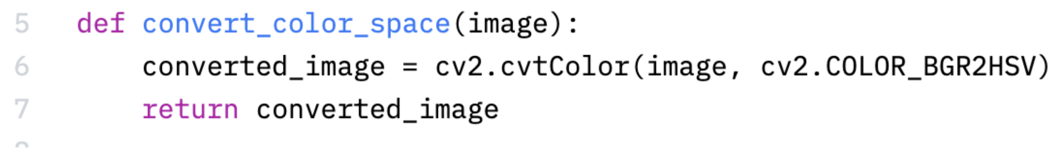
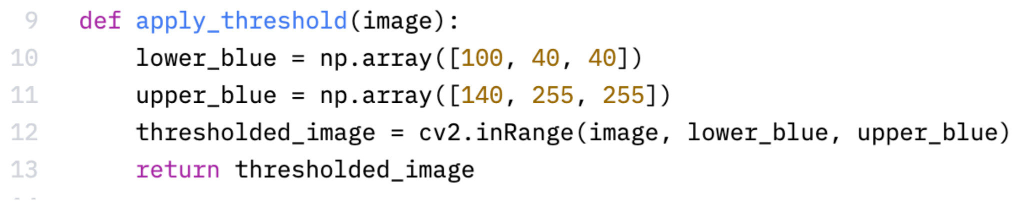
CS 5330 – Lab1
Documentation & Reflection
Spring 2024
Yixing Chen

In lab1 this project, the objective was to develop a computer vision application to identify sky pixels in images using traditional image processing techniques. The techniques I chose for this purpose were color space conversion, color thresholding, and morphological operations. In the beginning, I also chose the edge detection, but after testing, the result was not effective, then I deleted this function. These methods were chosen because of their simplicity and ease of use, their ability to effectively process images based on color features. And their ability to refine the results through structural modifications.
I: Color Space Conversion (RGB to HSV): In the first step I changed the image from the BGR color space to HSV. This change is important because HSV separates the color from the brightness, making it easier to pick out specific colors. I picked HSV particularly for how well it can highlight the blues and whites that often make up the sky.



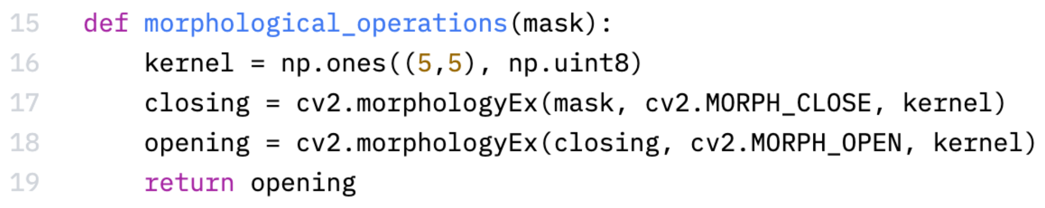
```
def convert_color_space(image):
    converted_image = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
    return converted_image
```

II: Color Thresholding: After the Color Space Conversion, I used Color Thresholds to pinpoint sky pixels. The sky, typically being blue to light blue and can be effectively segmented by defining a range of hue values in HSV space corresponding to these colors.



```
def apply_threshold(image):
    lower_blue = np.array([100, 40, 40])
    upper_blue = np.array([140, 255, 255])
    thresholded_image = cv2.inRange(image, lower_blue, upper_blue)
    return thresholded_image
```

III: Morphological Operations: To refine the segmentation and reduce noise, morphological operations such as closing and opening were utilized. This step helps in closing small holes within detected sky regions and removing small artifacts, respectively, resulting in a more coherent detection of the sky region.

