Percepción - Práctica 2

Percepción y Control para Sistemas Empotrados

Curso 2020-21

Izar Castorina – Grupo 1

En esta práctica, se nos ha pedido crear un programa capaz de detectar líneas verticales (± 10 grados de inclinación) en un set de imágenes que nos ha sido proporcionado. Efectuando varias pruebas, había que encontrar el número mínimo de columnas que permitan encontrar los bordes de las puertas presentes en cada imagen.

1 Extracción de contornos

Para crear un mapa de bordes ha sido empleado el algoritmo de extracción de contornos de Canny, integrado en la librería `scikit-image`. Antes de proceder con la extracción, la imagen ha sido convertida a escala de grises y ecualizada con la función `exposure.equalize_hist`. Este ha sido preferido al realizar una ecualización con CLAHE porque su mayor "agresividad" permite evidenciar mayormente los contornos de los objetos presentes en la escena. Los parámetros empleados para el algoritmo de Canny han sido `Sigma = 1`, `low_threshold = 0.2` y `high_threshold = 3`. El mapa de contornos resultante puede ser visualizado si la opción `only_show_result` del programa está desactivada.

2 Transformada de Hough

La transformada de Hough es un algoritmo que nos permite encontrar formas geométricas como líneas y círculos dentro de una imagen. En nuestro caso, vamos a emplearla para detectar líneas verticales con inclinaciones entre -10 y 10 grados.

Los grados de inclinación, `theta` en el algoritmo, nos permiten restringir la búsqueda y encontrar solo líneas aproximadamente verticales. En el código, los dos límites de -10 y +10 grados están definidos como `min_grados` y `max_grados` respectivamente, y a los dos valores ha sido sumado 90. Esto es debido a que nuestra referencia es 0 grados = línea completamente vertical, pero el programa trata imágenes, que tienen su origen de ejes en la esquina superior izquierda. Por tanto, lo que para nosotros serían 0 grados, para el programa sería una línea a 90 grados de inclinación.

Los parámetros `delta_theta` y `delta_rho` determinan respectivamente con cuánta precisión se deberían buscar líneas y cada cuántos píxeles de la imagen. Reducirlos garantiza más precisión y más posibilidades de encontrar líneas verticales, pero aumenta los tiempos de procesamiento.

El acumulador de Hough es una matriz de tamaño p x q en las cuales se irán almacenando los valores que indicarán cuántas curvas pasan por el punto correspondiente de la imagen, en nuestro caso, el mapa de bordes de la imagen original. Las dimensiones se han calculado con las siguientes fórmulas:

$$p = \frac{max\_grados - \min\_grados}{\Delta\theta} + 1$$

$$q = \frac{diagonal * 2}{\Delta\rho} + 1$$

La matriz se llena de ceros, y luego para cada pixel de imagen de bordes se calcula la ρ y la θ del punto actual, y se convierten estos parámetros en dos números enteros, s y t, que son coordenadas dentro de la matriz:

$$s = int(\frac{\theta_k - \min\_grados}{\Delta\theta} + 1)$$

$$t = int(\frac{\rho_k + diagonal}{\Delta\rho} + 1)$$

θ_k es el valor en grados que va de -10 (+90) hasta +10 (+90), mientras que ρ_k es dado por la siguiente expresión:

$$\rho_k = int(u * \cos\theta_k + v * sin\ \theta_k)$$

con u y v siendo respectivamente coordenada vertical y horizontal de la imagen en pixeles, y θ_k expresado en radiantes (multiplicado por $\pi/180$).

Una vez encontradas estas dos coordenadas, se suma 1 a la celda correspondiente, se incrementa θ_k de $\Delta\theta$ y se sigue hasta llegar a max_grados. Hecho esto, se pasa al siguiente pixel de la imagen de contorno.

3 Representación de líneas verticales

Una vez llenada la matriz, se procede a encontrar los valores máximos. El número de máximos a encontrar ha sido definido como 8 (constante `N_MAXIMOS`). Cada máximo se almacena en una lista, y su posición en la matriz, junta a la de sus 6 vecinos adyacentes (valor definido en la variable `intervalo`), se vuelve a poner a cero, para evitar encontrar líneas muy cercanas y también evitar que se vuelva a encontrar el mismo pico dentro de la matriz en la siguiente iteración de este proceso.

Con los máximos encontrados, se realiza la operación inversa, o sea calcular la θ y la ρ a partir de la s y la t, o sea las dos coordenadas de la matriz. Gracias a esos dos parámetros, y fijando la coordenada v como la línea a mitad de altura de la imagen, podemos determinar la coordenada u, o sea la posición horizontal de la línea correspondiente al pico. Una vez tratados los primeros 4 máximos y determinadas las coordenadas horizontales, se procede a pintar líneas rojas en correspondencia de estos valores, y visualizar la imagen final en pantalla.

4 Otras funciones del programa y conclusiones

Enable strected, factor de stretching, onlyshowresult, N encontrada, número de máximos a procesar, intervalo de celdas a limpiar, menú de usuario, conversión a eventual nodo de ROS – Python 2 no debería afectar, menos alguna conversión necesaria a float para evitar problemas, todas las imágenes resultantes han sido guardadas en la carpeta de Resultados

Este programa ha sido originariamente pensado para ser un nodo para el ROS, por tanto, las varias partes están definidas como funciones que aceptan parámetros sencillos (por ejemplo, una sola imagen), para que su eventual conversión a nodo pueda realizarse sin cambiar muchas partes del código. Habría únicamente que comprobar si las operaciones que realizan divisiones necesitan conversiones de numerador y/o denominador a tipo `float`, para que el resultado de la división no se convierta automáticamente en entero como es comportamiento esperado en versiones de Python inferiores a la 3.0.

Sin embargo, dado que la entrega es de un programa *standalone*, he decidido implementar un pequeño menú de usuario, en el cual se puede decidir qué imágenes procesar, y cambiar algunos parámetros:

- **`enable_stretched`** permite habilitar el stretching vertical de las representaciones gráficas de los acumuladores de Hough calculados; dado que suele haber diferencias substanciales entre los valores de p y q, las dos dimensiones del mismo acumulador, puede resultar difícil a veces ver gráficamente los resultados de la operación. Como p suele ser el parámetro más pequeño, esta opción permite multiplicar por 5 (valor por defecto, que puede ser modificado manualmente, almacenado en la variable `stretching`) la dimensión vertical del gráfico.
- **`only_show_result`** permite evitar que para cada imagen se visualicen mapa de bordes, acumulador de Hough, acumulador con celdas de pico vaciadas, y resultado final, limitando el output a sólo este último. Si se quieren procesar las 5 imágenes proporcionadas a la vez, se acaba con 20 ventanas abiertas, y a veces solo se quiere visualizar dónde el programa cree que estén las líneas verticales.

Otros parámetros, como `N`, `N_MAXIMOS` e `intervalo`, o los grados de pueden ser modificados exclusivamente cambiando manualmente el código fuente, al contrario de los dos parámetros mencionados previamente, que pueden

modificarse en tiempo de ejecución a través del menú de usuario. La N ha sido determinada de manera empírica, y 4 ha resultado ser el número mínimo de líneas, al menos para la configuración actual del programa, que permite encontrar en todas las imágenes los bordes de las puertas de manera fiable.