Prácticos 2023

11 de abril de 2023

Evaluación

Ejercicios prácticos para la regularización y aprobación directa de la materia. Cada práctico tiene un puntaje máximo asignado que se muestran en la tabla a continuación.

Número	Tema	Puntaje
Práctico 1	Eventos y callbacks	10
Práctico 2	Transformación Afín	10
Práctico 3	Rectificación	10
Práctico 4	Medición	10
Práctico 5	ARuCo	10
Práctico 6	Clasificación	15
Práctico 7	Detección	15
Práctico 8	Caso de Estudio	30
Publicación		50
Proyecto		50
Total		210

- Para la aprobación hay que sumar 60 puntos o más y defender al final del cursado uno o más prácticos con un coloquio individual.
- La nota sale del siguiente código python en donde puntos es la cantidad de puntos obtenidos:

```
if(puntos >= 100):
    nota = 10
else:
    nota = int(puntos/10) + coloquio
```

con $\text{coloquio} \in \mathbb{Z}$.

Práctico 1: Eventos y callbacks

```python
#! /usr/bin/env python
# -*- coding: utf-8 -*-

import cv2
import numpy as np

drawing = False  # true if mouse is pressed
mode = True  # if True, draw rectangle. Press 'm' to toggle to curve
ix, iy = -1, -1

def draw_circle(event, x, y, flags, param):
    global ix, iy, drawing, mode
    if event == cv2.EVENT_LBUTTONDOWN:
        drawing = True
        ix, iy = x, y
    elif event == cv2.EVENT_MOUSEMOVE:
        if drawing is True:
            if mode is True:
                cv2.rectangle(img, (ix, iy), (x, y), (0, 255, 0), -1)
            else:
                cv2.circle(img, (x, y), 5, (0, 0, 255), -1)
    elif event == cv2.EVENT_LBUTTONUP:
        drawing = False
        if mode is True:
            cv2.rectangle(img, (ix, iy), (x, y), (0, 255, 0), -1)
        else:
            cv2.circle(img, (x, y), 5, (0, 0, 255), -1)

img = np.zeros((512, 512, 3), np.uint8)
cv2.namedWindow('image')
cv2.setMouseCallback('image', draw_circle)
while(1):
    cv2.imshow('image', img)
    k = cv2.waitKey(1) & 0xFF
    if k == ord('m'):
        mode = not mode
    elif k == 27:
        break
cv2.destroyAllWindows()
```

Usando como base el programa anterior, crear un programa que permita seleccionar una porción rectangular de una imagen con el ratón, luego

- con la letra “g” lo guardamos a disco en una nueva imagen y salimos
- con la letra “r” restauramos la imagen original y volvemos a realizar la selección
- con la “q” salimos.

Práctico 2: Transformación afín - Incrustando imágenes

Teniendo en cuenta que:

- Una transformación afín se representa con una matriz de 2×3
- Tiene **6** grados de libertad y puede ser recuperada con **3** puntos

Usando como base el programa escrito en la práctica 1, se pide:

- Crear un programa que permita seleccionar con el ratón 3 puntos de una primera imagen.
- Luego crear un método que compute una transformación afín entre las esquinas de una segunda imagen y los 3 puntos seleccionados.
- Por último aplicar esta transformación sobre la segunda imagen, e incrustarla en la primera imagen.

Ayuda

- **`cv2.getAffineTransform`**
- **`cv2.warpAffine`**
- Generar una máscara para insertar una imagen en otra

Práctico 3: Rectificando imágenes

Teniendo en cuenta que:

- Una homografía se representa con una matriz de 3×3
- Tiene **8** grados de libertad y puede ser recuperada con **4** puntos

Usando como base el programa anterior, se pide:

- Crear un programa que permita seleccionar 4 puntos de una primera imagen.
- Luego crear un método que compute una homografía entre dichos 4 puntos y una segunda imagen estándar de $m \times n$ pixeles.
- Por último aplicar esta transformación para llevar (rectificar) la porción de la primera imagen definida por los 4 puntos elegidos a la segunda de $m \times n$.

Ayuda

- **`cv2.getPerspectiveTransform`**
- **`cv2.warpPerspective`**

Práctico 4: Medición de objetos sobre plano calibrado

La siguiente imagen es una foto de la fachada de una casa.

