UNIVERSITÀ
DI TRENTO

Computer Vision applied to sprout detection: Detection of sprouts and flowers

Department of Information Engineering and Computer Science

Project of Signal, Image and Video

Group: Bortolin Samuel, Grassi Alessandro

Introduction

- Detect/Isolate sprouts and flowers from images
- 88 images from Robinia
- Used libraries: Copy, **NumPy**, **OpenCV** and Typing
- Structure of the project:

```
sprout-detection
├── (images)            [ignored folder where you can add your images and then change the "image.extension" in the files with the name of your images]
└── src
    ├── image_utils               [package that contains the utils for images]
    │   └── standard_image_operations              [library which contains the main operations on images]
    ├── color_picker              [script to pick HSV colors and define ranges]
    ├── hsv_filter                [script to try our standard HSV filter on different kind of images]
    └── main                      [script to extract the relevant edges of the image using canny and sobel, also using an HSV filtering approach]
```

Developed library

image_utils package containing **standard_image_operations**.py where there is our library: **StandardImageOperations**.

We implemented there all the main operations on images that we needed in order to perform the detection.

- rescale_image(image_to_rescale, target_number_of_pixels)
- grab_contours(contour_tuple)
- find_canny_best_threshold(greyscale_image)
- canny_edges(greyscale_image, low_threshold)
- canny_elements(greyscale_image, low_threshold)
- canny_on_image(greyscale_image, low_threshold, original_image)
- sobel_edges(greyscale_image)
- remove_percentile(greyscale_image)
- apply_hsv_mask(original_image, hsv_image, lower_bound, upper_bound)
- get_hsv_mask(original_image, hsv_image, color)

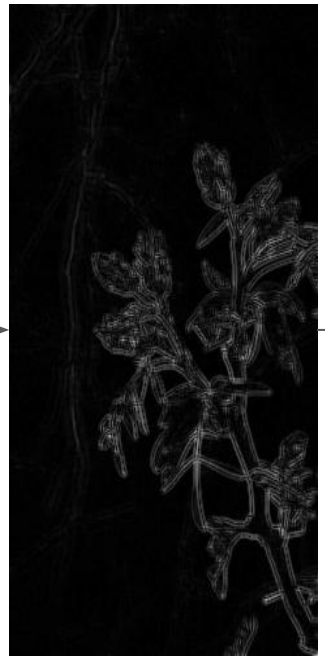
Developed scripts

- **color_picker**.py: in the first phase it allows us to pick pixels from an image and then clicking “q” it returns the range of HSV values that contains all the selected pixels; in the second phase it allows us to build an HSV filter using trackbars to modify the value of hue, saturation and value, clicking a button it updates the filter on the image and clicking “q” it returns the range of HSV values selected with the trackbars. This has been particularly useful to pick the right HSV values for the detection of flowers, and in particular for the removal of the sky in building our standard HSV filter.
- **hsv_filter**.py: script that performs an isolation of the color for different types of images containing flowers, leaves or branches using our standard HSV filter.
- **main**.py: script that extracts the relevant edges of the image using canny and sobel. It also tries an HSV color filtering approach and extracts the relevant edges of the HSV filtered image.

