GrabCut-python Guide book

GrabCut

The GrabCut [1] Algorithm is implemented using python (Please see the python file grabcut_func.py.) A GUI is built for interactive image segmentation using a python package named tkinter. (Please see the python file grabcut_GUI.py.)

a. A brief introduction of GrabCut Algorithm implemented in the code

GrabCut is built on its old version GraphCut [2]. The GraphCut Algorithm segments object from a given image by energy minimization. For every pixel z_i in image, there is an "opacity" value $\alpha_i \in \{0, 1\}$, with 0 for background and 1 for foreground. θ_f, θ_b describe image foreground and background distributions respectively, which are histograms of the two kinds of pixel values (foreground pixels or background pixels).

$$\theta = \{h(z;\ \alpha), \alpha = 0, 1\}$$

where $\int h(z;\ \alpha) = 1$.

GraphCut defines an energy function E as the loss function of segmentation task using "Gibbs" energy form. The object is to minimize E.

$$E(\alpha, \theta, z) = U(\alpha, \theta, z) + V(\alpha, z)$$

The data term U evaluates how well α and z are fit to each other with the given histogram θ (θ_f or θ_b).

$$U(\alpha, \theta, z) = \sum_i -logh(\alpha_i, z_i)$$

The smoothness term V evaluates the smoothness of the segmentation boundary.

$$V(\alpha, z) = \gamma \sum_{(i,j)\in img\ pixel} [\alpha_i \neq \alpha_j] B(i, j)$$

where $B(i, j)$ is high when the values of pixel i and pixel j are similar. Please refer to [1] for a detailed definition of $B(i, j)$.

So, the segmentation task can be described as an optimization problem shown below.

$$\hat{\alpha} = argmin\, E(\alpha, \theta, z)$$

A technique named Min Cut is used to find the global minimum.

GrabCut extends GraphCut mainly in two aspects. First, GrabCut replaces histogram with GMM (Gaussian Mixture Model) in the data term U. (GMM can be used for color space.) Second, the iterative user interaction in GrabCut helps the algorithm to achieve better performance.

There are two kinds of GMMs, one for background pixels and one for foreground pixels. Each GMM has K (typically K=5) components. So, the energy function E in GraphCut becomes:

$$E(\alpha, k, \theta, z) = U(\alpha, k, \theta, z) + V(\alpha, z)$$

where $k \in \{1, \dots K\}$.

The data term $U(\alpha, k, \theta, z)$ becomes:

$$D(\alpha_i, k_i, \theta_i, z_i) = -log\pi(\alpha_i, k_i) + \frac{1}{2} logdet\sum(\alpha_i, k_i) + \frac{1}{2}[z_i - \mu(\alpha_i, k_i)]^T \sum(\alpha_i, k_i)^{-1} [z_i - \mu(\alpha_i, k_i)]$$

$$U(\alpha, k, \theta, z) = \sum_i D(\alpha_i, k_i, \theta_i, z_i)$$

where $\mu(\alpha_i, k_i)$ means average values and $\sum(\alpha_i, k_i)$ means covariance matrix. Please refer to [1] for detailed information.

The smoothness term $V(\alpha, z)$ is the same as that one in GraphCut.

After initialization, the GrabCut Algorithm is carried out with the following steps:

i. Assign GMM components to pixels.

Apply k-means clustering for user hand-segmented foreground and background pixels respectively. The number of the clusters is K. For each pixel z_i in foreground and background respectively, assign k_i with the minimum $D(\alpha_i, k_i, \theta_i, z_i)$ value.

$$k_i = argmin\, D(\alpha_i, k_i, \theta_i, z_i)$$

ii. Learn GMM parameters from pixels z.

For foreground pixels assigned to k_i, calculate $\mu(\alpha_i, k_i)$ and $\sum(\alpha_i, k_i)$. Apply the same processing method to background pixels.

$$\theta = argmin\, U(\alpha, k, \theta, z)$$

iii. Construct graph and conduct Min Cut.

$$min\, E(\alpha, k, \theta, z)$$

iv. User interaction. Repeat the above steps.

