Automatic Sky Segmentation and Harmonious Substitution in Video Processing

Nguyen-Khoa Bui-Dinh
Faculty of Information Technology
VNUHCM - University of Science
Ho Chi Minh city, Viet Nam
19120018@student.hcmus.edu.vn

Quang Le-Nhat
Faculty of Information Technology
VNUHCM - University of Science
Ho Chi Minh city, Viet Nam
19120340@student.hcmus.edu.vn

Do Vu-Van
Faculty of Information Technology
VNUHCM - University of Science
Ho Chi Minh city, Viet Nam
19120007@student.hcmus.edu.vn

***Abstract*—Automatic sky replacement in video has been a new research topic in Computer Vision in recent years. More previous approaches were applied for static images, videos that require the use of inertial measurement units in smartphone and there have been some research methods for this topic recently. To contribute to the next step toward this topic, a solution is proposed for automatic sky substitution with three steps: sky segmentation, motion tracking estimation and background combination with image. Our methods use K-Nearest Neighbors for refine and inpaint annotations, SegNet for detecting sky regions, calculate the motion of the sky and blend skymask with image through some formulas in image processing. Our experiments which are applied for outdoor videos captured by smartphones or handheld cameras produce good abstractions in video quality even in lighting and weather conditions.**

***Index Terms*—Sky replacement, SegNet architecture, K-Nearest Neighbors.**

I. Introduction

There is a meaningful quote of anonymous artist about the beauty of sky: "If eyes are windows to the soul in a portrait, sky is the window to the soul in a landscape". In spite of the fact that its role is not appreciated and even unclear in the mind of photographers, it can create a considerable result in an image overall. The quality of the image can upset photographers because of undesirable weather conditions as well as lighting factors. A beautiful landscape but cloudy weather does not show remarkable feature of the beauty of landscape. Thanks to the development of computer vision, sky adjustment in photos becomes a main research topic [1]–[5] and is integrated into popular image-editing softwares like Photoshop, PicsArt and Befunky. These apps allow users to swap skies with custom images and effects. Photoshop now supports Quick Selection Tool and Sky Replacement Tool to swap sky for their custom image.

In addition, there is an upward trend in creating attractive short-form videos in many social media platforms like TikTok, Instagram, Facebook or YouTube. To produce high-quality short-form videos, content creators do not only have innovative ideas but also know how to use video-editing softwares. Most video clips contain sky - the most common but forgettable component overall. Sky can be substituted with unrealistic visual effects like galaxy, Internet memes and movie scenes. Despite the support of video editing softwares like Adobe Premiere Pro and Adobe After Effects, sky replacement is comprised of many complicated steps and manually adjusted frame-by-frame without an automatic solution of sky replacement. These tasks are time-consuming, laborious and require professional skills in video production.

In this paper, user experience should be improved by proposing a solution of sky replacement automatically without any user interactions. Previous method of sky substitution is only applied mostly in static photos [3], [5], [6] or videos which require gyroscope sensor and Internet connection [7]. This paper is going to propose a new solution for sky-containing videos with one-click.

Video sky replacement is comprised of three steps:

- **Sky segmentation** for detecting sky regions in video frames. We use our methods to process dataset, train deep learning model, detect and up-sample to original size. Our method produces soft sky matte for a more accurate detection result and a more visually pleasing blending effect.
- **Motion Estimation:** We detect feature of each adjacent frame for recovering the motion of the sky in original video. We suppose the sky and the in-sky objects (e.g.,sun, clouds) are located at infinity and their movement relative to the foreground is Affine.
- **Sky Image Blending:** We improve Bayesian matting equation through changing some parameters including foreground image, predicted sky matte and motion parameters. After creating the skybox, it be recolored and relighted and add halo effect to make the output video more realistic and beautiful.