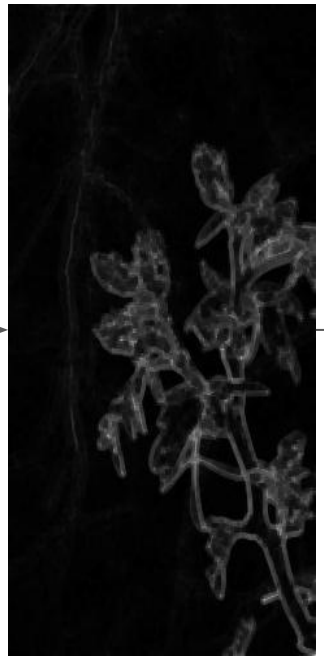
Sobel Edges

original

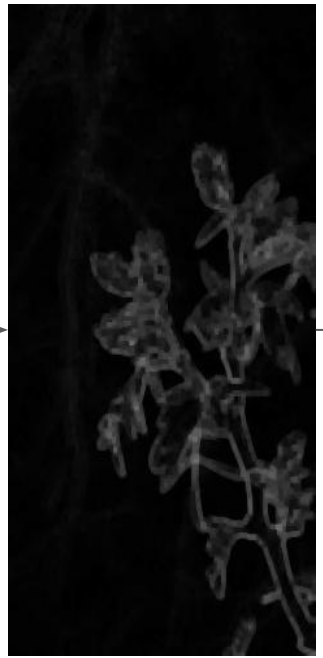
Sobel derivative



close



open



percentile removed

find_canny_best_threshold

```python
def find_canny_best_threshold(greyscale_image: np.ndarray) -> int:
    """
    Find the best threshold for canny edge detector
    """

    canny_on_previous_threshold = StandardImageOperations.canny_elements(greyscale_image, 25)
    canny_previous_delta = StandardImageOperations.canny_elements(greyscale_image, 24) - canny_on_previous_threshold
    picked_threshold = 0
    for threshold_value in range(26, 1000):
        canny_on_new_threshold = StandardImageOperations.canny_elements(greyscale_image, threshold_value)
        canny_new_delta = canny_on_previous_threshold - canny_on_new_threshold
        canny_on_previous_threshold = canny_on_new_threshold
        if (canny_new_delta + canny_previous_delta) < 50:
            picked_threshold = threshold_value
            break
        canny_previous_delta = canny_new_delta
    return picked_threshold
```

