Sky Region Detection for Day and Night Images with Computer Vision with Python and OpenCV

Bryan Chang Fa Kan

Department of Computing and Information Systems

Sunway University

Selangor, Malaysia

bryancfk@gmail.com

Abstract— **This paper presents a computer vision-based system to identify pixels belonging to the sky region. First, we obtain the gradient information of the image. Then, we obtain the optimal sky-ground border based on energy function optimization. Finally, a post-processing method is applied to refine the sky region detection result. To evaluate the accuracy of the segmentation, the ground truth masks are compared with the predicted sky-ground region masks. The skyline is also identified for the results for visualization. Moreover, the program classifies the images as either taken during the daytime or nighttime based on average brightness of images. The implementation of the program uses Python and OpenCV based on SkyFinder datasets (684, 10917, 623, 9708). Experimental results demonstrated the effectiveness and reliability of the proposed approach in accurately segmenting the sky region from a diverse set of outdoor images with average accuracy value of 90.99% and accurately labelling daytime and nighttime with average accuracy value of 88.75%.**

I. INTRODUCTION

Sky region detection is a fundamental task in computer vision, aimed at automatically identifying and segmenting the sky area from outdoor images and video frames. The problem involves accurately distinguishing the sky region, which holds crucial information about the surrounding environment, from other elements in the scene. The primary objective of sky region detection is to obtain a precise boundary between the sky and non-sky regions in images. This boundary is essential for various applications, including image interpretation, object recognition, weather forecasting, autonomous navigation, and content-based image manipulation. Accurate sky region detection is challenging due to factors such as varying lighting conditions, weather, complex outdoor scenes or even due to bad quality of the image.

The accurate detection and segmentation of the sky region in outdoor images hold paramount importance in various real-world applications involving computer vision and image processing. The sky serves as a vital subject matter that conveys essential information about the outdoor environment and its surroundings. Consequently, precise identification and extraction of the sky region are crucial for subsequent processing tasks, such as classification, object recognition, and content-based image manipulation.

The knowledge about the sky condition enables the extraction of valuable features related to weather and illumination conditions. This information can be harnessed to enhance various applications and aid in constraining the search domain for more complex image processing and recognition tasks. For example, in the context of obstacle detection or path planning systems for autonomous or unmanned vehicles, accurate sky region detection provides valuable context for decision-making and situational awareness. Considering the wide array of applications that benefit from precise sky region detection, there is also a growing demand for improved algorithms with enhanced speed, robustness, and applicability to diverse images.

Furthermore, achieving higher accuracy in sky segmentation ensures more reliable results, making it an active area of research and development in the field of computer vision and image processing.

In this paper, we present a computer vision-based system aimed at automatically detecting and segmenting the sky region from outdoor images. The proposed approach utilizes gradient information and energy function optimization to find the optimal sky-ground border, followed by post-processing techniques to refine the detection result. Through experimental evaluation on the SkyFinder datasets consisting of images from different scenarios, we demonstrate the accuracy of our approach in accurately segmenting the sky region and labelling images as daytime or nighttime thus contributing to the advancement of various outdoor-based computer vision applications.

A. Differences Between the Proposed Algorithm and Existing Algorithms

One noticeable difference of our proposed algorithm with any other algorithms is the applicability and robustness in handling different sky conditions. One of the problems of existing sky segmentation systems is they often encounter challenges when dealing with dark sky images. These algorithms typically employ simple and low-complexity techniques, primarily focusing on identifying the horizon line using edge detection methods and then distinguishing between the sky and non-sky regions. However, such approaches may not deliver accurate results in various scenarios, especially when the sky region exhibits low contrast or complex lighting conditions. Due to these limitations, there is a need for more advanced and robust sky segmentation algorithms that can handle a wide range of image variations and lighting conditions. Our proposed approach aims to address these shortcomings by enabling the processing of dark or bright sky images with good accuracy. Furthermore, existing sky segmentation does not automatically label whether an image is taken during daytime or nighttime. In our system, it will have the capability to differentiate and label the images to be daytime or nighttime with good accuracy. This advancement in sky segmentation techniques can significantly enhance the performance and applicability of outdoor image processing and computer vision applications, providing more comprehensive and reliable insights into the outdoor environment. Furthermore, rather than just displaying the results of a final mask, the proposed system aims to display a final mask and a skyline image to better visualize the segmentation of sky and ground region.

II. LITERATURE REVIEW

In this literature review section, we will provide an overview and summary of the related systems, methods, and algorithms for sky detection and segmentation that inspired the design and methodology of our proposed approach.

