar el algoritmo de agrupamiento denominado k-means (propuestos por Hartigan, (1975)).
3. Aplicar el método de clasificación natural al combinar las tres bandas espectrales, y el etiquetando cada una de las clases (propuesto por Macedo *et al.* 2011).
4. Aplicar el modelo de clasificación y mezcla (propuesto en Macedo *et al*. 2011), para la clasificación de las imágenes.
5. Determinar las unidades de muestréo para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
6. Determinar el número de muestras para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
7. Definir un esquema de muestreo y aplicar el muestreo digital en las imágenes originales y en las clasificadas.
8. Calcular la matriz de error determinista y Fuzzy (Anderson et al. (1976), Congalton y Green, 2009, Macedo *et al*. 2011)
9. Determinar la precisión del clasificador para cada clase.

Escribir el reporte de prácticas señalado en los criterios para la entrega de ejercicios

Ejercicio 7.4. Analice el artículo adjunto (avena2011.pdf) y dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB programar y analizar lo siguiente:

1. Convertir las imágenes de modelo de color RGB a L*a*b.

2. A cada banda espectral aplicar el método de segmentación denominado thresholding Fuzzy (propuestos por Huang y Wang, (1995)).
3. Aplicar el método de clasificación natural al combinar las tres bandas espectrales, y el etiquetando cada una de las clases (propuesto por Macedo *et al.* 2011).
4. Aplicar el modelo de clasificación y mezcla (propuesto en Macedo *et al*. 2011), para la clasificación de las imágenes.
5. Determinar las unidades de muestréo para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
6. Determinar el número de muestras para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
7. Definir un esquema de muestreo y aplicar el muestreo digital en las imágenes originales y en las clasificadas.
8. Calcular la matriz de error determinista y Fuzzy (Anderson et al. (1976), Congalton y Green, 2009, Macedo *et al*. 2011)
9. Determinar la precisión del clasificador para cada clase.

Escribir el reporte de prácticas señalado en los criterios para la entrega de ejercicios

Ejercicio 7.5. Analice el artículo adjunto (avena2011.pdf) y dadas las imágenes que se adjuntan con la práctica en el modelo de color RGB programar y analizar lo siguiente:

1. Convertir las imágenes de modelo de color RGB a L*a*b.
2. A cada banda espectral aplicar los cuatro algoritmos de segmentación desarrollados en los ejercicios 7.1, 7.2, 7.3, 7.4, para obtener el valor del umbral.
3. Calcular el promedio del umbral determinado por los cuatro modelos aplicados en el inciso anterior.
4. Segmentar las tres bandas espectrales con el umbral promedio obtenido en el inciso anterior.
5. Aplicar el método de clasificación natural al combinar las tres bandas espectrales, y el etiquetando cada una de las clases (propuesto por Macedo *et al.* 2011).
6. Aplicar el modelo de clasificación y mezcla (propuesto en Macedo *et al*. 2011), para la clasificación de las imágenes.
7. Determinar las unidades de muestréo para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).

8. Determinar el número de muestras para la validación del clasificador, puede basarse en el propuesto en Macedo *et al*. (2011) o seleccionar algún otro (Congalton y Green, 2009).
9. Definir un esquema de muestreo y aplicar el muestreo digital en las imágenes originales y en las clasificadas.
10. Calcular la matriz de error determinista y Fuzzy (Anderson et al. (1976), Congalton y Green, 2009, Macedo *et al*. 2011)
11. Determinar la precisión del clasificador para cada clase.

Escribir el reporte de prácticas señalado en los criterios para la entrega de ejercicios

PROYECTO DE FIN DE CURSO

Objetivo

El participante será capaz de aplicar los conocimientos adquiridos en el curso, en un caso de estudio particular de reconocimiento de patrones y presentarlo ante la comunidad del postgrado.

