Microsoft Research Cambridge

SIGGRAPH2004

GrabCut

Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother
Vladimir Kolmogorov
Andrew Blake

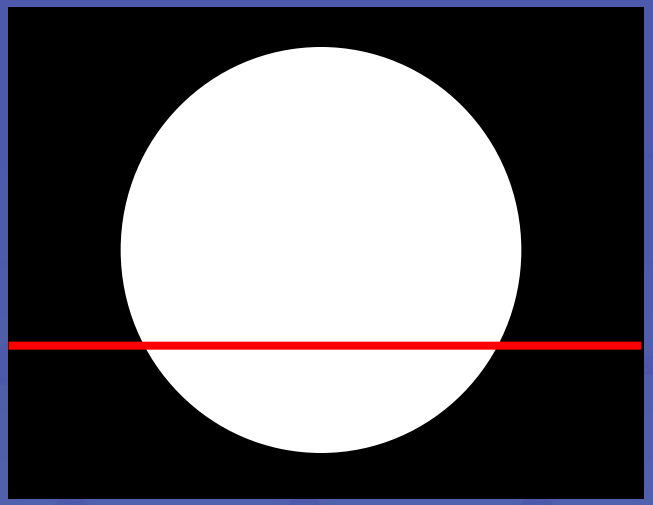
Microsoft Research Cambridge-UK

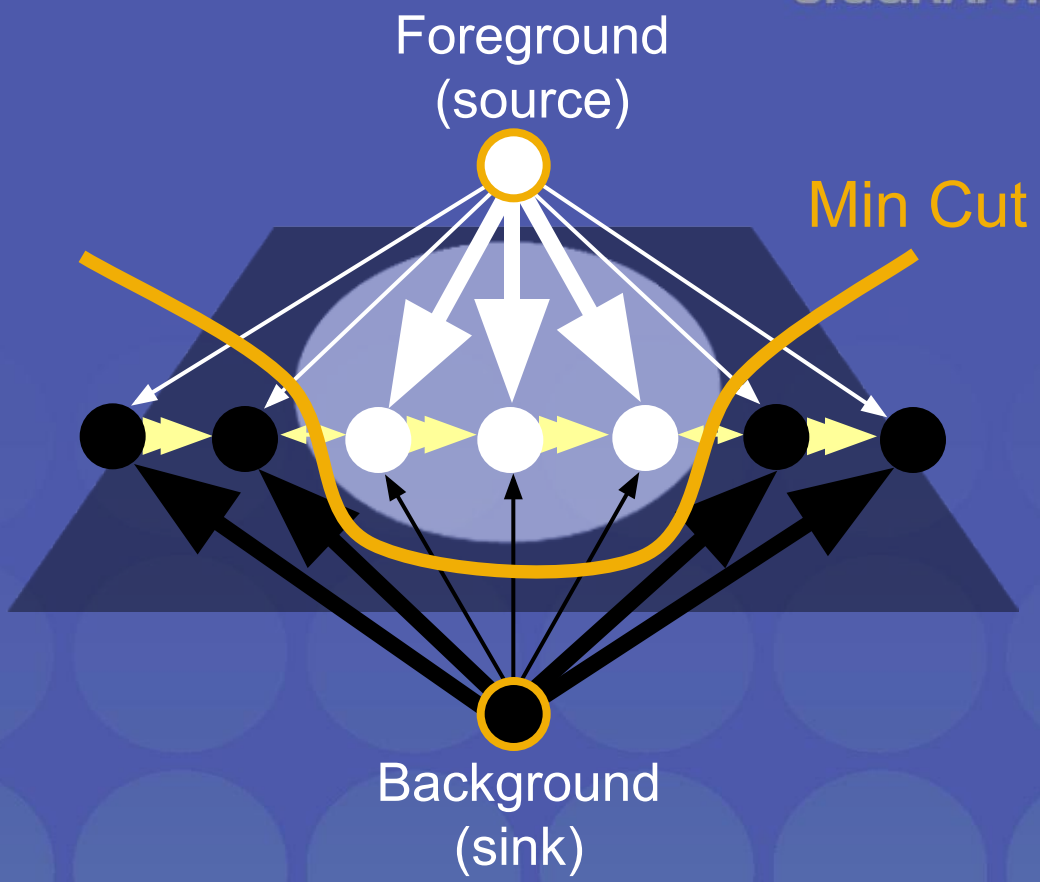
Problem

Fast &
Accurate ?