The contributions of this work for automatic video sky replacement are summarized as follows:

1) We propose an effective method of sky segmentation using K-Nearest Neighbors for refining and inpainting annotations, SegNet architecture whose accuracy is high and adjust some parameters in Image Processing formulas to satisfy our method.
2) Our methods conducted in videos instead of static images and videos using sensors and applied for offline video and online video framework.

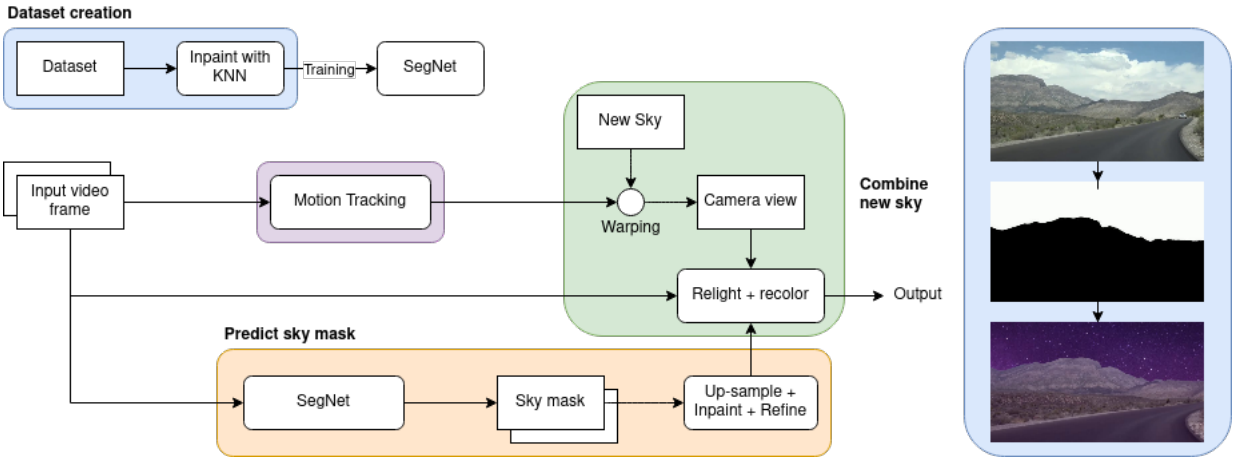


Fig. 1: An overview our method

II. Related Work

Sky replacement editing is common in photo processing and film post-production. Place *et al.* proposed a method named Segmenting Sky Pixels in Images [8], they fine-tuned the RefineNet-Res101-Cityscapes model on the existing big dataset that are SkyFinder dataset and subset from SUN dataset. Their method has very good results with the picture having clear boundaries between sky and foreground but not when the picture contains people or something that has complex details. Sky Optimization proposed by Liba *et al.* is to our best knowledge one of the first works that involve sky replacement.

Liba *et al.* [4] focus process images of the sky in low-light. They introduce refined and inpainted annotations of the sky image dataset based on the subset of the ADE20k dataset [9]. These "undetermined" areas are inpainted by using density estimation and refined by using guided filter. They also designed a UNet like CNN architecture for sky segmentation and followed their method Morphnet for model compression and acceleration. Density estimation only uses the distribution of RGB values of the annotated "sky" pixels to inpaint the pixels in the "undetermined" section. SkyAR of Zou [1] uses the same methods to refine and inpaint annotations. Different from sky segmentation in models architecture, Zou [1] uses ResNet-50 as an encoder and the decoder are consists 5 layers by Coordinate Convolution.

To estimate the motion of the camera, we estimate the motion of the feature in the original video and then render the virtual sky background by warping a 3D skybox template image onto the perspective window. See Fig.1 to understand how we blend the frame and sky template based on the estimated motion and the predicted sky matte.