Shen and Wang [1] introduced a sky region detection algorithm within a single image based on gradient information and energy function optimization. Unlike most of the existing methods, this algorithm has the capability to process either colour or greyscale images. In pre-processing, input image will be converted to greyscale image and the gradient image is obtained with Sobel operator. Then, the gradient magnitude image is applied in energy function optimization to obtain the optimal sky border position. The modified energy function in the proposed algorithm was inspired by the work in [2]. Lastly, they applied refinement and post-processing method to remove fake sky regions in some images. The experimental results demonstrated the robustness and relatively low computational cost of the algorithm compared to existing methods. However, the algorithm has limitations in handling complex images or dark sky conditions, and it is primarily suitable for robot navigation tasks where the sky region has smooth borders and is clearly distinguishable.

The Sky Region Detection based on Border Points was introduced in [3]. It can handle complex sky regions with buildings and flags as well as different conditions such as cloudy sky. It overcomes the limitations of single border line algorithms such as the one in [1] by using colour and gradient information to identify border points and detect the sky region. A border point is known as the pixel which separates sky and non-sky region in each column of image. This algorithm is especially useful for processing images that contains multiple sky regions which are separated by buildings. However, it may also mistakenly identify objects as sky region especially white objects as it is wrongly perceived as sky.

Laungrungthip et al. [4] proposed an edge-based detection method for identifying sky regions in images for solar exposure prediction. By utilizing Canny edge detection and morphological closing, it successfully identifies sky boundaries, enabling precise segmentation of the sky area for solar exposure analysis. The algorithm achieves promising performance in accurately detecting sky regions in diverse images, even in challenging scenarios such as complex backgrounds or varying lighting conditions. However, it can only work well if an appropriate threshold for canny edge detection was set.

Zafarifar and Weda [5] proposed a hybrid algorithm combining edge-based and colour-based horizon detection techniques. It takes advantage of their complementary strengths to improve the robustness of horizon detection. The combined system is able to compensate for the lack of accuracy of colour-based-detector with the high precision edge-based detector. Similarly, the vulnerability of the edge-based detector to non-horizon straight lines or its potential failure in the absence of a distinct horizon line is mitigated by the incorporated initial estimate provide by colour-based detector.

The hybrid probability model algorithm proposed by Wang et al. [6] is another form of hybrid algorithm that combines two types of algorithms to find the sky-ground boundary vector and label pixels belonging to the sky. The formula used to obtain sky ground boundary vector was inspired by the formula proposed by [2]. Then, the colour center and standard deviations obtained were used for the probability models which will generate the final probability map for sky region detection.

III. PROPOSED METHOD

The proposed system is designed to take in 1 input from the user, which is the sample size. The sample size determines the number of sample images to be taken from each of the dataset from SkyFinder datasets. The SkyFinder datasets to be use in the proposed system are datasets 684, 10917, 623 and 9708. Once the input is entered, the system will perform image processing which will be discussed in the following sections. The output of the program includes the output images and the output in the console. For the output images, they are stored in the Results folder according to their dataset folder. For the output in the console, it produces the accuracy of sky-ground segmentation and daytime or nighttime labelling of each sample images. At the end of the output, it will also display average accuracy of sample images in each dataset and the total average accuracy of all the datasets.

1. Creating necessary folders for output images

The program will first create a result directory that will store all the program's outputs. In the result directory, it holds the folders of the dataset. Each of the dataset folders will hold their respective output of sample images. These outputs include the original image of the sample image, the final mask produced after the sky-ground segmentation and the skyline image.

2. Preprocessing

For preprocessing, the program will load a sample of images from the SkyFinder datasets. In this case, datasets 684, 10917, 623 and 9708 were used to obtain the samples. These sample images are then converted to grayscale to simplify further processing. Then, Canny ddge detection is used to detect the edges. As there is no theoretical basis for setting upper threshold and lower threshold for Canny edge detector, trial and error was performed by manually adjusting the thresholds and evaluating the results obtained from the Canny edge detector. It was experimentally decided that the upper threshold and lower threshold will work relatively well with the value of 255 for upper threshold and the mean intensity value of the image * 0.55 for lower threshold. After obtaining the edges, we perform Sobel operator to obtain the gradient magnitude image. It is performed by convolving the horizontal and vertical operator and get two gradient images. The 2 gradient images are combined to obtain the gradient magnitude image.