Bitwise_and Sobel edges with Canny edges

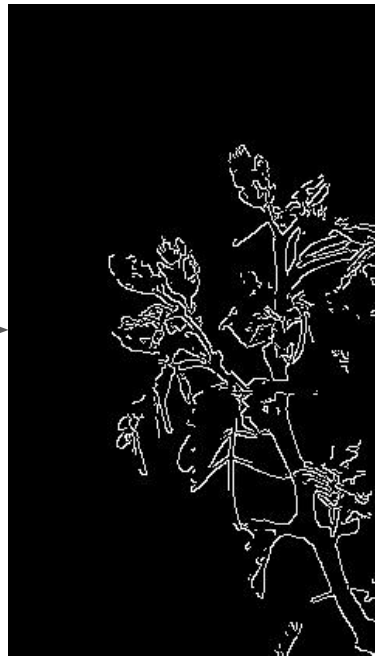
original

Canny edges

percentile removed

X

final edges



Final result expansion

final edges

edges expanded

Results

original leaves image

Sobel-Canny edges approach applied on a leaves image

Results

contours area approach on a leaves image

Sobel-Canny edges on a leaves image

Results

original branch image

Sobel-Canny edges approach applied on a branch image

Results

contours area approach on a branch image

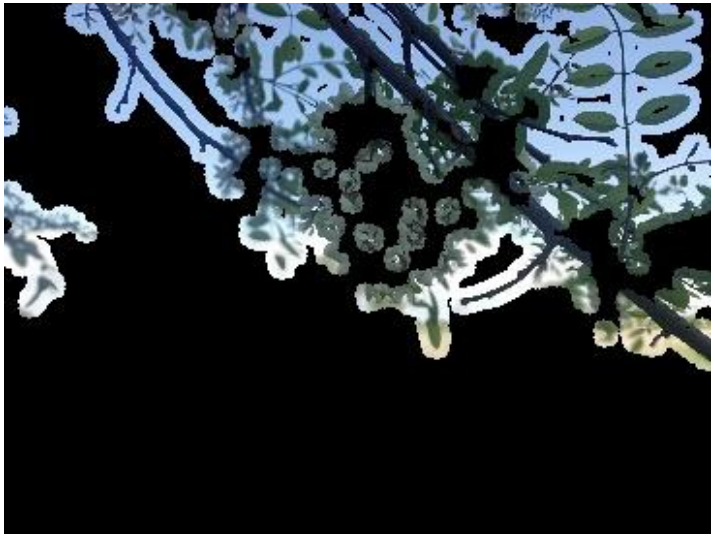
Sobel-Canny edges on a branch image

Results

original flowers image

Sobel-Canny edges approach applied on a flowers image

Results



contours area approach on a flowers image

Sobel-Canny edges on a flowers image

get_hsv_mask

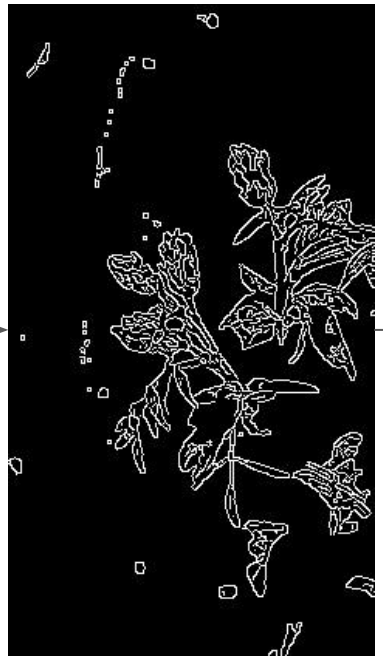
```python
def get_hsv_mask(original_image: np.ndarray, hsv_image: np.ndarray, color: str) -> np.ndarray:
    """
    Apply an hsv mask based on a color of interest on the original image
    """

    if color == "f":
        lower_bound = np.array([0, 0, 50])
        upper_bound = np.array([35, 255, 255])
        lower_image = StandardImageOperations.apply_hsv_mask(original_image, hsv_image, lower_bound, upper_bound)
        lower_bound = np.array([75, 0, 50])
        upper_bound = np.array([90, 255, 255])
        upper_image = StandardImageOperations.apply_hsv_mask(original_image, hsv_image, lower_bound, upper_bound)
        lower_image = cv.bitwise_or(lower_image, upper_image)
        lower_bound = np.array([110, 0, 50])
        upper_bound = np.array([179, 255, 255])
        upper_image = StandardImageOperations.apply_hsv_mask(original_image, hsv_image, lower_bound, upper_bound)
        return cv.bitwise_or(lower_image, upper_image)
    elif color == "l":
        lower_bound = np.array([20, 0, 0])
        upper_bound = np.array([80, 255, 255])
    elif color == "b":
        lower_bound = np.array([0, 0, 0])
        upper_bound = np.array([30, 255, 255])
    else:
        lower_bound = np.array([0, 0, 0])
        upper_bound = np.array([180, 255, 255])
    return StandardImageOperations.apply_hsv_mask(original_image, hsv_image, lower_bound, upper_bound)
```

Bitwise_and HSV-filtering edges with Sobel-Canny edges

HSV-filtered

HSV-filtering edges

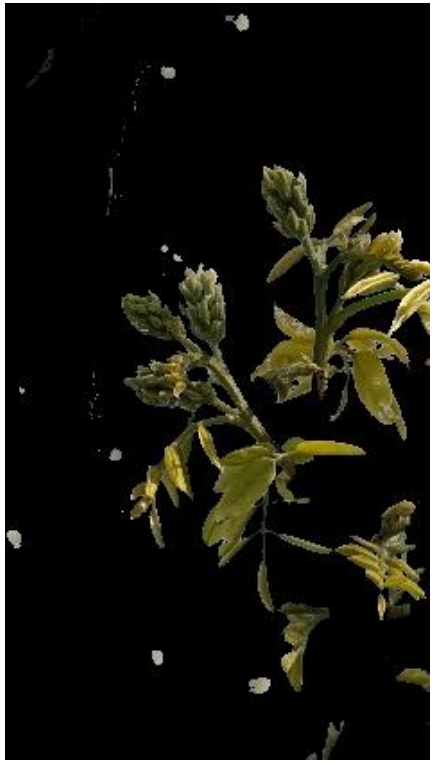


Sobel-Canny edges

X

final edges

Results



hsv-filtered leaves image

hsv-filtering edges + Sobel-Canny edges approach applied on a leaves image

Results

hsv-filtering edges + Sobel-Canny edges approach on the hsv-filtered leaves image