In the stage of combining video foreground with background image, our first naive approach is extracting segmented part of background image and pasting it into the video mask. However, this approach is not effective because there is a high contrast between two components [10]. Therefore, there are some suggested approaches to improve the realism. In terms of image blending, Rawat *et al.* [6] finds association between foreground query and background images and then uses Laplacian Pyramid Blending in a seamless way to reduce composition artifacts. In this approach, two images are blurred and subsampled to create pyramid, then combined through each layer and finally the blended pyramid is collapsed to create a new image. SkyFinder *et al.* [5] applies category-specific color transfer in HSV space, computes the color transfer variables and transfers to non-sky region. In a recent work of Tran *et al.* [7], a method called "Fakeye" does not replace sky completely and uses Photoshop-like image blending algorithms. T. Halperin *et al.* [11] inherits Bayesian matting equation with α as segmentation mask and also applies exposure changes and camera vignetting to the transferred sky pixels. SkyAR of Zou [1] uses the same equation but applies recoloring and relighting methods and adds halo effect.

III. Methodology

A. Dataset creation

Dataset We train our sky matting network on the ADE20K dataset [9]. It is a large and high-quality dataset for Semantic Segmentation. We get all images which contain sky labels and resize them to (224,224). There are 3,000 images in the training set and 500 images in the validation set.

Refined and Inpainted Annotations Here we show a method for efficiently inpainting annotations. We start with coarse manual annotation dividing the image into two sections: "sky", "not sky". The annotations were made by humans that they clicked to create polygons so the annotations are not details (see Fig. 1). The Sky region always did not contains enough "sky" pixels but the Foreground region may contain both "sky" and "not sky" pixels.

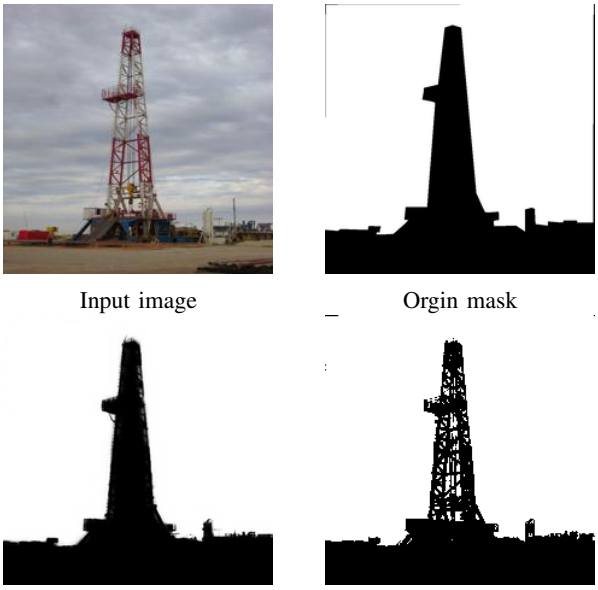
Masks in the dataset introduced by Liba *et al.* [4] were inpainted by density estimation and guided filter. Because this method only uses RGB values to inpaint, it just made boundaries between sky and foreground more clear but it doesn't work well if the picture has many sky partitions and large range color of sky. To predict pixel is "sky" or "not sky",

it bases on the pixel's coordinate, color and texture. We use K-Nearest Neighbors to inpaint Foreground region. Pixels I is defined by vector V_I:

$$V_I = [\lambda_1 h, \lambda_2 I_r, \lambda_3 I_g, \lambda_4 I_b, \lambda_5 D(I_c, J_c)]$$

where (w, h) is the I coordinate, I_r, I_b, I_c are RGB values. $I_c = [0.7 * I_b, 0.3 * I_r]$ and J coordinate is $(w, h-1)$. $D(I_c, J_c)$ is Euclidean distance between I_c and J_c and λ's are the weights for each term. Here, we use 0.7 and 0.3 in $I_c = [0.7 * I_b, 0.3 * I_r]$ because we want to highlight sky areas. We use equal weights for 4 terms ($\lambda_1 = \lambda_2 = \lambda_3 = \lambda_4 = 1$) and higher weight ($\lambda_5 = 100$) for ensuring the boundary smoothness.