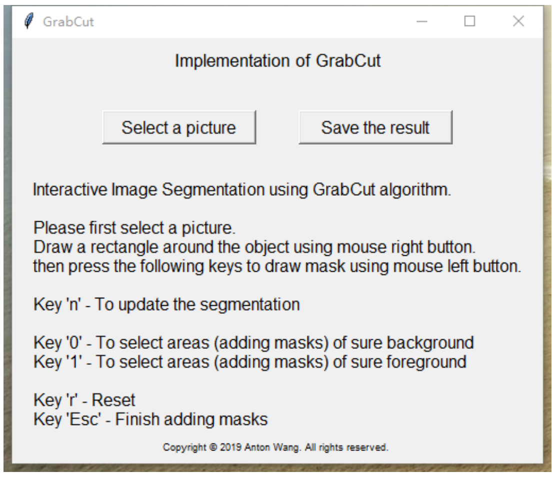
b. Experiment

System requirement: python 3.7.3, numpy, cv2, tkinter, igraph, sklearn

The codes were implemented with the reference of open-source codes on github and a python package named cv2.

How to Run the code

Run **.\GrabCut\grabcut_GUI.py** using PyCharm. Then you will see a GUI shown below.



Implementation of GrabCut

Interactive Image Segmentation using GrabCut algorithm.

Please first select a picture.
Draw a rectangle around the object using mouse right button.
then press the following keys to draw mask using mouse left button.

Key 'n' - To update the segmentation

Key '0' - To select areas (adding masks) of sure background
Key '1' - To select areas (adding masks) of sure foreground

Key 'r' - Reset
Key 'Esc' - Finish adding masks

Copyright © 2019 Anton Wang. All rights reserved.

Figure 1

Please read the instruction carefully.

First, push the button ‘Select a picture’ and select a picture. The sample pictures are stored in the file folder **.\GrabCut\test_pictures**. Here, for example, select **.\GrabCut\test_picture\dog1.jpg**. Then two windows will pop up. They show input and output images.

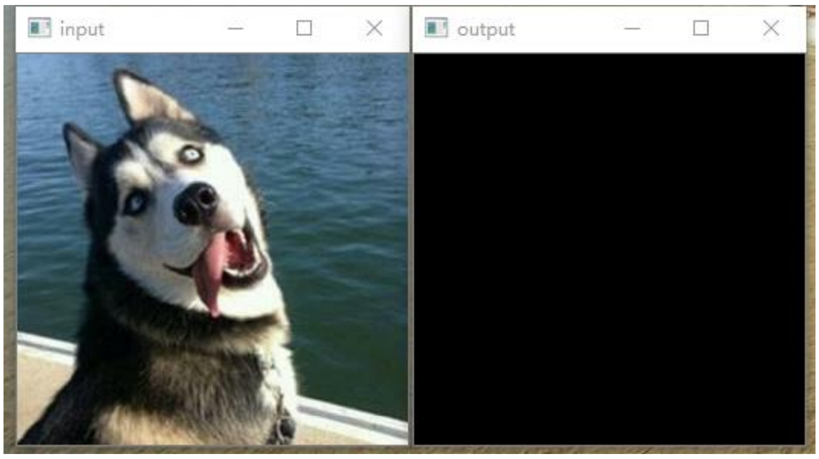
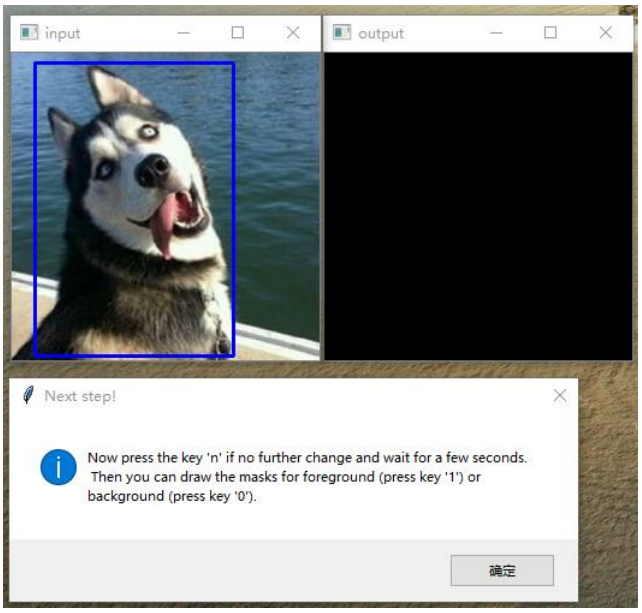


Figure 2

Draw a rectangle around the object using mouse right button. Then a message box will pop up.