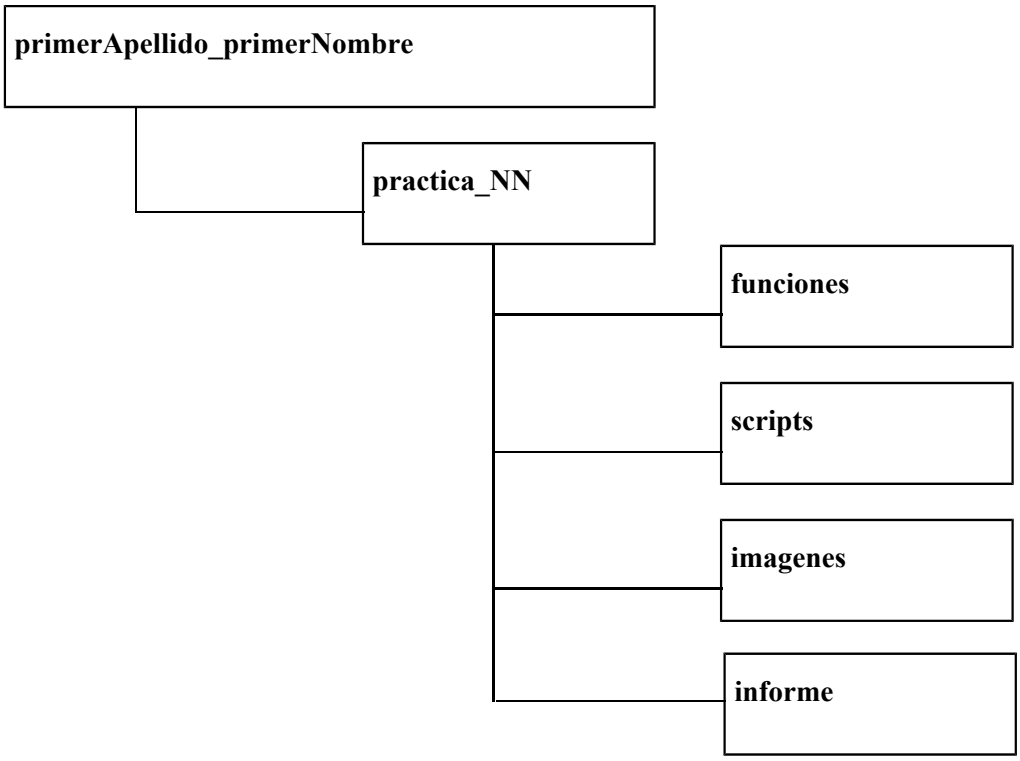
El proyecto de fin de curso puede ser seleccionado por el estudiante de acuerdo a su trabajo de investigación de tesis, o el participante puede seleccionar uno de varios temas propuestos por el profesor.

Criterios para la entrega el proyecto final

El trabajo es individual

Se entregará:

- Informe impreso con los resultados y conclusiones de la práctica.
- Archivo comprimido con el informe, los fuentes y las instrucciones para generar los resultados del informe. El archivo comprimido será en formato ".zip" y tendrá la siguiente estructura:



- En el directorio **funciones** se incluirán aquellas funciones pedidas en la práctica.
- En el directorio **scripts** se incluirán los programas principales que usan las funciones creadas y generan los resultados, gráficas, etc.. En este directorio se incluirá también un archivo de texto con las instrucciones para ejecutar los programas.
- En el directorio **scripts** se incluirá a modo de resumen un script (de nombre **main.m**) que muestre los resultados de la práctica.
- En el directorio **imagenes** se colocarán **solamente** las imágenes que se usen en la práctica y que no pertenezcan a la base de test (se entiende por base de test el conjunto de imágenes suministradas para la práctica del curso).
- En el directorio **imagenes** no se incluirán las imágenes resultado de la práctica. Éstas aparecerán en el informe y deberán poder generarse corriendo el script **main.m**.
- El acceso entre archivos de los directorios funciones, scripts, imagenes y base_test se hará mediante caminos relativos o suponiendo que los directorios pertenecen al path de Matlab.
- Se trabajará en Matlab (preferentemente). Salvo aclaración específica las funciones pedidas en las prácticas se programarán sin recurrir a bibliotecas existentes.

- Con todo lo anterior prepara una presentación en PowerPoint o imprimirá un cartel y lo presentará a la comunidad el día que le corresponda.