Graph Cuts modelling in images

Image





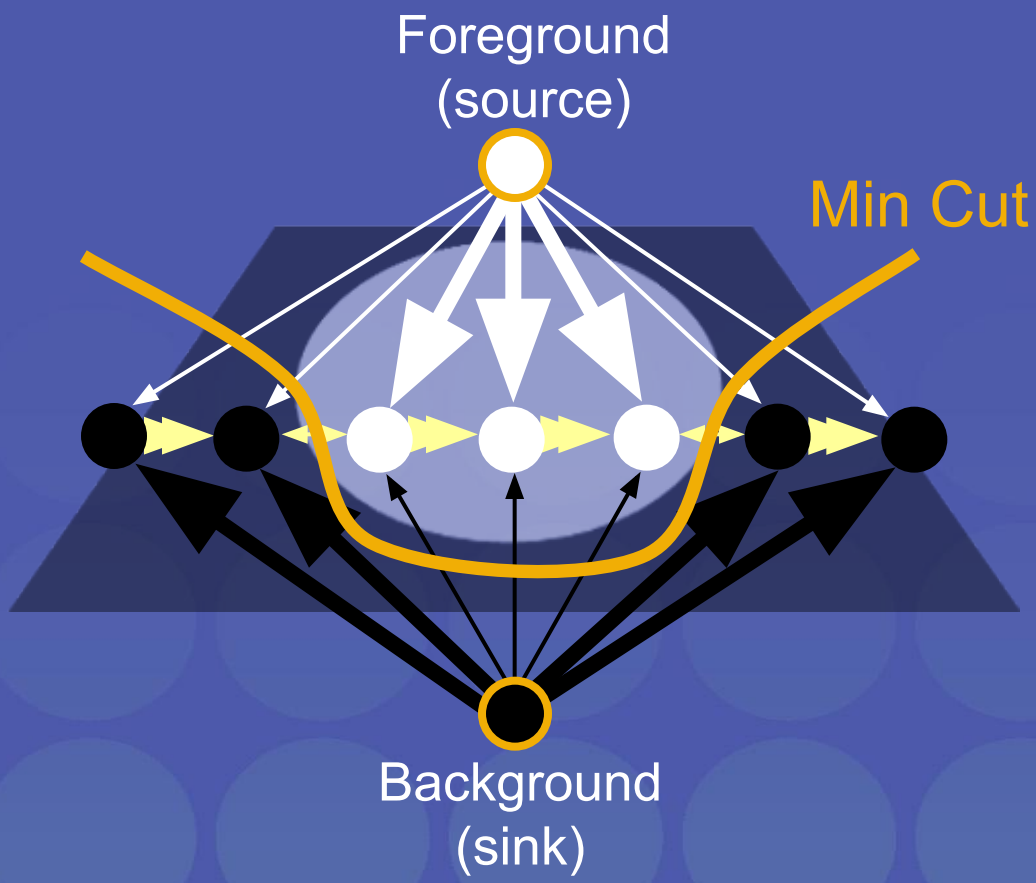
Cut: separating source and sink; Energy: collection of edges