Now press the key 'n' if no further change and wait for a few seconds.
Then you can draw the masks for foreground (press key '1') or background (press key '0').

Figure 3

Follow the instruction of the message box. Press key ‘n’, wait for a few seconds, and the initial result will be shown in window ‘output’.

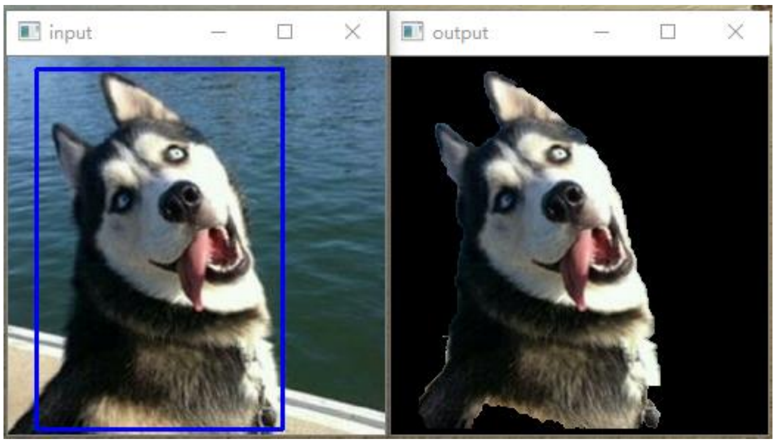


Figure 4

Further, press key ‘0’ to draw masks (black) for background or press key ‘1’ to draw masks (white) for foreground. Remember to press key ‘n’ when finishing drawing masks. And wait for a few seconds.

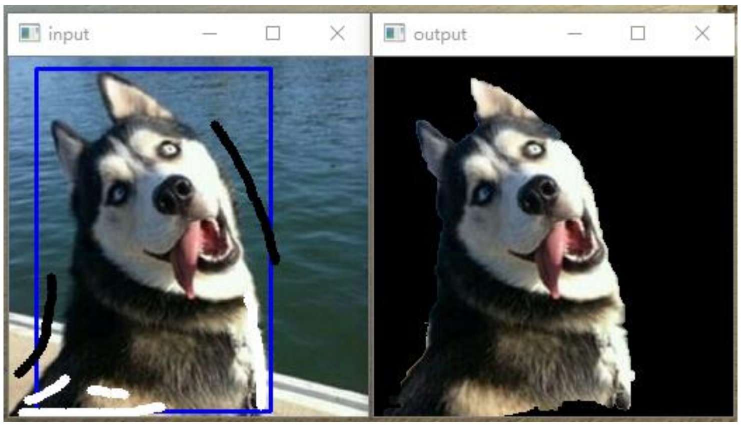


Figure 5

Please press key 'Esc' when finishing all the steps. The result can be saved as **.\GrabCut\grabcut_output.png** by pushing the 'Save the result' button on GUI. Pressing key 'r' will reset the segmentation process.

Please refer to the program notes of **.\GrabCut\grabcut_GUI.py** and **.\GrabCut\grabcut_func.py** for implementation details. **.\GrabCut\grabcut_GUI.py** builds the GUI and **.\GrabCut\grabcut_func.py** contains two classes, grabcut and GMM, which are implementation of GrabCut Algorithm.