CONCLUSIONES

El objetivo de una aplicación de Visión Artificial es la extraer información útil de una escena con el fin de proceder a una toma de decisiones.

La organización de estas sesiones se basa en una primera parte de exposición de teoría por parte del profesor, y una segunda parte de trabajo autónomo del alumno resolviendo prácticas. La dinámica de las sesiones será fundamentalmente práctica, los alumnos estudiarán los conceptos teóricos y seguidamente realizarán ejercicios que facilitarán el aprendizaje. En el temario de estas sesiones se estudiarán los diferentes métodos y técnicas de tratamiento de imágenes digitales.

Las sesiones de laboratorio son el eje central de la asignatura y serán sesiones teórico- prácticas.

El desarrollo del temario ha seguido el proceso típico del procesado de una imagen: trasformar geométricamente la imagen, obtener contornos o filtrar el ruido, adaptar la estructura de los objetos sin perder información (morfología), extraer características de interés para la aplicación, clasificación de imágenes y evaluación de resultados.

Se ha diseñado una amplia colección de ejercicios que desarrollen las capacidades de los alumnos en programación y resolución de problemas de visión artificial. Los estudiantes deberán decidir la estrategia y aplicar los diferentes métodos para llegar a la solución del problema. Cada práctica engloba los conocimientos estudiados con anterioridad y los nuevos conocimientos aprendidos en la sesión.

Los ejercicios de aprendizaje junto con el proyecto final, afianzan los conocimientos de teoría y proporcionan a los alumnos destreza a la hora de programar y buscar soluciones a problemas de Visión Artificial.

Con todo ello los alumnos tendrán unos conocimientos-base muy amplios que les permitirán profundizar en técnicas más avanzadas de Visión Artificial en su futuro tanto académico (para su investigación de tesis) como profesional.

BIBLIOGRAFÍA BASICA

Libros

Bow, S. T. **2002**. *Pattern Recognition and Image Preprocessing*. 2nd ed. Marcel Dekker. New York, NY, USA. 698 p.

Cuevas E. D., Zaldívar. Pérez M. **2010**. *Procesamiento digital de imágenes con Matlab y Simulink*. Ed. Ra-Ma. 816p.

Congalton, R.G.; Green, K. **2009**. *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices*, 2nd ed.; CRC/Taylor & Francis: Boca Raton, FL, USA, 178p.

Gonzalez, Rafael; woods, Richard. **2008**. *Digital Image Processing***.** Tercera edición. Prentice Hall. 954 p. ISBN: 0131687288.

Hartigan, J.A. **1975**. *Clustering Algorithms;* Wiley: New York, NY, USA.

Macedo C., A. 2012. *Técnicas de clasificación automática de uso de suelos agrícolas y forestales basadas en imágenes digitales*. Tesis Doctoral. Universidad Complutense de Madrid, España.

Pajares G. y J. M. de la Cruz. **2007**. *Ejercicios resueltos de visión por computador.* Ed. Ra-Ma. 354p.

Pajares G. y J. M. de la Cruz. **2001**. *Visión por computador. Imágenes digitales y aplicaciones***.** Ed. Ra-Ma, ISBN 847897-472-5.

Sossa A. J. **2013**. *Visión Artificial: Rasgos descriptores para el reconocimiento de objetos*. Ed. Ra-Ma. 282p.

Theodoridis, S. and Koutroumbas, K. **2009**. *Pattern Recognition*. Fourth ed. Academic Press, ELSEVIER, California, USA.

Artículos

Anderson, J.R.; Hardy, E.E.; Roach, J.T.; Witmer, R.E. **1976**. A land use and land cover classification system for use with remote sensor data. In *Geological Survey*; Government Printing Office: Washington, DC ,USA,.

Congalton, R.G. **2004**. Putting the map back in map accuracy assessment. In *Remote Sensing and GIS Accuracy Assessment*; Lunetta, R.S., Lyon, J.G., Eds.; Lewis: Boca Raton, FL, USA, p. 292.

Huang, L.K.; Wang, M.J. **1995**. Image thresholding by minimising the measures of fuzziness. *Patt. Recog. 21*, 41-51.