Min Cut: Global minimal enegry in polynomial time

Graph Cuts for foreground extraction

SIGGRAPH2004

Assume we know foreground is **white** and background is **black**



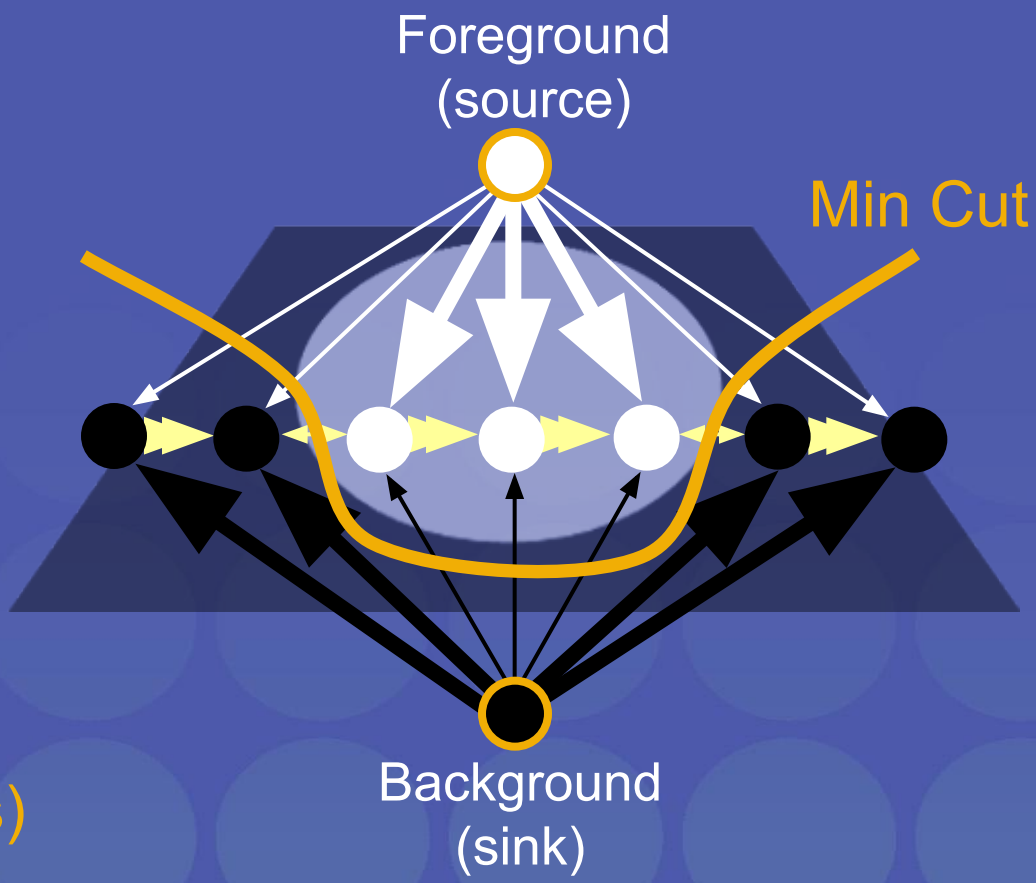
Graph Cuts for foreground extraction

SIGGRAPH2004

Assume we know foreground is **white** and background is **black**

Data term =
(cost of assigning label)

Regularization =
(cost of separating neighbors)