For each image, it was down-sampled from size (224,224) to size (112,112). We create the dataset for KNN models with vectors that contain information of all pixels in the image and labels from the original mask (1-'sky', 0-'foreground') and choose k=10. Finally, we predicted "not sky" pixels in (224,224) image and overwrite the origin mask. (See Fig.3).



Input image | Orgin mask

Inpaint mask with DE+DF introduced by Liba *et al.* | Inpaint mask with our KNN method

Fig. 2: Results of Refined and Inpainted Annotations

B. Model architecture

We use SegNet introduced by Badrinarayanan *et al.* [12]. SegNet has an encoder network and a corresponding decoder network, followed by a final pixelwise classification layer. We implement SegNet in which corresponds to the first 13 convolutional layers in the VGG16 network and during upsampling, the max-pooling indices at the corresponding encoder layer are recalled to upsample. Follow Badrinarayanan *et al.* [12], SegNet outperforms FCN, DeconvNet, DeepLabv3..., has higher accuracy for large-size classes and reach accuracy 0.961 with sky class on CamVid dataset for Road Scene Segmentation.

Segnet was trained with the refined and inpainted dataset. We set batch size to 16. Input images have size (224,224,3). We set learning rate $1e^{-4}$ and reduce 100 times per 300 epochs. With GPU Nvidia P100 and stop at 780 epochs. We train our model by using Adam optimizer. Our matting network is implemented based on PyTorch. Image augmentations we used in training include horizontal-flip, random-crop and random-brightness (see Fig.3).

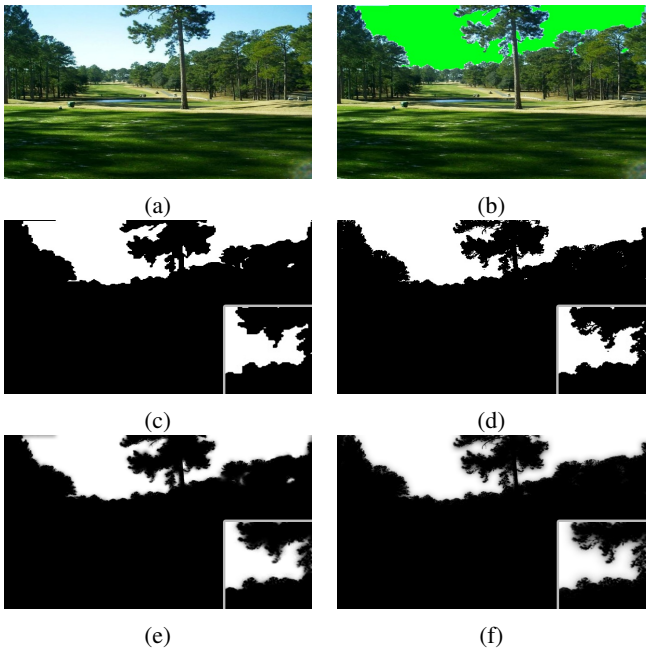


Fig. 3: Our method for post-process the sky segmentation. (a) An RGB input image. (b), (c) Mask is predicted by SegNet but it is not fit all boundaries. (d) Mask of (c) following KNN inpainting without refinement is detailed enough but sky matte is not soft. (e) Mask of (d) following guided filter refinement seems soft but it does not detail enough. (f) This mask of (d) follow inpainting and refinement is detailed and includes continuous values that make the mask seem soft.

C. Up-sample and Refine

Up-sampling Output masks have (224,224) pixels but we need to apply mask to the original size (,680) frame. Mask was resize miss or wrong some "sky" pixel. Alternatively, the outputs of SegNet don't fit all boundaries and are wrong on edge of the frame. We resize the output mask to the original size and use the same above methods in inpainted annotations for inpaint new resize mask but we inpaint both "sky" pixels and "not sky" pixels (see Fig. 2(c)(d)).

