Introduction to Image Processing and Computer Vision

Laboratory Project 1

Plant Segmentation and Labeling

Łukasz Niedziałek
niedzialekl@student.mini.pw.edu.pl

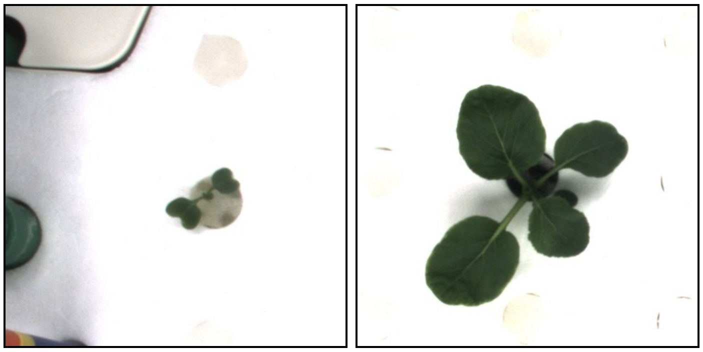
Contents

1 Problem description

In computer vision, image segmentation is the process of partitioning a digital image into multiple segments (sets of pixels, also known as image objects). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze.[1]

Our goal was to segment plants from background using photos from KOMATSUNA dataset[2]. Those images are pretty good when it comes to image processing, due to good unification and simplification of theirs backgrounds. We have been given two tasks:

1. generate plant mask for each image (optionally draw boundary box on original image)
2. create fully automated tool separating each leaf from others.



[1] Source: https://en.wikipedia.org/wiki/Image_segmentation
[2] Dataset's website: http://limu.ait.kyushu-u.ac.jp/~agri/komatsuna/