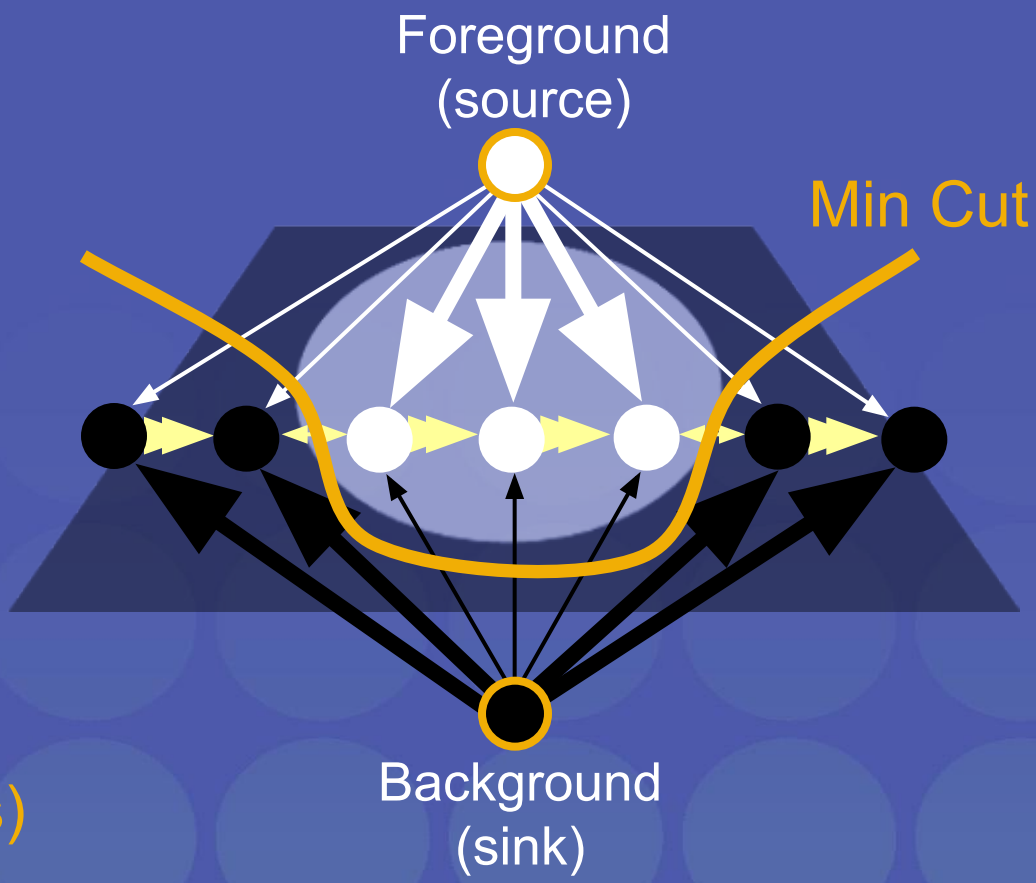
Graph Cuts for foreground extraction

SIGGRAPH2004

Assume we know foreground is **white** and background is **black**

Data term = **whiteness**
(cost of assigning label)

Regularization =
(cost of separating neighbors)