Refinement After we obtain an inpainted mask, we want our mask softer and looked real when combining new sky and blending with continuous values instead of hard pixel mask. We use the Guided Filtering technique [13] to refine. Guided Filtering is a well-known structure-preserving filtering method. It is high efficiency and simplicity and has better behaviors near edges and better computational complexity than Bilateral Filter [14] which is another popular filter. Liba *et al.* [4] and

Zou [1] also introduced this technique for refinement and the results they got are impressive. (see Fig.3)

D. Motion Estimation

In our method, we assume the motion of the sky patterns is modeled by an Affine matrix $M \in R^{3\times3}$. We use Shi-Tomasi corner detection to detect feature points and track these points repetitively by applying optical flow which is called Lucas-Kanade method [15]. For each pair of adjacent frames, given two sets of 2D feature points, we use the RANSAC-based robust Affine estimation to compute the optimal 2D transformation with four degrees of freedom (limited to translation, rotation, and uniform scaling). Since we focus on the motion of the sky background, we only use feature points located within the sky area to compute the Affine parameters. When there are not enough feature points detected, we run depth estimation [16] on the current frame, and then use the feature points in the second far regions to compute the Affine parameters.

We assume the background motion between two frames is dominated by translation and rotation and apply kernel density estimation on the Euclid distance between the paired points in each adjacent frame. The matched points with small distance probability $P(d) < \eta$ be removed. because we experimented and we realized the feature points in the sky area usually have lower quality than those in the close-range area, we find that it is sometimes difficult to obtain stable motion estimation results solely based on setting a low tolerance on the RANSAC re-projection error.

After obtaining the movement of each adjacent frame,the Affine matrix $M^{(t)}$ across between the initial frame and the t-th frame in the video can be written as the following matrix multiplication form:

$$\tilde{M}^{(t)} = M^{(c)} \cdot (M^{(t)} \cdot M^{(t-1)} ... M^{(1)})$$

where $M^{(i)}$ (i = 1,...t) represents the estimated motion parameters between frame i-1 and i. $M^{(c)}$ are are the center crop parameters of the 3Dwhere $M^{(i)}$ (i = 1,...t) represents the estimated motion parameters between frame i-1 and i. $M^{(c)}$ are are the center crop parameters of the 3D template, i.e., shift and scale, depending on the field of view set by the virtual camera. The final sky background $B^{(t)}$ within the camera's perspective field at the frame t can be thus obtained by warping the background template

Template, i.e., shift and scale, depending on the field of view set by the virtual camera. The final sky background $B^{(t)}$ within the camera's perspective field at the frame t can be thus obtained by warping the background template B using the Affine parameters $M^{(t)}$. In our method, we use a simple way to build the 3D sky background image where we tile the image when the perspective window goes out of the image border during the warping. We set the final center cropping region of the warped sky template to the field of view of the virtual camera. We set the size of the virtual camera's view as 1/2 height x 1/2 width of the template sky image. Note that other center cropping sizes are also applicable, and using a smaller range of cropping produce a distant view effect captured by a telephoto camera.

E. Sky Image Blending

$I^{(t)}$, $A^{(t)}$ and $B^{(t)}$ are subsequently the video frame, the predicted sky matte and the background image at time t. Specifically, $A^{(t)}$ is the sky when its output value is high. Based on Bayesian Matting equation [17], the result $Y^{(t)}$ is linear combination of the $I^{(t)}$ and the background $B^{(t)}$, with $A^{(t)}$ as combination of pixel-wise.

