Color tracker: using *OpenCV* for automatic color-aware region selection

Lucas S. Althoff Student
ls.althoff@gmail.com
Teofilo de Campos Prof
teo.decampos@gmail.br

Departamento de Ciência da Comptutação
Universidade de Brasília
Campus Darcy Ribeiro, Asa Norte
Brasília-DF, CEP 70910-900, Brazil,

Abstract

When one needs to compute features from an image or video frames the first stage of computation commonly resides in region selection. In this sense, automatic region selection in visual data is an essential task for computer vision area. Applications dealing with computer vision tasks, i.e object detection, ROI selection mechanism, image segmentation, instance detection, visual sumarization, automatic cinematography and others.

The purpose of this work is to explore OpenCV package giving simple solutions for real-time video and image user interaction functionallities. To do that we propose four specific goals: 1. Present a simple user-interface system that allows users to navigate in an image or video frames tracking pixels coordinate (row,column); 2. Automatically mark image/frame regions that has a color within a tolerance factor relative to the selected pixel; 3. Incorporate this interactive functionality to a real-time webcam video streaming

1 Introduction

User interaction for image feature handling represents a fundamental functionality for a plenty of computer vision applications. Where user interactions, background data processing, remote requests, and streaming data read and modify the interface at the same time. Such interactive behaviors leverage users freedom to explore features in real-time [7].

When one needs to compute features from an image or video frames the first stage of computation commonly resides in region selection. In this sense, automatic region selection in visual data is an essential task for computer vision area. More specifically, region selection can be done in an interactive manner based on some device. The simpler one are mouse clicks or mouse movements, aka mouse events.

This kind of mouse events traking is usefull to select regions in real-time and then to extract features that could not be codefied before. It can also be used to interactively define objects to be tracked with a variety of applications, like sport analysis [6] or with educational goals [4], [5]. Other Other implemantations propose interactive big urban trajectory exploratory analysis [8] automatic selection of scene instances, . These systems can also be used for building human-based dataset annotation for supervised-learning of machine learning models, *i.e* input to an image classification algorithm [1] or predictive models [3].

The purpose of this work is to explore *OpenCV* package which gives simple solutions for mouse events handling [2], and finally propose a color tracker system for images, videos and

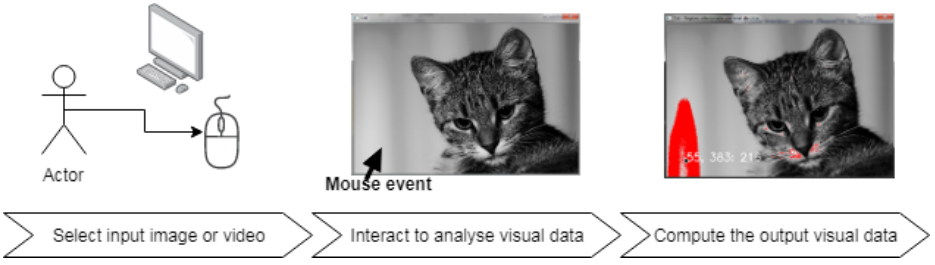

Figure 1: The three main stages of our proposed color tracker system

live-streamings. To do that we assume four specific goals: 1. Present a simple user-interface system that allows users to navigate in an image or video frames tracking pixels coordinate (row,column); 2. Automatically mark image/frame regions that has a color within a tolerance factor relative to the selected pixel; 3. Incorporate this interactive functionality to a real-time webcam video streaming.

2 Methodoly

In the following section, we explain our developed color-aware region selection. Based on single mouse events, our system allows the user to specify the pixel that will be reference select pixels from the entire image within a tolerance factor of proximity.

2.1 Analisys statement

Our algorithm have a control system that interactively asks what kind of midia the user want to analyse and to specify directily the path. The user have the freedom to explore visual data from diferent folder paths. But objectively, in order of answer, the user determine 1. the kind of midia (image, video or webcam) to be analysed 2. display the image or frame in the window 3. wait for mouse event selecting one pixel 4. print position and intensity/color information of the selected pixel

2.2 Region selection

The essential *OpenCV* function used for mouse event handling are the cv2.setMouseCallback(), which permits the pixel-based computation of the color or intensity proximity. In the case of color image we implemented the euclidian distance of each color feature (red, green and blue intensity) from the clicked pixel and each pixel of the image or frame.

$$T > \sqrt{\left((R_c - R_i)^2 + (G_c - G_i)^2 + (B_c - B_i)^2\right)} \quad (1)$$

In this work we used a fixed Tolerance factor (T) of 13.