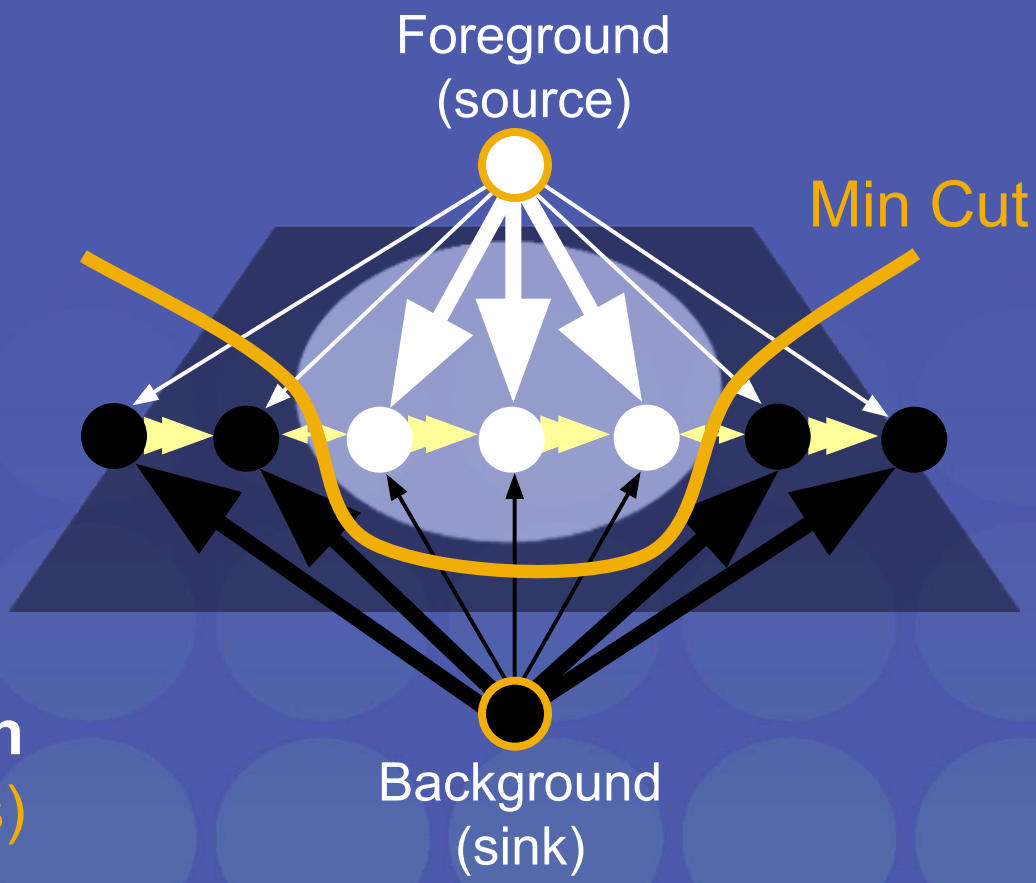


Graph Cuts for foreground extraction

Assume we know foreground is **white** and background is **black**

Data term = **whiteness**
(cost of assigning label)

Regularization = **color match**
(cost of separating neighbors)



We are all set now !