$$Y^{(t)} = (1 - A^{(t)})I^{(t)} + A^{(t)}B^{(t)}$$

However, there is an inevitable difference between the foreground $I^{(t)}$ and the background $B^{(t)}$ with color tone and intensity; therefore video outputs may become irrational. Using recoloring and relighting techniques, we can balance the color and intensity between the video frame and the background image. We can follow these formulas to adjust $I^{(t)}$ before linear combination:

$$\hat{I}^{(t)} = I^{(t)} + \alpha(\mu^{(t)}_{B(A=1)} - \mu^{(t)}_{I(A=0)})(*)$$

$$I^{(t)} = \beta(\hat{I}^{(t)} + \mu^{(t)}_{I} - \hat{\mu}^{(t)}_{I})(**)$$

In two formulae above, $\mu^{(t)}_{I}$ and $\hat{\mu}^{(t)}_{I}$ are mean values of color pixels in the image $I^{(t)}$ and the image $\hat{I}^{(t)}$. Meanwhile, $\mu^{(t)}_{I(A=0)}$ and $\mu^{(t)}_{B(A=1)}$ are subsequently the mean value of color pixels at the location of sky regions and the mean value of color pixels at the location of non-sky regions. α and β are constants for recoloring and relighting factor with values below 1. In this adjustment step, the background transferred regional color-tune to the foreground image and the step (**) is adjusting the range of color values so that foreground and background image can be balanced well.

Another adjustment is extra halo effect, which makes video more realistic but lowers frame rate. Halo effect can be calculated by the following formula:

$$Z^{(t)} = 1 - (1 - Y^{(t)}) * (1 - H^{(t)})$$

$H^{(t)}$ is the halo effect value, which can be calculated by using blur function in OpenCV multiply with 0.5. In blur function, the source is $A^{(t)}B^{(t)}$ and the kernel size depending on output video resolution width is $(w/5, w/5)$.

IV. Experimental Analysis

A. Speed performance

Below table show the speed performance of our method. This results were tested on Google Colab with NVIDIA T4 GPU card and 1050Ti at the ouput resolutions of 640-320. Performance of 2 devices have same speed. Our methods reach 17 fps with only using SegNet [12]. When we inpaint and refine predicted mask, we need spend 35 - 37 seconds for rendering 1 frame . In this, refine processing take 5 - 6 seconds. We can easily speed up the processing by replacing the backbone, eg. ResNet-50 [18], MobileNet [19] and only refine mask.

Fig. 4: Video sky augmentation results by using our method.

	Method	Speed
640*360 pxl	SegNet	17 fps
	SegNet + Refine	0.198 fps
	SegNet + Refine + Inpaint	0.0248 fps

B. Aesthetics

We survey on our results (see in Fig.4) and get 78 feedbacks of students who learn in Art major and Computer Science major about sky mask, demo with yellow sky and purple sky (row 3 and 4 in Fig.4). Below table is average rating score of mask and 2 demo videos.

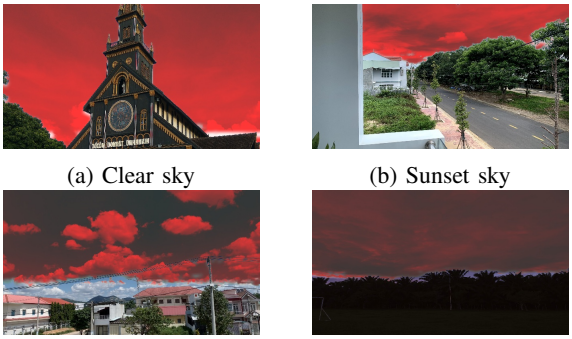
Video	Rating score
Sky mask	4.31
Demo with yellow sky	4.28
Demo with purple sky	4.13

Most of the feedback we get is about making the boundary between sky and foreground more realistic.

C. Controlled Experiments

Sky Segmentation We evaluate SegNet model on the validation set of our dataset. Below table show the IoU score on 2 GPU card NVIDIA T4 and 1050Ti. We get different scores because SegNet architecture [12] and different hardware handle floating point ops differently.

