Moving Object Removal in Unlabeled Image Databases

Victor Chen, Daniel Mendoza, Yechao Zhang
Stanford University

***Abstract*—Moving object removal in consecutive images with the same frame of reference is a popular problem in image processing. Although various methodologies have been proposed, there has not been much exploration of machine learning techniques for solving this problem. In this paper, we identify a limitation of one of the conventional approaches for moving object removal in consecutive images called the Median Stack Filter. We then develop techniques that utilize machine learning algorithms such as K-Means and Deep Neural Networks. We show visually appealing results that exemplify improvement over the Median Stack Filter approach.**

I. Introduction

A. Problem Definition

Moving object removal is a frequently experienced problem for image processing. It is often the case that in consecutive images taken within a short period of time, moving objects in the foreground may clutter the intended subject of the images, the non-moving background. This project aims to develop techniques for removing moving objects in a sequence of images with the same frame of reference and a stationary camera and output a single image with the background extracted from the input images. Currently, most conventional approaches use metrics such as mean, median, and frequency of pixel colors to extract background. This project applies modern machine learning techniques for moving object removal in unlabeled datasets where ground truth background is unknown.

There have been previous classical approaches to the problem that do not make use of machine learning. As used in Adobe Photoshop's tool to remove moving objects, one method is "stack" all images on top of each other, and for each pixel of the output image, take the median value of the pixels at that corresponding location. This technique is called the Median Stack Filter (MSF) [9]; other metrics like the mean instead can also be used in a similar fashion. MSF performs poorly on a sequence of images in which there are "slow moving objects." Figure 1 illustrates the moving objects to be removed and the output of the MSF given 40 images. Notice that the MSF does not do well at removing the moving objects.

With the median stack filter as a baseline method, we designed and implemented an unsupervised learning technique as well as a general pipeline that can make use of machine learning techniques.

B. Related Work

Hori et al. address the problem of moving objects in panoramic images by calculating differences in color intensities of regions to identify candidates for moving objects and compensating for these regions with other color corrected images at nearby positions [2]. Azumi et al. implement the techniques as a case study in their proposed hardware/software codesign framework [8]. The essence of these approaches includes calculating the similarity of input images to detect the common background of each image. The similarity is evaluated by statistical metrics such as mean, median, and frequency of pixel colors. These techniques can be thought of as high pass filters in which only common pixel colors are outputted in the final image with the moving objects removed. This places an assumption on the rate at which objects are moving between input images frames as it does not handle slow moving objects well since slow moving objects are expected to have high similarity between input images. Wang et al. propose a technique to detect motion combing flux tensors and Gaussian modeling [6]. Recent development in machine vision algorithms that leverage deep learning have shown promising results. Shetty et al. propose a technique to apply deep learning to remove objects of a given class [7]. These new developments have yet to be applied to moving object removal in images and this project aims to explore the potential of deep learning to improve upon previous approaches. From defining a moving object class, we propose that deep learning can be utilized to improve upon the accuracy of moving object removal algorithms

Fig. 1: Left: Example image with moving objects highlighted; Right: Result of Median Stack Filter given 40 images

II. Unsupervised Methods

Our unsupervised model makes use of the K-Means algorithm to choose pixels from the image sequence that are most likely to be part of the background. In this section, we discuss three techniques for moving object removal with K-Means which are called Most Popular Cluster (MPC) selection, Lowest Variance Cluster (LVC) selection, and LVC selection with inpainting denoising. For each pixel in an array of input images, there are certain RGB values associated with the background (desired output) and the moving objects. We wish to differentiate between the RGB values for the background

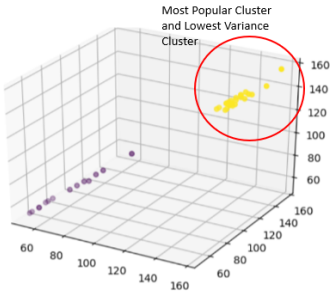

Fig. 2: K-Means clustering applied to a set of RGB values of the same pixel position over 40 images