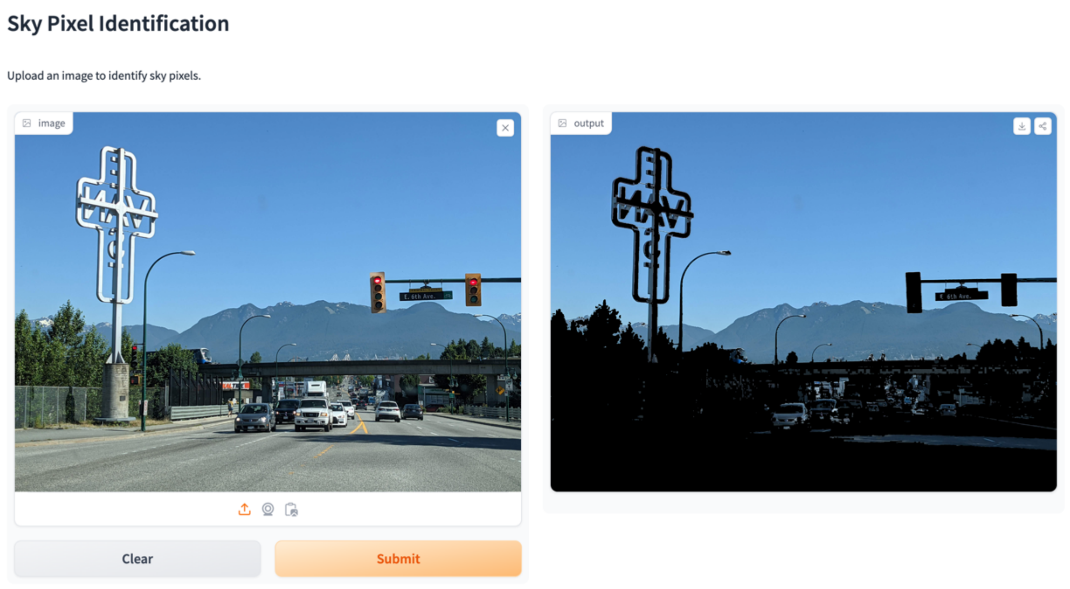


```
def morphological_operations(mask):
    kernel = np.ones((5,5), np.uint8)
    closing = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
    opening = cv2.morphologyEx(closing, cv2.MORPH_OPEN, kernel)
    return opening
```

Challenges and Solutions:
One of the main challenges was the variability of sky appearances under different lighting conditions and times of the day. I had to play around with the range of colors to make sure I could still find the sky in a bunch of different pictures. Another challenge involved the presence of noise and small non-sky regions being detected as part of the sky. Then Morphological operations was applied to fix it.

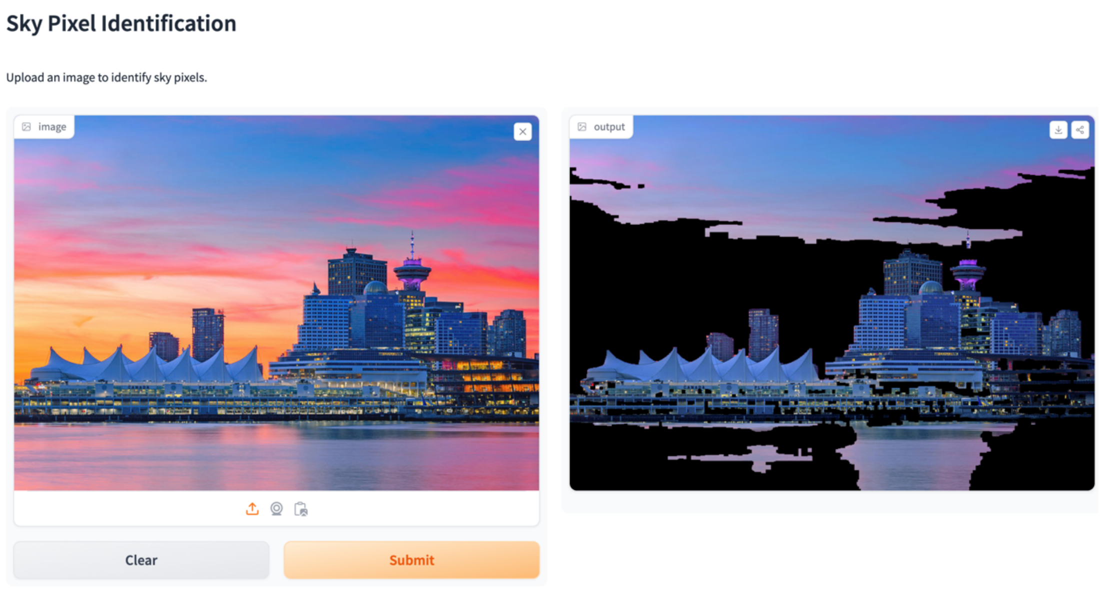
Reflection

Looking back on this project, some pictures can identify sky pixels effectively.


Sky Pixel Identification

Upload an image to identify sky pixels.

However, it also had its limitations. The biggest one was how much the results depended on the specific HSV values I chose, which might not work for every single picture because of how varied the sky can look. And sometimes it mistakenly identified things that were similar in color to the sky, like water or certain buildings, or at the sunset and sunrise.


Sky Pixel Identification

Upload an image to identify sky pixels.

Even though it still has some issues, the project was a great learning experience. It showed me how challenging image processing can be and taught me the value of picking and finetuning my methods based on what I am trying to achieve.

I think in the future, I could make sky identification better by using machine learning to adjust the HSV thresholds based on the image's context.

Overall, this assignment really deepened my understanding of image processing, such as color space conversion, thresholding techniques and morphological operation. I also learned how to use the Gradio library to create interactive applications.