Feature matching para la detección de objetos

Romero Erikson, Reiny David, y Gómez Diego
{u1201732, u1201730, y u1201714}@unimilitar.edu.co
Visión por computador 2019-2

Resumen — En el presente documento, se explica los fundamentos, objetivos, desarrollo y resultados de la implementación de un sistema para el reconocimiento de objetos mediante el uso del descriptor de características ORB. El sistema está diseñado de tal forma que se pueda detectar cuatro diferentes objetos, establecidos por su idoneidad para el proyecto planteado y desarrollado.

Palabras clave — características, descriptor, detección, objetos, ORB.

I. Objetivo del trabajo

El objetivo principal del trabajo era el desarrollo de un sistema de detección de objetos mediante la utilización de algún descriptor de características como SIFT, SURF, ORB u otro similar.

Para ello, se requería generar una número considerable de puntos en común, además de poder visualizar el recuadro delimitador y nombre del objeto que se vaya reconociendo a lo largo de la ejecución. Igualmente, el número de objetos a reconocer, fue establecido como cuatro.

II. Marco teórico

ORB

Este descriptor de características, podría ser considerado como una combinación del detector *FAST* junto con el descriptor *BRIEF [3]*.

Para realizar la implementación del *ORB*, se debe tener en cuenta el procedimiento que se realiza, el cual inicia con la determinación de puntos claves mediante *FAST*, para luego utilizar *Harris* para encontrar la cantidad *n* de puntos.

El *FAST* no calcula la orientación y rotación variable, si no el centroide a través de la intensidad de una región con una esquina en el centro *[3]*, del cual el vector de dirección hacia el centroide determina la orientación.

Así mismo, el descriptor *BRIEF* es utilizado en el caso de existir una rotación dentro de un plano. Pero en *ORB* la matriz de rotación es calculada mediante la orientación, previamente calculada, y los los valores obtenidos acorde a la orientación, al usar *BRIEF*.

Por último, los momentos son hallados debido a que mejoran procesos de rotación invariante, tal como se muestra en las siguientes fórmulas:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y)$$

$$C = \left(\frac{m_{10}}{m_{00}}, \frac{m_{01}}{m_{00}} \right)$$

$$\theta = \text{atan2}(m_{01}, m_{10})$$

Respecto al desempeño obtenido en la utilización del *ORB*, y en comparación con métodos como *SIFT* o *SURF*, es más rápido, basado en el algoritmo del cual hace uso. Así mismo, este método (*ORB*) se concentra principalmente en objetos o elementos los cuales se encuentren en el centro de la imagen digital de entrada, a diferencia de sus similares, en donde la distribución es más amplia sobre el mapa de bits.

III. Desarrollo

Para el proceso de desarrollo, se utilizó el lenguaje de programación *Python* junto con la librería de procesamiento de imp̀agenes digitales y visión por computador, *OpenCV*.

Esas dos herramientas, fueron seleccionadas debido a su amplia difusión, documentación y soporte, tanto oficial como de comunidad, en diferentes temáticas de transformaciones, detección, extracción de características, entre otros.

A continuación, se presenta el orden secuencial y esquemático del sistema desarrollado:

IV. Resultados

Una vez, con el proceso de desarrollo completado, a continuación se presenta los resultados obtenidos durante el procedimiento inicial de ejecución del código fuente.

A continuación, se presenta los resultados obtenidos para los diferentes objetos:

V. Conclusiones

A partir del proceso de desarrollo y de obtención de resultados, se pueden realizar las siguientes conclusiones:

- El descriptor de características *ORB* presenta un buen rendimiento, pero como fue mencionado en el breve marco teórico de este documento, este método se concentra más en elementos que se hallen en el centro de la
- El lenguaje de programación *Python*, ofrece muy buenas capacidades para el desarrollo de sistemas computacionales enfocados al procesamiento de señales e imágenes digitales, visión por computador, ciencia de datos y demás áreas; mediante la fácil integración de módulos o librerías al desarrollo, tal como en este caso, que se utilizó las capacidades de *OpenCV*.

Referencias

1. OpenCV (Documentación). Feature Matching. URL: https://docs.opencv.org/master/dc/dc3/tutorial_py_matcher.html (visitado 30-10-2019).
2. Abid K (abidrahmank).. ORB (Oriented FAST and Rotated BRIEF). URL: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_feature2d/py_orb/py_orb.html (visitado 30-10-2019).
3. E. Karami, S. Prasad & M. Shehata. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images. Faculty of Engineering and Applied Sciences, Memorial University, Canada. URL: https://arxiv.org/pdf/1710.02726.pdf (visitado 30-10-2019).