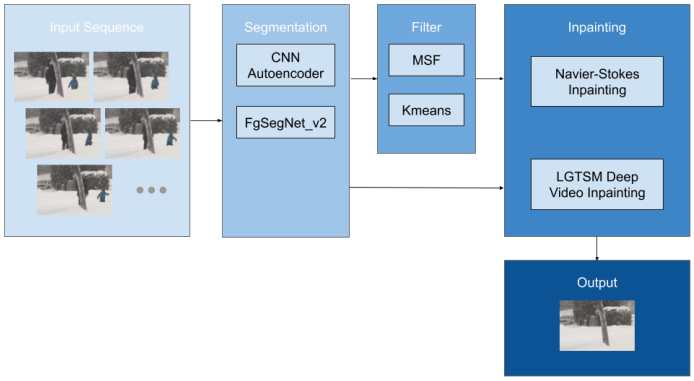

Fig. 3: Pipeline Algorithm for Moving Object Removal in Stationary Image Sequences

and the moving objects. If the RGB values of the background and moving objects are sufficiently different, then K-Means clustering can be applied to group together the pixels with similar RGB values at each position in the input images. The pixels at a certain position with similar RGB values can be assumed to approximately represent the same object. So, when applying K-Means to a set of RGB values at the same position in each input image, the resulting K clusters indicate the RGB values that belong to the background and $K-1$ moving objects.

A. Most Popular Cluster Selection

With K-Means we can group the RGB values of each pixel into clusters that represent different objects. For each pixel, the MPC algorithm executes K-Means and outputs the centroid with the most points assigned to it. For instance Figure 2 illustrates a plot of the RGB values of a pixel across multiple images. The yellow points correspond the most popular cluster and the MPC algorithm outputs that cluster's centroid.

B. Lowest Variance Cluster Selection

We implement another model using K-Means called Lowest Variance Cluster (LVC) selector that outputs the KMeans cluster centroid the corresponds to the cluster with the lowest variance for each pixel. This model does not depend on frequency of RGB values. The intuition behind this is that the RGB values of a pixel that has a moving object going through it will likely have a high variance. Thus, when K-Means groups the RGB values, the group with high variance corresponds to the RGB values of a moving object while the group with low variance corresponds to the background. Figure 2 reflects this intuition. The purple group which represents the RGB values of the moving object has a higher variance than the yellow group which represents the background.

C. LVC Selection with Inpainting Denoising

MPC and LVC selection exhibit trade-offs between frequency and variance of RGB values. High frequency RGB values for the same pixel across multiple images are likely to be the desired background. At the same time, K-Means clusters with low variance also seem likely to represent the background RGB values. When running both MPC and LVC selection on the same input image sequence, they would produce similar K-Means clusters, but choose different outputs. For some input sequences, MPC performs better than LVC and vice versa. Thus there exists ambiguity between which K-Means cluster corresponds to the desired output. Thus we implement LVC selection with inpainting denoising, in which for pixels that have contrasting cluster selection between MPC and LVC, we output an inpainting mask. The mask is then used for inpainting on the output of the LVC selector. We use the Navier-Stokes inpainter [3]. The inpainter fills in the RGB values of the output where the ambiguity of the background cluster exists.

III. Pipeline Algorithm

The previous unsupervised method builds the output image from a pixel basis; the color of each pixel is chosen only from the information of that exact pixel position in the image sequence and does not take into account the context of the image. In some situations, like with slow moving objects, the background in some locations is never fully revealed in a given sequence of images. When this occurs, MSF and our unsupervised method cannot choose the correct pixel value as it must be generated. We design a pipeline algorithm that takes into account the context of the image contents by using a combination of supervised and unsupervised techniques. The pipeline divides the problem into: segmentation of the image to identify and remove the foreground, filtering the extracted foreground masks, and using inpainting techniques to generate pixels to fill in any remaining holes in the image. Figure 3 illustrates each stage of the pipeline algorithm. The subcomponents of the pipeline are interchangeable as different techniques for the task at each stage can be applied. For instance in the segmentation stage, we have tested a vanilla convolutional autoencoder and the Foreground Segmentation Network [5]. For this paper, we have selected and evaluated a few non-machine learning and machine learning techniques for segmentation, filtering, and inpainting. The following sections describe each component of the pipeline and specific techniques we tested.