Ridler, T.W.; Calvard, S. **1978.** Picture thresholding using an iterative selection method. *IEEE Trans. Syst. Man Cybern.*, *8*, 630-632.

Macedo-Cruz, A.; Pajares, G.; Santos, M.; Villegas-Romero, I. **2011.** Digital image sensor-based assessment of the status of Oat (Avena sativa L.) Crops after frost damage. *Sensors*, *11*, 6015-6036.

Otsu, N. **1979.** A threshold selection method from gray level histogram. *IEEE Trans. Syst. Man Cybern*, *9*, 62-66.

Páginas web:

MATLAB Image Processing Toolbox. (http://www.mathworks.es/products/image)

IEEE Xplore (http://ieeexplore.ieee.org)

BIBLIOGRAFIA COMPLEMENTARIA

Díaz de León, J. L. y H. Sossa. **1996.** On the Computation of the Euler Number of a Binary Object. *Pattern Recognition, 29* (3): 471-476.

González R. C., R. E. Woods, **2007.** *Digital Image Processing*, Prentice Hall, 3.a edición.

Abutaleb A. S., **1989.** Automatic Thresholding of Gray-Level Pictures using Two Dimensional Entropy. Computer, Vision, *Graphics and Image Processing, 47*:22-32.

Brink A. D., **1992.** Thresholding of Digital Images using Two-Dimensional Entropies. *Pattern Recognition. 25*: 803-808.

Chang C., K. Chen, J. Wang y M. L. G. **1994.** Althouse A Relative Entropy Approach to Image Thresholding. *Pattern Recognition 27*:1275-1289.

Cheng J., F-J. Chang y Sh. Chang. **1995.** A New Criterion for Automatic Multilevel Thresholding. *IEEE Transactions on Image Processing, 4*(3):370-378.

Fu K. S. & J. K. Muí, **1980**. A Survey on Image Segmentation. *Pattern Recognition, 13*:3-16.

González R. C. & R. E. Woods, **2002.** *Digital Image Processing*, Prentice Hall, Inc., Nueva Jersey.

Haralick R. M. & L. G. Shapiro. **1991.** *Computer and Robot Vision (Computer & Robot Vision)* Addison-Wesley.

Jain R., R. Kasturi y B. G. Schunck. **1995.** *Machine Vision*, McGraw-Hill Cience / Engineering / Math, l.a Edición.

Jiulun F. & X. Winxin. **1997.** Minimun Error Thresho lding: A Note. *Pattern Recognition Letters, 18*:705-709.

Kapur J. N., P. K. Sahoo y A. K. C. Wong. **1985.** A New Method for Gray-Level Picture Thresholding using the Entropy of the Histogram. Computer, Vision, *Graphics and Image Processing, 29*:273-285.

Kittler J. & J. Illingworth. **1986.** Minimun Error Thresholding. *Pattern Recognition, 19*(1):41-47.

Lee S. U. & S. Y. Chung. **1990.** A Comparative Performance Study of Several Global Thresholding Techniques. Computer, Vision, *Graphics and Image Processing, 52*:171-190.

Li C. H. & C. K. Lee. **1993.** Mínimum Cross Entropy Thresholding. *Pattern Recognition, 26*:617-625.

Li L., J. Gong & W. Chen. **1997.** Gray-Level Image Thresholding Based on Fisher Linear Projection of Two-Dimensional Histogram. *Pattern Recognition 30*(5):743-749.

De Luca A. & S. Termini. **1972.** *A Definition of a Non-probabilistic Entropy in the Setting of Fuzzy set Theory*, Instrumentation and Control. 20:301-312.

Morel J. M. & S. Solimini. **1994.** *Variational Methods in Image Segmentation: With Seven Image Processing Experiments (Progress in Nonlinear Differential Equations and Their Application)*, Birkhauser Boston.

Pal S. K. & A. Rosenfeld. **1988.** Image Enhancement and Thresholding by Optimization of Fuzzy Compactness. *Pattern Recognition Letters, 7*:77-86.

Pal N. R. & S. K. Pal. **1989.** Entropic Thresholding. *Signal Processing 16*:97-108.

Pal N. R. & S. K. Pal, 1993. A Review of Image Segmentation Techniques. *Pattern Recognition 26*:1277-1294.