3. Optimal Sky Border Position, OptimalSkyBorderPos()

After that, the gradient magnitude image and input image are used to compute the optimal sky border position. Firstly, the algorithm seeks an appropriate threshold value that maximizes the energy optimization function. This process involves three parameters: the minimum threshold, maximum thresholds, and the search step value. Based on Shen and Wang [1], the experiment conducted by them have shown that when the threshold values exceeding 600, it will fetch the value of energy function that remains nearly constant. Therefore, 600 was chosen for the maximum threshold value and similarly the minimum threshold of five was chosen as that is where the graph began when extrapolated. Furthermore, a search step of 5 was chosen to obtain a balance between search precision and computational complexity. The program then determines the number of sample points to search based on the values of the three parameters. These three parameters will also be used for computing the optimal threshold value, t. The for loop iterates through a range of threshold values, calculating the energy function for each threshold. It keeps track of the threshold value that gives the highest energy function value, representing the optimal sky border position. The loop calculates the threshold values based on the minimum, maximum threshold, and the search step value provided as input to the function. The goal is to find the threshold that maximizes the energy optimization function, which corresponds to the best estimation of the sky border position for the input image.

4. Calculate Border Position, CalculateBorder()

The CalculateBorder function takes in the gradient image and a threshold value t as inputs. It iterates through each column of the gradient image. For each column, it finds the maximum vertical position where the gradient value is greater than the threshold t. This position represents the border between the sky and non-sky regions in that column. The function creates a 1D NumPy array to store these border positions for each column. If a valid border position is found, it is stored in the corresponding entry of the sky array. After processing all columns, the function returns the sky array, which represents the sky border position function for the entire image.

5. Energy Function, EnergyFunction()

The EnergyFunction function calculates the energy value based on the input image and a given sky border position (bTmp) obtained from CalculateBorder() function. It first creates a binary mask (skyMask) to separate the sky and ground regions in the image using the estimated border position. Then, the function calculates the energy value (J_n) according to the formula proposed by Shen and Wang [1]:

$$J_n = \frac{1}{\gamma \cdot \left|\boldsymbol{\Sigma}_s\right| + \left|\boldsymbol{\Sigma}_g\right| + \gamma \cdot \left|\lambda_1^s\right| + \left|\lambda_1^g\right|}$$

where $\gamma = 2$ is determined experimentally.

The function extracts the pixel values of the sky and ground regions from the original image based on the skyMask. It then computes the covariance matrices of sky and ground region, Σs and Σg and their respective average RGB values. np.linalg.det and np.linalg.eig were used to obtain the determinants, $|\cdot|$ and Eigen values, λ_i^s and λ_i^g for the sky and ground regions to be used for the energy optimisation function respectively. Using the values, the function calculates the energy value (J_n) according to the formula above. The energy value reflects the effectiveness of the current sky border position in distinguishing between the sky and non-sky regions.

Finally, the EnergyFunction returns the calculated energy value (J_n) as the output. This value which will be processed by OptimalSkyBorderPos is crucial in the process of finding the optimal sky border position that maximizes the energy optimization function for the given image.

In the OptimalSkyBorderPos function, the most optimal energy value is determined based on all the energy value (J_n) computed. At the end of the function, the function will return the optimal sky border position that maximizes the energy optimization function by fetching the highest energy value (J_n).

6. Create Mask, CreateMask()

After getting all the optimal sky border position, the subsequent step in the program involves creating a binary mask that distinguishes the sky region (labelled with pixel value 1) as the region of interest and the non-sky region (labelled with pixel value 0). The binary mask will display the ground and the ground objects as black, while the sky regions will appear white. This function utilizes the list of optimal border points and the input image to generate this binary mask. Initially, it creates a blank binary mask with the same size as the input image. Then, it iterates through the list of border points, assigning an intensity value of 255 to the corresponding x and y coordinates and all the pixels below it in the binary mask. Consequently, the ground regions in the initial mask are marked with a value of 255, while the sky regions are marked with 0. To make the sky region the region of interest (pixel value 255) and the non-sky region as 0, the mask is inverted using the cv2.bitwise_not(mask) function. Finally, the function returns the generated mask to the calling function.

7. Postprocessing and Producing Final Mask and Skyline

The algorithm can be directly applied to the test or dataset images. However, it was observed that there are gaps within the ground or non-sky region of the generated mask. To address this issue and make the mask more representative of the actual image's ground region, a post-processing step is applied. The post-processing function fills in the gaps by performing morphological closing on the inverted mask using a 20x20 square kernel. Experimental results indicate that this kernel size yields the most connected ground region and achieves the highest segmentation accuracy.

After post-processing, the final mask will be saved in the Results directory under its respective dataset's folder. The saved mask is named after the initial image, with "_mask" appended, and it is stored in PNG format.

The results can also be visualised using skyline. Based on the post-processed mask, skyline can be extracted by finding the contours in the mask. Then, create a new blank mask that will be used to draw the contours. Finally, Draw the contours of the skyline on the new mask. The skyline will be saved in the Results directory under its respective dataset's folder. The saved skyline is named after the initial image, with "_skyline" appended, and it is stored in PNG format.