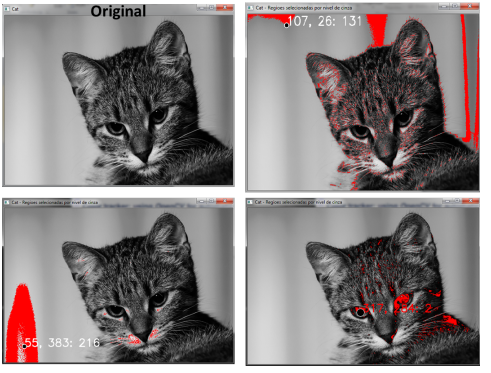

Figure 2: Gray image analysis showing three different selected pixels and its relative region.

3 Results

We organized the results in the four kinds of media analysed. 2 shows three different selected pixels and its relative region. For color images 3 shows how the better performance when selecting dark pixels.

4 Discussion

We observed from experiments with different types of image that region selection have a low effective result for detecting regions, with better performance when selecting dark pixels, this problem resides on the low explanation of a single pixel that alone can not represent a color region. For example, inside a set of red pixels, there can be some white pixels, the cause of this pontualization can be explained with the digital image formation which have a process of halftoning that limits a color region to be resolved with more then one pixel.

5 Final Remarks

We conclude from this work that OpenCV offers simple solutions to handle mouse events and other interactive operabilities with computer vision systems. Eventhough, our results showed that luminosity interfer too much in pixel , some kind of interpolation of a region could behaviour more effectively So, as a future study we could try to adjust the Tolerance factor or to improve performance considering not only one pixel but a set of pixel smoothed to represent better that color of color region selection.

References

[1] Abhishek Dutta and Andrew Zisserman. The VIA annotation software for images, audio and video. In *Proceedings of the 27th ACM International Conference on Multimedia*, MM '19, New York, NY, USA, 2019. ACM. ISBN 978-1-4503-6889-6/19/10.

Figure 3: Color image analysis the first row show the originals second and third row show the better performance when selecting dark pixels.

Figure 4: Three frames of a analysed video. The limited effectivety of region selection based on color can be seen even when selecting white pixels.

Figure 5: Webcam analysis.

doi: 10.1145/3343031.3350535. URL https://doi.org/10.1145/3343031.3350535.

[2] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2018.

[3] Tiffany CK Kwok, Eugene Yujun Fu, Erin You Wu, Michael Xuelin Huang, Grace Ngai, and Hong-Va Leong. Every little movement has a meaning of its own: Using past mouse movements to predict the next interaction. In *23rd International Conference on Intelligent User Interfaces*, pages 397–401. ACM, 2018.

[4] F Rohman, A Fauzan, et al. Integration of technology in project based learning with tracker on practicum activities. In *Journal of Physics: Conference Series*, volume 1185, page 012036. IOP Publishing, 2019.

[5] Meiju Shih and Li-Chieh Chen. An interactive and spherical panorama for multicultural city exploration. In *Proceedings of the 2nd International Conference on Education and Multimedia Technology*, pages 39–43. ACM, 2018.

[6] Manuel Stein, Halldor Janetzko, Daniel A Keim, and Tobias Schreck. Tackling similarity search for soccer match analysis: Multimodal distance measure and interactive query definition. *IEEE computer graphics and applications*, 2019.

[7] Yifan Wu, Remco Chang, Eugene Wu, and Joe Hellerstein. Programming with timespans in interactive visualizations. *arXiv preprint arXiv:1907.00075*, 2019.

[8] Chuang Yang, Yilan Zhang, Bo Tang, and Min Zhu. Vaite: A visualization-assisted interactive big urban trajectory data exploration system. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*, pages 2036–2039. IEEE, 2019.