Papamarkos N. & B. Gatos. **1994.** A New Approach for Multilevel Threshold Selection. *CVGIP: Graphical Models and Image Processing, 56*:357-370.

Pun T., **1981.** Entropic Thresholding a New Method. Computer Vision, *Graphics and Image Processing, 16*:210-239.

Renyi A. **1961.** On Measures of Entropy and Information. *Proc. Fourth Berkeley Symp. Math. Statist. Prob. 1960 1*:547-561, University of California Press, Berkeley.

Riddler T. W. & S. Calvard. **1978.** Picture Thresholding using Iterative Selection Method. *IEEE Transactions on Systems, Man and Cybernetics, 8*:630-632.

Sahasrabudhe S. C. & K. S. Das Gupta **1992.** *A Valley-Seeking Threshold Selection Histogram Technique.* CVGIP: Computer, Vision, Graphics and Image Processing, 55-65.

Sahoo P. K., S. Soltani, A. K. C. Wong y Y. Chen, **1988.** A Survey of Thresholding Techniques. CVGIP: Computer, Vision, *Graphics and Image Processing, 41*:233-260,

Sahoo P., C. Wilkins & J. Yeager, **1997.** Threshold Selection using Renyi's Entropy. *Pattern Recognition, 30*(1):71-84,

Trussell H. J., **1979.** Comments on Picture Thresholding using an Iterative Selection Method, *IEEE Transactions on Systems, Man and Cybernetics, 9*:311,

Weska J. S., **1978.** A Survey of Threshold Selection Techniques. *CVGIP: Computer, Vision, Graphics and Image Processing, 7*:259-265.

Weska J. S., R. N. Nagel y A. Rosenfeld. **1974.** A Threshold Selection Technique. *IEEE Transactions on Computers, 23*:1322-1326.

Wong A. K. C. & P. K. Sahoo, **1989.** A Gray-Level Thresholding Selection Method based on Maximum Entropy Principie. *IEEE Transactions on Systems, Man and Cybernetics, 19*:866-871.

Wu X-J., Y-J. Zhang & L-Z. Xia. **1999.** A Fast Recurring Two Dimensional Entropic Thresholding Algorithm. *Pattern Recognition, 32*:2055-2061,

Yager R. R., **1979.** On the measure of fuzziness and negation. Part 1: Membership in the unit interval. International *Journal of General Systems, 5*:221-229.

Yan H., **1996.** Unified Formulation of a Class of Image Thresholding Techniques, *Pattern Recognition 29*(12):2025-2032.

Zadeh L. A., **1975.** *Fuzzy sets and their applications to cognitive and decision processes*. pp. 01-39, Academic Press, London.

Abdula W. H., A. O. M. Saleh & A. H. Morad, **1988.** A Preprocessing Algorithm for Hand-Written Character Recognition, *Pattern Recognition Letters, 7*:13-18.

Arceli C., **1981.** Pattern Thinning by Contour Tracing, *Computer Graphics and Image Processing, 17*:130-144.

Arrebola F., A. Bandera, P. Camacho & F. Sandoval. **1997.** Corner Detection by Local Histograms of Contour Chain Code, *Electronic Letters, 33*(21):1769-1771.

Arumigam A., T. Radhakrishnan, C. Y. Suen & P. S. P. Wang, **1993.** A Thinning Algorithm based on the Force Between Charged Particles, *International Journal of Pattern Recognition and Artificial Intelligence, 7*(5):987-1008.

Beaudet P. R., **1978.** *Rotational Invariant Image Operators 4th. International Conference on Pattern Recognition*, Tokyo, 579-583.

Benedetti A. & P. Perona. **1998.** *Real Time 2-D Feature Detection on a Reconfigurable Computer, Proceedings of the International Conference on Computer.* Vision and Pattern Recognition, pp. 586-593,

Beus H. L. and S. S. H. Tiu. **1987.** An Improved Corner Detection Algorithm based on Chain-Coded Plane Curves. *Pattern Recognition, 20*(3):291-296.