Results of GrabCut

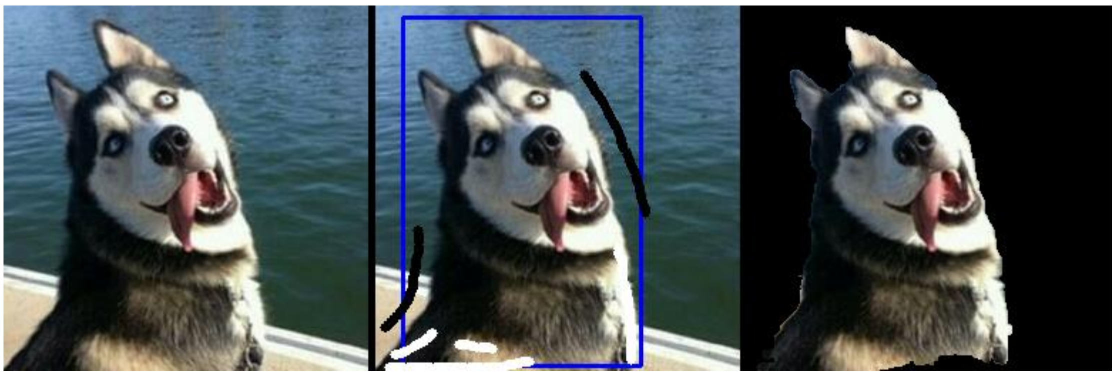


Figure 6 A

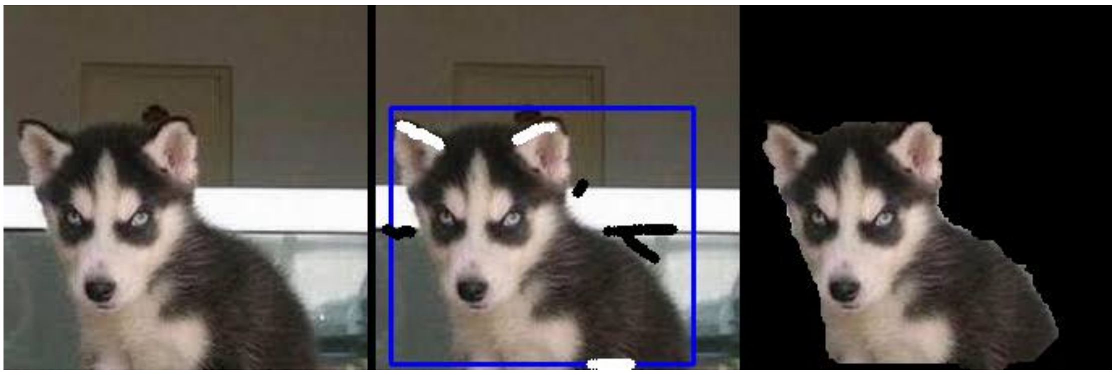


Figure 6 B

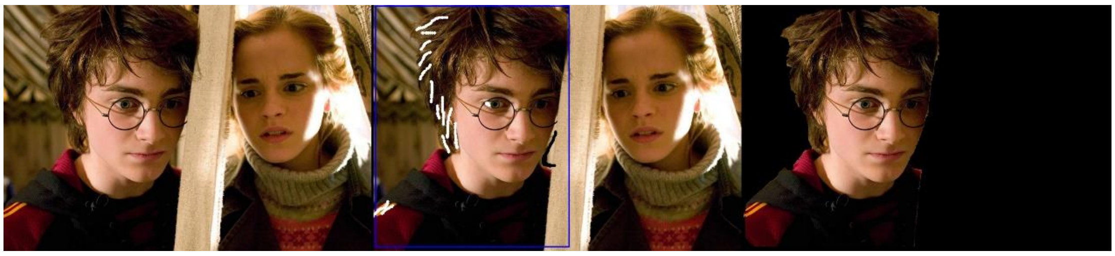


Figure 6 C

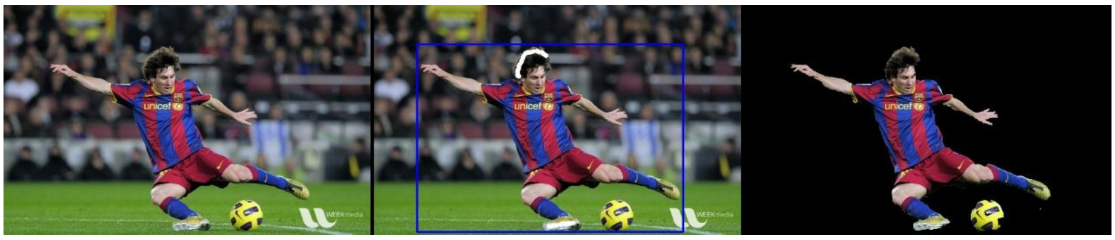


Figure 6 D

Figure 6 for each one (A-D), from the left to right is: original picture, drawn with rectangle and masks, and segmentation result.

Please find the above results in **.\GrabCut\test_picture_results**.

c. Reference

[1] Rother, C., Kolmogorov, V., & Blake, A. (2004, August). Grabcut: Interactive foreground extraction using iterated graph cuts. In ACM transactions on graphics (TOG) (Vol. 23, No. 3, pp. 309-314). ACM.

[2] Boykov, Y. Y., & Jolly, M. P. (2001, July). Interactive graph cuts for optimal boundary & region segmentation of objects in ND images. In Proceedings eighth IEEE international conference on computer vision. ICCV 2001 (Vol. 1, pp. 105-112). IEEE.