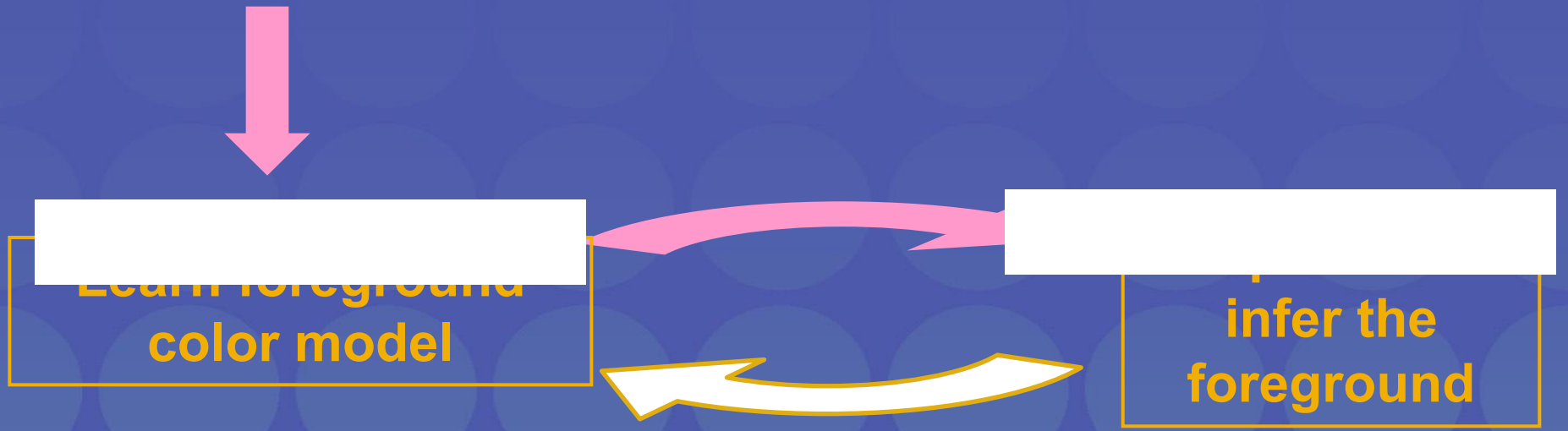
User Initialisation

color model

infer the foreground

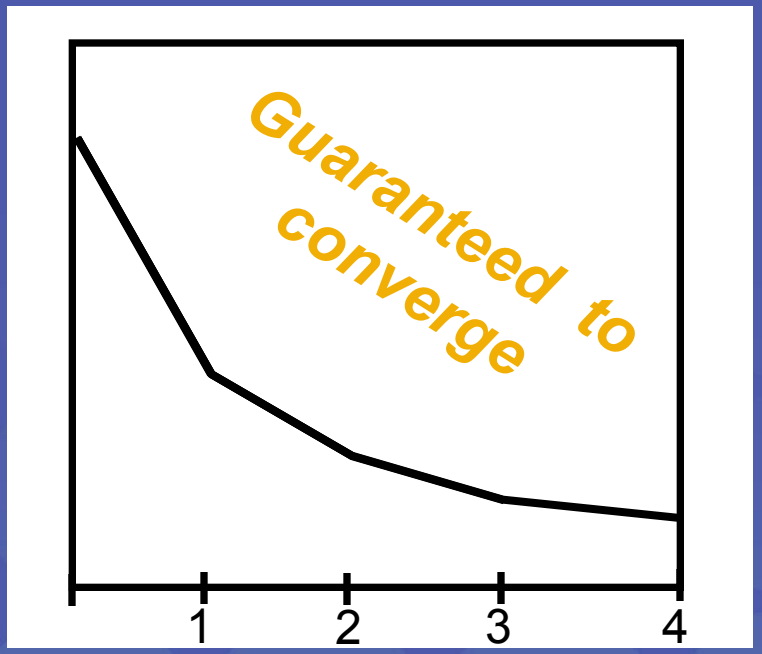
Iterated Graph Cuts

User Initialisation



Iterated Graph Cuts

Result



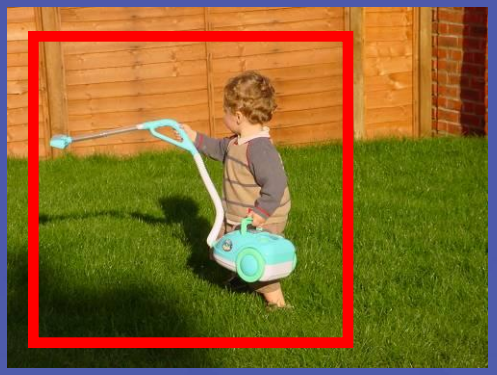
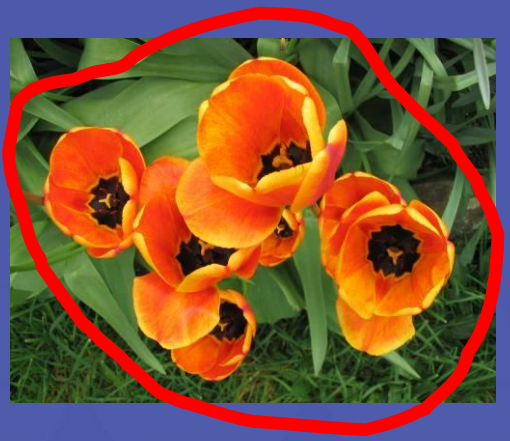
Energy after each Iteration

GrabCut algorithm

The GrabCut algorithm works by:

- Accepting an input image with *either* (1) a **bounding box** that specified the location of the object in the image we wanted to segment or (2) a **mask** that *approximated* the segmentation
- Iteratively performing the following steps:
  - **Step #1:** Estimating the color distribution of the foreground and background via a Gaussian Mixture Model (GMM)
  - **Step #2:** Constructing a Markov random field over the pixels labels (i.e., foreground vs. background)
  - **Step #3:** Applying a graph cut optimization to arrive at the final segmentation