Blum B., **1964.** *A Transformation for Extracting New Descriptor of Shape, Proc. of Symposium on Models for Perception, Speech and Visual Form.* Cambridge, MA.

Chang H.-H., **1996.** Skeletonization of Binary Digital Patterns using a Fast Euclidean Distance Transformation. *Optical Engineering, 35*(4):1003-1008.

Chen Y.-Sh. & W.-H. Hsu, **1988.** A Modified Parallel Algorithm for Thinning Digital Patterns. *Pattern Recognition Letters, 7*:99-106.

Chen Y.-Sh. & W.-H. Hsu. **1989.** A Systematic Approach for Designing 2-Subcycle and Pseudo 1-subcycle Parallel Thinning Algorithms. *Pattern Recognition 22*(3):267-282.

Chen Y.-Sh. & W.-H. Hsu. **1990.** A Comparison of Somne One-Pass Parallel Thinnings. *Pattern Recognition Letters, 11*:35-41.

Cheng F. H. & W. H. Hsu. **1988.** Parallel Algorithm for Corner Finding on Digital Curves. *Pattern Recognition letters, 8*:47-53.

Chu Y. K. & Ch. Y. Suen. **1986.** An Alternate Smoothing and Stripping Algorithm for Thinning Digital Binary Patterns. *Signal Processing, 11*:207-222.

Datta A. & S. K. Parui. **1994.** A Robust Parallel Thinning Algorithm for Binary Images. *Pattern Recognition, 27*(9):1181-1192.

Davies E. R. & A. P. N. Plummer. **1981.** Thinning Algorithms: A Critique and a New Methodology. *Pattern Recognition, 14*(1-6):53-63.

Deriche R. & G. Giraudon. **1993.** A Computational Approach for Corner and Vertex Detection. *International Journal on Computer Vision, 10*(2):101 -124.

Deutsch E. S. **1972.** Thinning Algorithms on Rectangular, Hexagonal and Triangular Arrays. *Communications of the ACM, 15*(9):827-837.

Fayolle J., L. Riou & Ch. Ducottet. **2000.** Robustness of a Multiscale Scheme of Feature Points Detection. *Pattern Recognition, 33*:1437-1453.

Eckhardt U., **1988.** A Note on Rutowitz Method for Parallel Thinning. *Pattern Recognition Letters, 8*:35-38.

Fischler M. A. & H. C. Wolf. **1994.** Locating Perceptually Salient Points on P1anar Curves. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 16*(2): 113-129.

Forstner W. **1986.** A Feature Based Correspondence Algorithm for Image Matching International Arch Photogrametry. *Remote Sensing, 26*:150-166.

Gonzalez R. C. & R. E. Woods, **1992.** *Digital Image Processing.* Addison Wes1ey.

Govindan V. K. & A. P. Shivaprasad **1987.** A Pattern Adaptive Thinning Algorithm. *Pattern Recognition, 20*(6):623-637.

Harris C. & M. Stephens. **1988.** *A Combined Corner and Edge Detector.* Proc. of 4th. Alvey Vision Conference, Manchester, Vol. 15:147-151.

Hilditch C. J. **1969.** *Linear Skeletons form Square Cupboards, Machine Intelligence IV, B. Meltzer and D. Michie Eds.* American Elsevier, Nueva York, pp. 403-420.

Ho1t Ch. M., A. Stewart, M. Clint & R. H. Perrot. **1997.** An Improved Parallel Thinning Algorithm. *Communications of the ACM, 30*(2):156-160.

Jain R., R. Kasturi & B. G. Schunck. **1995.** *Machine Vision.* McGraw-Hill Science/Erigineering/Math.

Jang B. K. & R. T. Chin. **1992.** One-Pass Parallel Thinning: Analysis, Properties and Quantitative Evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 14*(11): 1129-1140.

Kitchen L. & A. Rosenfeld. **1982.** Gray-Level Corner Detection. *Pattern Recognition Letters 1*:95-102.

Kwok P. C. K., 1988. A Thinning Algorithm by Contour Generation. *Communications of the ACM, 31*(11):1314-1324.

Lam L., S.-W. Lee & Ch. Y. Suen. **1992.** Thinning Methodologies-A Comprehensive Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 14*(9):869-865.