[1]https://bit.ly/ppnckh-skyreplacment-demo



(a) Clear sky (b) Sunset sky

(c) Cloudy sky (d) Other objects have same color with sky

Fig. 5: Results of Refined and Inpainted Annotations

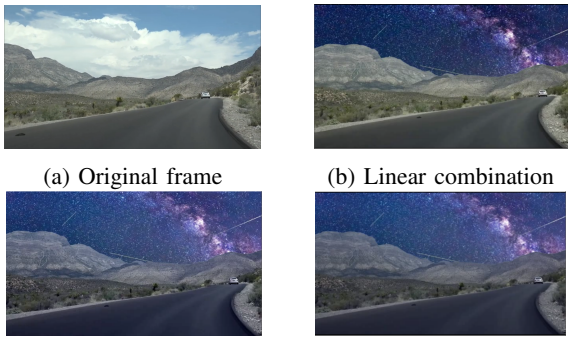
GPU card	IoU score
NVIDIA 1050Ti	0.75
NVIDIA T4	0.83

It can be seen that the hard pixel segmentation produces undesired artifacts near the sky boundary. Inpaint and Refine methods can get a more accurate sky region and get a much more visually pleasing blending result (Fig.3). In Fig.5 (a) Inpainting and refining made the sky mask more detail. It fit all details of the church and crown and the same result with the image (b), it is also discriminative between white wall, roof and sky have the white clouds. See in Fig.5 (c), thank for

inpainting methods, sky mask does not contain electric strings although they are little details across the sky.

Weakness of this method is processing with the red sky (sunset or sunrise sky), dark sky, and complex sky, eg. many clouds with shadow or rain or backlighted image. See in Fig.5 (c) and (d), SegNet detect almost sky but not all. Take a closer look, the sky mask in (c) has boundary fit the end of the cloud. The reason is that the clouds in the image are close to the photographer and create a shadow, SegNet predicts that is the boundary and we can not inpaint another sky region because there are too many wrong labels for KNN models. Almost all sky images in the dataset are daytime. Models easily predict fail results with nighttime images.

Blending with Relighting and Recoloring Adjustment We evaluate final blending results of our method with or without using recoloring and relighting technique with two different images. In terms of first image "night sky", in Fig.6, we compare the result of linear combination only Fig .6b, "linear + recoloring" Fig.6c and "linear + recoloring + relighting" Fig.6d. There is no considerable difference between Fig.6c and Fig.6d and these images give good harmonization as compared to Fig.6b. However, if the background input image is "sky with high brightness", it is obvious that Fig.7c makes video become realistic because of high albedo value overall, while Fig.7d shows high contrast between foreground and background of the video.



(a) Original frame (b) Linear combination

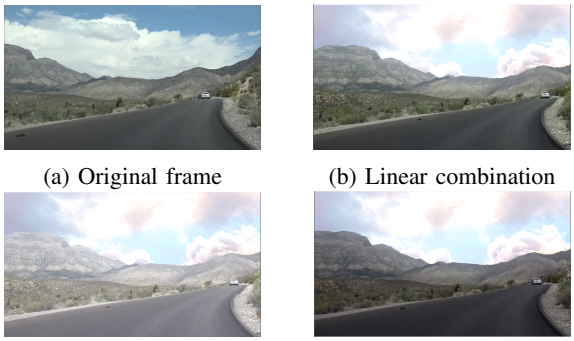
(c) Linear combination with recoloring (d) Linear combination with recoloring and relighting

Fig. 6: Results of Sky Image Blending with night sky background

Motion Estimation Because we assumed the feature points of videos that are located at infinity and using the feature points of objects that are second faraway to estimate motion that leads to inevitable errors. This creates non-sync movement between the foreground and the background (the sky).

V. Future work and Conclusions

In our future work, we focus on improving our research by solving three problems. First of all, we explore the effectiveness of sky-rendering based data augmentation for object detection and segmentation. Second, we import the processing capabilities of our model to be able to process videos with high frame rate (FPS). Finally, we are looking for a new approach



(a) Original frame (b) Linear combination

(c) Linear combination with recoloring (d) Linear combination with recoloring and relighting

Fig. 7: Results of Sky Image Blending with sunny sky

to estimate the motion of the sky.