Moderately straightforward examples



… GrabCut completes automatically

Difficult Examples

SIGGRAPH2004

Camouflage & Low Contrast

Fine structure

No telepathy

Initial Rectangle

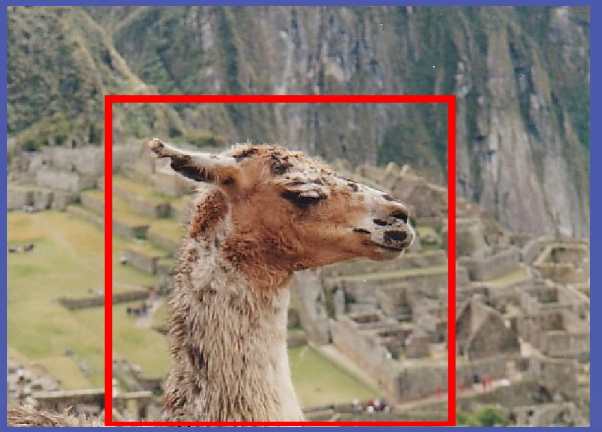
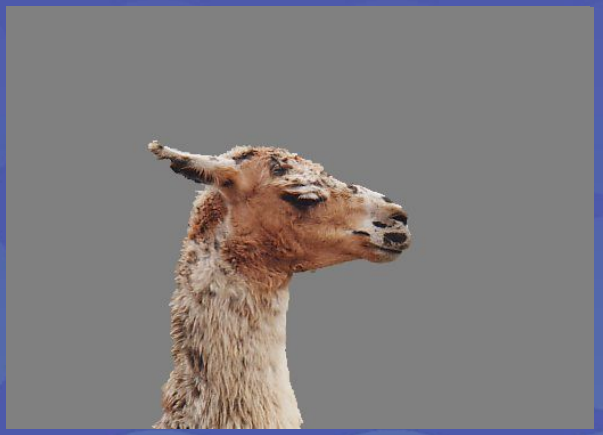
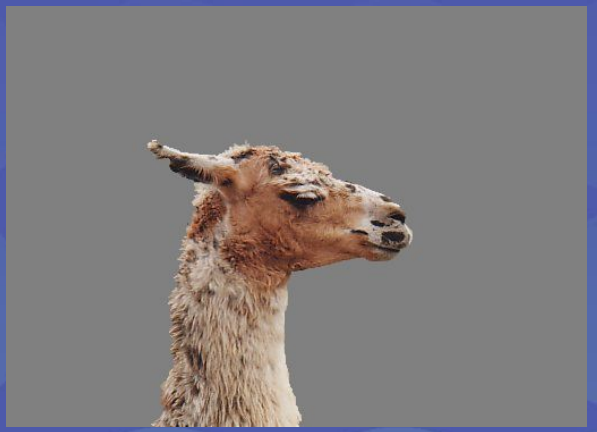
Initial Result

Evaluation – Labelled Database

Available online: http://research.microsoft.com/vision/cambridge/segmentation/

Comparison

SIGGRAPH2004

	Boykov and Jolly (2001)	GrabCut
User Input		
Result		
	Error Rate: 0.72%	Error Rate: 0.72%

Summary









Magic Wand (198?)

Intelligent Scissors Mortensen and Barrett (1995)

Graph Cuts Boykov and Jolly (2001)

LazySnapping Li et al. (2004)

GrabCut Rother et al. (2004)

cv2.grabCut()

- Syntax:

  cv2.grabCut(image, mask, rectangle, backgroundModel, foregroundModel, iterationCount[, mode])
- Parameters:
- image: Input 8-bit 3-channel image.
- mask: Input/output 8-bit single-channel mask. The mask is initialized by the function when mode is set to GC_INIT_WITH_RECT. Its elements may have one of following values:
  - GC_BGD defines an obvious background pixels.
  - GC_FGD defines an obvious foreground (object) pixel.
  - GC_PR_BGD defines a possible background pixel.
  - GC_PR_FGD defines a possible foreground pixel.

cv2.grabCut()