hsv-filtering edges + Sobel-Canny edges approach on a leaves image

Results

hsv-filtered branch image

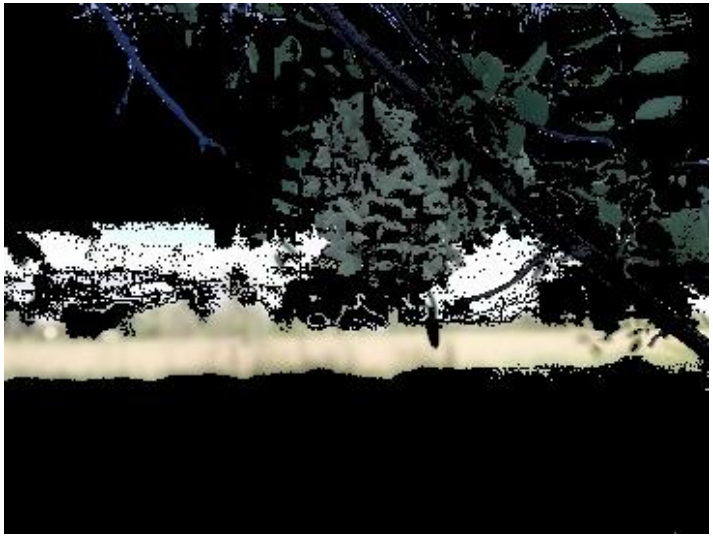
hsv-filtering edges + Sobel-Canny edges approach applied on a branch image

Results

hsv-filtering edges + Sobel-Canny edges approach on the hsv-filtered branch image

hsv-filtering edges + Sobel-Canny edges approach on a branch image

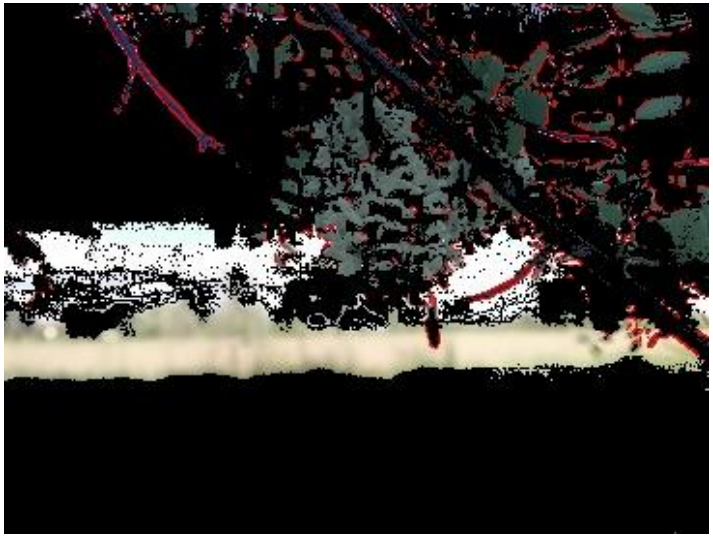
Results



hsv-filtered flowers image

hsv-filtering edges + Sobel-Canny edges approach applied on a flowers image

Results



hsv-filtering edges + Sobel-Canny edges approach on the hsv-filtered flowers image

hsv-filtering edges + Sobel-Canny edges approach on a flowers image

Results

original image

hsv-filtering edges + Sobel-Canny edges approach on image

Results

original image

hsv-filtering edges + Sobel-Canny edges approach on image

Results

original image

hsv-filtered image

Results

original image

Sobel-Canny edges on image

Results

original image

Sobel-Canny edges on image

Results

original image

contours area approach on image

Results

original image

Sobel-Canny edges on image

hsv-filtering edges + Sobel-Canny edges approach on the hsv-filtered image

Results

original image

Sobel-Canny edges on image

hsv-filtered image

Conclusions

- From these image results we can see that even if the images are not optimal for this task, we detected in an effective way the main subject of the image.
- In a lot of images the main subject is not on focus or in the foreground and our edge detection approach does not work so well. Instead, when the object of interest is in the foreground and clearly identifiable our edge detection approach works very well.
- A note on the hsv approach is that it is not valid in general for all kinds of images, this is due to different subjects. In fact our main code requires at the beginning to specify what do you want to analyze and type it.
- When specified the subject for a specific image, the HSV approach works also well and isolates the subject. Sometimes there is an issue with that approach since most of the pictures are in a countryside setting. In such a context the background of the images could be of the same color of the object and not filtered out.