In spite of the fact that our method shows a considerable improvement, there are other unexpected weather factors like the night sky and cloudy sky. Sky video augmentation is a new domain in computer vision, and automatic sky replacement with harmonization in video can be done with purely vision-based solution. Our method is comprised of three steps: soft sky matting, motion estimation, and sky blending. Our method does not require any sensor in smartphone as well as user interaction. With our method, it is hopeful that people have a way to easily generate highly realistic and cool sky animations in real-time. In COVID-19 pandemic period, there is a high hope that our research could have practical uses in images processing, such as real-time sky animation in videos, or sky swapping to generate new photos from old ones.

References

[1] Z. Zou, "Castle in the sky: Dynamic sky replacement and harmonization in videos," 2020.

[2] R. P. Mihail, S. Workman, Z. Bessinger, and N. Jacobs, "Sky segmentation in the wild: An empirical study," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2016, pp. 1–6.

[3] Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, and M.-H. Yang, "Sky is not the limit: semantic-aware sky replacement," *ACM Transactions on Graphics*, vol. 35, pp. 1–11, 07 2016.

[4] O. Liba, L. Cai, Y.-T. Tsai, E. Eban, Y. Movshovitz-Attias, Y. Pritch, H. Chen, and J. T. Barron, "Sky optimization: Semantically aware image processing of skies in low-light photography," 2020.

[5] L. Tao, l. Yuan, and J. Sun, "Skyfinder: Attribute-based sky image search," *ACM Trans. Graph.*, vol. 28, 08 2009.

[6] S. Rawat, S. Gairola, R. Shah, and P. Narayanan, *Find Me a Sky: A Data-Driven Method for Color-Consistent Sky Search and Replacement*. Springer, 01 2018, pp. 216–228.

[7] T. T. Yen Le Anh, "Fakeye: Sky augmentation with real-time sky segmentation and texture blending," in *Fourth Work-shop on Computer Vision for AR/VR*, 2020.

[8] A. B. Cecilia La Place, Aisha Urooj Khan, "Segmenting sky pixels in images," 2018.

[9] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[10] J.-F. Lalonde and A. A. Efros, "Using color compatibility for assessing image realism," in *2007 IEEE 11th International Conference on Computer Vision*. IEEE, 2007, pp. 1–8.

[11] T. Halperin, H. Cain, O. Bibi, and M. Werman, "Clear skies ahead: Towards real-time automatic sky replacement in video," in *Computer Graphics Forum*, vol. 38, no. 2. Wiley Online Library, 2019, pp. 207–218.

[12] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[13] K. He, J. Sun, and X. Tang, "Guided image filtering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 6, pp. 1397–1409, 2013.

[14] C. Tomasi and R. Manduchi, "Bilateral filtering for gray and color images," in *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, 1998, pp. 839–846.

[15] L. Y. Siong, S. S. Mokri, A. Hussain, N. Ibrahim, and M. M. Mustafa, "Motion detection using lucas kanade algorithm and application enhancement," in *2009 International Conference on Electrical Engineering and Informatics*, vol. 02, 2009, pp. 537–542.

[16] C. Godard, O. Mac Aodha, M. Firman, and G. J. Brostow, "Digging into self-supervised monocular depth prediction," October 2019.

[17] A. R. Smith and J. F. Blinn, "Blue screen matting," in *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996, pp. 259–268.

[18] S. R. Kaiming He, Xiangyu Zhang and J. Sun, "Deep residual learning for image recognition," 2015.

[19] B. C. Andrew G. Howard, Menglong Zhu, T. W. Dmitry Kalenichenko, Weijun Wang, and H. A. Marco Andreetto, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," 2017.