Li X. & N. S. Hall. **1993.** Corner Detection and Shape Classification of on-Line Handprinted Kanji Strokes. *Pattern Recognition, 26*(8): 1315-1334.

Lin R. S., Ch. H. Chu & Y. Ch. Hsueh. **1998.** A Modified Morphological Corner Detector. *Pattern Recognition Letters, 19*:279-286.

Liu H. C. & M. D. Srinath, **1990.** Corner Detection from Chain-Code. *Pattern Recognition, 23*:51-68.

Maravall D., **1993**. *Reconocimiento de formas y visión artificial*. Addison Wesley Iberoamericana.

Moravec H. P., **1977.** *Towards Automatic Visual Obstacle Avoidance.* Fifth International Joint Conference on Artificial Intelligence, pp. 584.

Naccache N. J. & R. Shinghal. **1984.** STPA: A Proposed Algorithm for Thinning Binary Patterns. *IEEE Transactions on Systems, Man, and Cybernetics, 14*(3):409-418.

Plamondon R., Ch. Y. Suen, M. Bourdeau & C. Barriere. **1993.** Methodologies for Evaluating Thinning Algorithms for Character Recognition. *International Journal of Pattern Recognition and Artificial Intelligence, 7*(5):1247-1270.

Quddus A. & M. M. Fahmy, **1999.** Fast Wavelet-Based Corner Detection Technique. *Electronic Letters, 35*(4):287-288.

Rosenfeld A. & E. Johnston. **1973.** Angle Detection on Digital Curves. *IEEE Transactions on Computers, 22*:875-878.

Rosenfeld A. & J. S. Weska **1975.** An Improved Method of Angle Detection on Digital Curves. *IEEE Transactions on Computers, 24*:940-941.

Schmid C., R. Mohr & Ch. Bauckhage. **2000.** Evaluation of Interest Point Detectors. *International Journal of Computer Vision, 37*(2):151-172.

Shi J. & C. Tomasi **1994.** Good Features to Track, Proceedings of the International Conference on Computer *Vision and Pattern Recognition,* pp. 593-600.

Sohn K., J. H. Kim & W. E. Alexander. **1998.** A Mean Field Annealing Approach to Robust Corner Detection", *IEEE Transactions on Systems, Man and Cybernetics, 28*(1):82-90.

Sossa H., **1989.** *An Improved Parallel Algorithm for Thinning Digital Patterns.* Pattern Recognition Letters, pp. 77-80.

Stefanelli R. & A. Rosenfeld. **1971.** Some Parallel Thinning Algorithms for Digital Pictures. *Journal of the Association of Computing Machinery, 18*(2):255-264.

Susuki S. & K. Abe, **1987.** Binary Picture Thinning by an Iterative Parallel Two-Subcycle Operation. *Pattern Recognition, 20*(3):297-303.

Tomasi C. & T. Kanade, **1991.** *Detection and Tracking of Point Features, Technical Report* CMU-CS-91-132, Carnegie Melon University, Pittsburgh, PA, April.

Tommasini T., A. Fusiello, E. Trueco & V. Roberto, **1998.** *Making Good Features Track Better*. Proceedings of the International Conference on Computer Vision and Pattern Recognition, pp. 178-183.

Trujillo L. & G. Olague. **2008.** Automated Design of Image Operators that Detect Interest Points. *Evolutionary Computation 16*(4):483-507.

Tsai D. M., H. T. Hou & H. J. Su. **1999.** Boundary Based Corner Detection using Eigenvalues of Covariance Matrix. *Pattern Recognition Letters, 20*:31-40.

Zhang T. Y. & Ch. Y. Suen. **1984.** A Fast Parallel Algorithm for Thinning Digital Patterns. *Communications of the ACM, 27*:236-239,

Zheng Zh., H. Wang & E. K. Teoh. **1999.** Analysis of Gray Level Corner Detection. *Pattern Recognition Letters, 20*:149-162.

Zitova B., J. Kautsky, G. Peters & J. Flusser. **1999.** Robust Detection of Significant Points in Multi Frame Images. *Pattern Recognition Letters, 20:*199-206.