2 Solutions

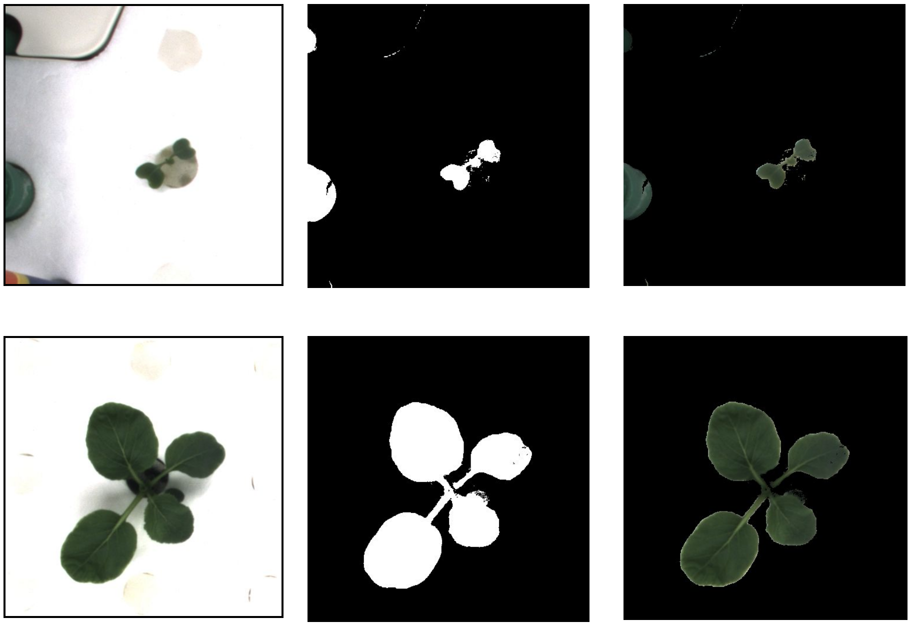
2.1 Plant mask

Firstly we have to read the images.

```
self.img = cv2.imread(path)
```

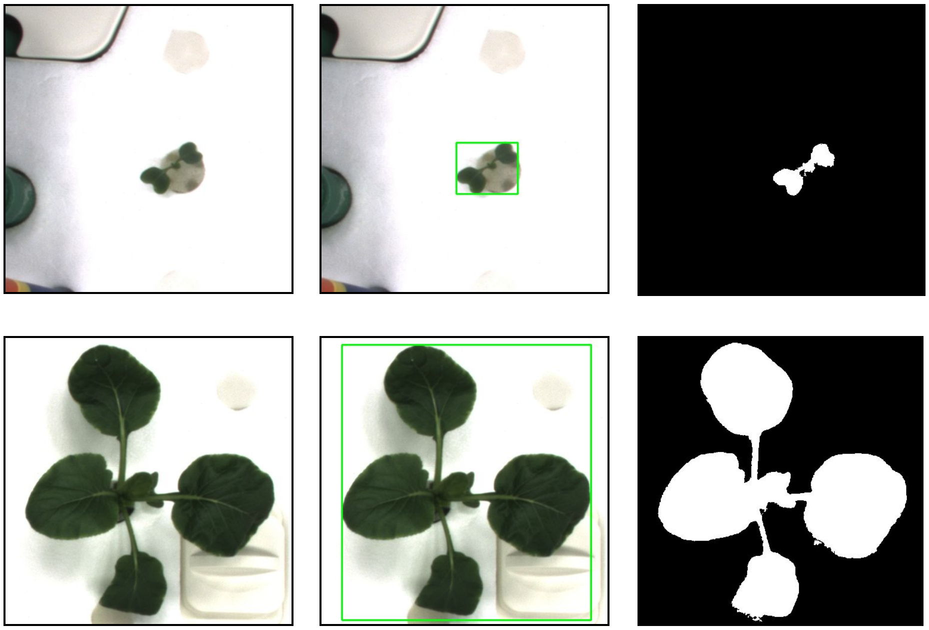
I have observed that most of the images have plain white (bright) background, and decided to create a simple green color mask to separate those plants. After few tries I have came up with satisfying values of the mask.

```
self.hsv = cv2.cvtColor(self.img, cv2.COLOR_BGR2HSV)

__low_green = np.array([32, 40, 35])
__high_green = np.array([90, 255, 255])

green_mask = cv2.inRange(self.hsv, self.__low_green,
self.__high_green)

img_with_green_mask = cv2.bitwise_and(self.img, self.img,
mask=self.get_green_mask())
```



Then I realized that the noise objects are either much smaller than the plant itself or they touch boundaries of the images, so I have generated contours, removed objects that touch boundaries (+/- 5 pixels) and then I kept only the biggest contour.

```
conts, hierarchy = cv2.findContours(self.get_green_mask(), cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)

proper_conts = []

for cont in self.get_contours():
   x, y, w, h = cv2.boundingRect(cont)

   if x < 5 or (x + w) > self.img_w:
       continue

   if y < 5 or (y + h) > self.img_h:
       continue

   proper_conts.append(cont)

proper_conts = self.get_proper_contours()
biggest_area = 0
biggest_proper_contour = proper_conts[0]

for cont in self.get_contours():
   x, y, w, h = cv2.boundingRect(cont)
   area = w * h

   if area > biggest_area:
       biggest_area = area
       biggest_proper_contour = cont

contour_mask = np.zeros(self.img.shape[:2], np.uint8)

cv2.drawContours(contour_mask, [self.get_biggest_proper_contour()], -1, (255, 255,
255), -1)
```



Conclusions

This algorithm has produced images with mean accuracy of 98.98 % with the worst score being 88.27 %. All of the errors are created by the green mask and not perfectly distributed light over the image. The worst cases occur when there is a gap in the middle of a plant mask which causes a leaf cut.

2.2 Leaves segmentation

I have started by searching for some help on how to do such automated leaves segmentation. I haven't found any idea on how to solve this problem (labeled images from dataset where labeled manually), but I thought that maybe finding a center of a mask mass, interpreting it as a center of a plant and dividing leaves from that point will be a good idea.

I had to work on images divided by a specific plant and camera, in order to keep that center at the same position and be able to rely on information from previous iterations (I worked on images in chronological order). In order to find a center of mass I have found a contour of a plant and calculated its center (this is done only for first images of each plant-camera group).

```
conts, hierarchy = cv2.findContours(img_black_white, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)

M = cv2.moments(conts[0])
cX = int(M["m10"] / M["m00"])
cY = int(M["m01"] / M["m00"])
self.center = (cX, cY)
```

Then I had to divide this contour, so I have created a circle, with a radius of 7 pixels, in the middle of it. This divides that contour into small partitions (it results in error, but it is really small error compared to the scale of image) which can be interpreted as separate leaves. Then I could color those leaves.

```
conts2, hierarchy = cv2.findContours(img_black_white, cv2.RETR_LIST,
cv2.CHAIN_APPROX_SIMPLE)

cv2.circle(img, self.center, 7, self.__colors[0], -1)

for i in range(0, len(conts2)):
   cv2.drawContours(img, [conts2[i]], -1, self.__colors[i], -1)
```

Results

I have not developed this further, because first results have shown that it is clearly wrong way of doing it. The main problems were: expansion of the plants in the middle and also the fact that there is no particular shape that can separate leaves in all of those cases (which can be easily chosen).