8. Labelling Daytime and Nighttime

To identify whether an image is taken during daytime or nighttime, the average brightness of a sample image is compared against the average brightness of all the images in the dataset. To calculate the average brightness, the RGB images are converted to HSV. The V channel of HSV is observed since it represents the brightness or intensity of the color. The calculation of average brightness is done by obtaining the sum of brightness of all pixels in an image divided by the area of the image.

9. Measuring the accuracy of sky-ground segmentation and the labelling of daytime and nighttime

To obtain the accuracy value for sky-ground segmentation, the final mask is compared with its corresponding ground truth mask obtained from Skyfinder dataset which serves as a benchmark against the final mask. To measure the accuracy, the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values are calculated. These values represent the number of correctly predicted sky pixels, correctly predicted non-sky pixels, incorrectly predicted sky pixels as non-sky, and incorrectly predicted non-sky pixels as sky, respectively. It then proceeds to calculate the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values. These values represent the number of correctly predicted sky pixels, correctly predicted non-sky pixels, incorrectly predicted sky pixels as non-sky, and incorrectly predicted non-sky pixels as sky, respectively. Finally, the accuracy of the segmentation is computed as the ratio of the sum of true positive and true negative values to the total number of pixels in the image. This provides a measure of how well the post-processed mask matches the ground truth mask and indicates the quality of the segmentation algorithm's performance. The accuracy value ranges from 0 (completely inaccurate) to 1 (perfect segmentation). The result is multiplied by 100 to give a percentage value between 0% to 100% in 2 decimal places.

To obtain the accuracy value of daytime and nighttime labels, the output label is compared with its corresponding ground truth label. Since the ground truth labels for daytime and nighttime are not given in the SkyFinder dataset, we will be manually labelling the images by observing the images. The accuracy is obtained by taking a set of images and determine how many of those images are labelled correctly.

IV. RESULTS AND DISCUSSIONS

In order to analyse the results quantitatively, the program is tested with Skyfinder datasets (684, 10917, 623, 9708) to obtain the average accuracy for sample images for each of the dataset. The total average accuracy based on all the sample images of the datasets will also be obtained. These accuracy values measure how often the program is able to segment these regions in images correctly and accurately.

Moreover, the accuracy to identify daytime and nighttime is also measured. To obtain the correct daytime and nighttime labels, we manually label 20 images from each of the datasets which serves as the ground truth for the predicted labels.

Table 1

Table below shows the average accuracy values taken from 20 sample images from each dataset when compared to the ground truth mask for segmenting sky-ground regions. The total average displays the average of all the average accuracy values.

Dataset	Accuracy
684	88.95%
10917	89.89%
623	93.67%
9708	91.44%
Total Average	90.99%

Table 2

Table below shows the average accuracy values taken from 20 sample images from each dataset for labelling daytime and nighttime. The total average displays the average of all the average accuracy values.

Dataset	Accuracy
684	95%
10917	100%

623	100%
9708	60%
Total Average	88.75%

The goal is to achieve an average of 80-100% for the accuracy value for segmenting sky-ground region and accuracy value for labelling daytime and nighttime. The results have shown that the goal has been fulfilled with 90.99% and 88.75% respectively.

REFERENCES

[1] Y. Shen and Q. Wang, “Sky Region Detection in a Single Image for Autonomous Ground Robot Navigation,” *International Journal of Advanced Robotic Systems*, Oct. 2013, doi: 10.5772/56884.

[2] S. M. Ettinger, M. C. Nechyba, P. G. Ifju, and M. Waszak, “Vision-guided flight stability and control for Micro Air Vehicles,” *IEEE/RSJ International Conference on Intelligent Robots and System*. doi:10.1109/irds.2002.1041582

[3] Z. Zhijie, W. Qian, S. Huadong, J. Xuesong, T. Qin, and S. Xiaoying, "A novel sky region detection algorithm based on border points," *Int. J. Signal Process., Image Process. Pattern Recognit.*, vol. 8, no. 3, pp. 281-290, 2015.

[4] N. Laungrungthip, A. E. McKinnon, C. D. Churcher, and K. Unsworth, "Edge-based detection of sky regions in images for solar exposure prediction," in *2008 23rd International Conference Image and Vision Computing New Zealand*, pp. 1-6, Nov. 2008.

[5] B. Zafarifar and H. Weda, "Horizon detection based on sky-color and edge features," in *Visual Communications and Image Processing 2008*, vol. 6822, pp. 683-691, Jan. 2008.

[6] C. W. Wang, J. J. Ding, and P. J. Chen, "An efficient sky detection algorithm based on hybrid probability model," in *2015 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA)*, pp. 919-922, Dec. 2015.