- Parameters:
- rectangle: It is the region of interest containing a segmented object. The pixels outside of the ROI are marked as obvious background. The parameter is only used when mode==GC_INIT_WITH_RECT.
- backgroundModel: Temporary array for the background model.
- foregroundModel: Temporary array for the foreground model.
- iterationCount: Number of iterations the algorithm should make before returning the result. Note that the result can be refined with further calls with mode==GC_INIT_WITH_MASK or mode==GC_EVAL.
- mode: It defines the Operation mode. It can be one of the following:
  - GC_INIT_WITH_RECT: The function initializes the state and the mask using the provided rectangle. After that it runs iterCount iterations of the algorithm.
  - GC_INIT_WITH_MASK: The function initializes the state using the provided mask. Note that GC_INIT_WITH_RECT and GC_INIT_WITH_MASK can be combined. Then, all the pixels outside of the ROI are automatically initialized with GC_BGD.
  - GC_EVAL: The value means that the algorithm should just resume.


```python
# Python program to illustrate
# foreground extraction using
# GrabCut algorithm

# organize imports
import numpy as np
import cv2
from matplotlib import pyplot as plt

# path to input image specified and
# image is loaded with imread command
image = cv2.imread('image.jpg')

# create a simple mask image similar
# to the loaded image, with the
# shape and return type
mask = np.zeros(image.shape[:2], np.uint8)

# specify the background and foreground model
# using numpy the array is constructed of 1 row
# and 65 columns, and all array elements are 0
# Data type for the array is np.float64 (default)
backgroundModel = np.zeros((1, 65), np.float64)
foregroundModel = np.zeros((1, 65), np.float64)
```

https://www.geeksforgeeks.org/python-foreground-extraction-in-an-image-using-grabcut-algorithm/

```
# define the Region of Interest (ROI)
# as the coordinates of the rectangle
# where the values are entered as
# (startingPoint_x, startingPoint_y, width, height)
# these coordinates are according to the input image
# it may vary for different images
rectangle = (20, 100, 150, 150)

# apply the grabcut algorithm with appropriate
# values as parameters, number of iterations = 3
# cv2.GC_INIT_WITH_RECT is used because
# of the rectangle mode is used
cv2.grabCut(image, mask, rectangle,
            backgroundModel, foregroundModel,
            3, cv2.GC_INIT_WITH_RECT)
```

https://www.geeksforgeeks.org/python-foreground-extraction-in-an-image-using-grabcut-algorithm/

```
# In the new mask image, pixels will
# be marked with four flags
# four flags denote the background / foreground
# mask is changed, all the 0 and 2 pixels
# are converted to the background
# mask is changed, all the 1 and 3 pixels
# are now the part of the foreground
# the return type is also mentioned,
# this gives us the final mask
mask2 = np.where((mask == 2)|(mask == 0), 0, 1).astype('uint8')

# The final mask is multiplied with
# the input image to give the segmented image.
image = image * mask2[:, :, np.newaxis]

# output segmented image with colorbar
plt.imshow(image)
plt.colorbar()
plt.show()
```

https://www.geeksforgeeks.org/python-foreground-extraction-in-an-image-using-grabcut-algorithm/

Tham khảo

- https://www.pyimagesearch.com/2020/07/27/opencv-grabcut-foreground-segmentation-and-extraction/
- https://docs.opencv.org/3.4/d8/d83/tutorial_py_grabcut.html
- https://cvg.ethz.ch/teaching/cvl/2012/grabcut-siggraph04.pdf

"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother* Vladimir Kolmogorov† Andrew Blake‡
Microsoft Research Cambridge, UK

Figure 1: **Three examples of GrabCut.** The user drags a rectangle loosely around an object. The object is then extracted automatically.

```
def GraphSeg(path_filename, x, y, w, h):
  img = cv2.imread(path_filename)
  img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
  plt.imshow(img)
  plt.show()
  print(img.shape)

  mask = np.zeros(img.shape[:2], np.uint8)
  backgroundModel = np.zeros((1, 65), np.float64)
  foregroundModel = np.zeros((1, 65), np.float64)
  rectangle = (x, y, w, h)
  cv2.grabCut(img, mask, rectangle,
              backgroundModel, foregroundModel,
              5, cv2.GC_INIT_WITH_RECT)

  mask2 = np.where((mask == 2) | (mask == 0), 0, 1).astype('uint8')
  res = img * mask2[:, :, np.newaxis]
  plt.imshow(res)
  plt.show()
  res = cv2.cvtColor(res, cv2.COLOR_BGR2RGB)
```