Figura 1: Foto de ejemplo. Sacar una foto propia.

Tomando esta foto como ejemplo y considerando como dato conocido el tamaño del vano de la puerta, 2.10m de alto por 0.73m de ancho, se pueden determinar las dimensiones de cualquier objeto de la imagen siempre que esten en el mismo plano de la fachada.

Como práctica del tema se pide:

- capturar una imagen propia con perspectiva, que puede ser de una fachada o cualquier otra escena plana, que tenga un objeto rectangular de dimensiones conocidas,
- escribir un programa que permita encontrar la transformación perspectiva entre los 4 vértices del objeto en la imagen y un rectángulo con la misma relación de aspecto que el objeto real,
- que luego de elegir los 4 vértices aplique dicha transformación a la imagen y la muestre en una nueva ventana “calibrada para medición”,
- sobre esta ventana de medición el programa debe permitir que el usuario elija con el ratón dos puntos cualquiera y debe mostrar la distancia en metros entre dichos puntos;
- la ventana de medición debe poder ser restaurada cuando se presione la tecla “r” para realizar una nueva medición;
- por último medir dos objetos en la imagen (ventana, casilla del gas, etc.) y verificar los resultados con la medida real.

Práctico 5: Práctico Libre ArUCo

En base al teórico de ArUCo y a los videos presentados, se propone realizar una aplicación simple en base a ArUCo. Puede ser por ejemplo realidad virtual en 2D o en 3D, estimación de posición y orientación de objetos en movimiento, seguimiento de objetos, o simplemente hacer instalar algún repositorio interesante y aplicarlo a algo.

- `https://www.youtube.com/watch?v=nsu9tNIJ6F0`
- `https://youtu.be/VsIMl8O_F1w?t=55`
- `https://www.youtube.com/watch?v=qqDXwKDr0vE`
- `https://www.youtube.com/watch?v=RNwTGPvuhPw&t=8`
- `https://www.youtube.com/watch?v=xzG2jQfxLlY`
- `https://www.youtube.com/watch?v=jwu9SX3YPSk`
- `https://www.youtube.com/watch?v=FBl4Y55V2Z4`

El puntaje del práctico dependerá de la complejidad, utilidad o nivel de innovación de la aplicación.

Práctico 6 - Clasificación de imágenes usando CNN

En este práctico vamos a implementar un clasificador de imágenes usando CNN. El objetivo será diseñar y entrenar un algoritmo que en forma automática clasifique una imagen como correspondiente a una clase entre varias. Las clases las vamos a elegir nosotros (perros y gatos, autos y motos, o algo más interesante o útil).

Para probar el modelo, se deben buscar 2 imágenes nuevas, mostrarlas en pantalla (matplotlib), clasificarlas con el modelo y mostrar el resultado de la clasificación (puntaje para cada clase).

Usar plantilla disponible en colab.

Práctico 7 - Detección de objetos usando CNN

El objetivo de este práctico es entrenar un detector de objetos en imágenes usando la red Yolo.

Para probar el modelo buscar 2 imágenes nuevas, mostrarlas en pantalla (matplotlib), detectar objetos con el modelo y mostrar el resultado de la detección (recuadro que indique la ubicación y clase del objeto).

Usar template disponible en colab.

Práctico 8 - Caso de estudio en una empresa

En este práctico se deberá seleccionar una empresa a la cual se tenga acceso y en dicha empresa se deberán investigar que problemas se pueden resolver o que mejoras se pueden realizar usando técnicas de visión por computadora.

En base a la búsqueda se deberá realizar lo siguiente:

- indicar que algoritmos se podrían utilizar para resolver el problema justificando la propuesta (con aplicaciones que ya lo realizan o con el sustento teórico correspondiente),
- armar un diagrama en bloques en donde se distingan el orden de las diferentes etapas del algoritmo y
- seleccionar el hardware necesario para correr el algoritmo en planta junto con un estimativo de costos del mismo.

Publicación/presentación

Realizar una publicacación/presentación en un congreso (como por ejemplo las jornadas de ciencia y tecnología que se realizarán en nuestra facultad).

Proyecto

Realizar un proyecto que use visión por computadora: proyecto final, tesis o aplicación propuesta en el práctico 8.