A. Segmentation

The role of segmentation is to extract masks of the moving objects (the foreground) from each image in the sequence. This step introduces context of where the moving objects are, so that pixels belonging to them can accurately be excluded from the final output. Many different algorithms have been developed for the goal of foreground detection/extraction. One common approach is the use of convolutional neural networks (CNNs), which can be used to extract low and high level features representations from images and perform well for foreground extraction. However, low level features resolution are reduced due to consecutive pooling and strided convolution operations in the pre-trained models [5].

We implement two deep learning techniques for the segmentation step: a vanilla convolutional autoencoder and a state of the art architecture, FgSegNet_v2 [5]. Both architectures are in essence CNNs. Each of these networks are trained on the same dataset for comparison of performance. We train the models with 25 images for each category in the CDNet2014 dataset [1] using 80%-20% split for training-validation.

CNN Autoencoder: Our CNN autoencoder implementation resembles a vanilla autoencoder architecture. The network inputs the gray-scale difference between two images and outputs the segmentation mask. The architecture contains 8 hidden layers each with ReLU activation while the output layer is sigmoid activation. The model is trained mean squared error loss.

Foreground Segmentation Network v2: FgSegNet_v2 that we implement for this project is designed to be a robust foreground segmentation network that can be trained with only a few examples but still provide accurate segmentation. This segmentation method uses fusion of features in various layers of the network that allow for multi-scale feature extraction in input images. This model is trained using binary cross entropy loss. More details on this architecture can be found in [5].

B. Filtering

Once masks of the moving objects for each frame of the sequence have been extracted, we can apply some filtering to the images with these masks. Because the sequence of images are temporally connected, if we only consider pixels that are not part of a mask at any time, we can use a filtering technique, like MSF or our K-Means approach, to confidently create the output image with pixels that are only part of the background (not included in the mask). This indicates that the quality of the output is dependant upon the performance of the segmentation stage. However, this filtering step nay still leave holes in the image in areas that are always part of the moving object masks in all frames of the sequence. For instance, Figure 4 illustrates an example input sequence in which part of the background is never shown and the corresponding mask that exemplifies the unrevealed area. Thus, the pipeline generates pixels in those unrevealed areas via inpainting.

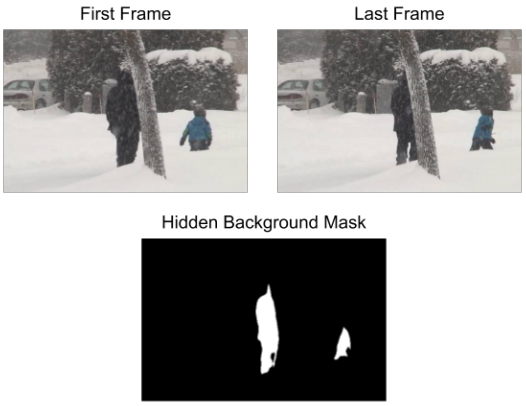

Fig. 4: Top: First and last input Frame of a sequence of images. Bottom: A mask that represents the area of the images in which the background is never revealed.

C. Inpainting

Inpainting is the process used to fill in and recover missing parts of an image. We apply inpainting in this scenario to generate the pixels of the background that are covered by the moving objects in the set of input images. In our procedure, we test two inpainting techniques: a non-machine learning approach with the Navier-Stokes equation and a state of the art video inpainter built with a Learnable Gated Temporal Shift Module (LGTSM) [4].

Navier-Stokes Inpainting: The algorithm connects computational fluid dynamics and image inpainting by solving the Navier-Stokes equations for an incompressible fluid [3]. The procedure first travels along the edges of the region selected to be generated and then continues lines joining points with same intensity while matching the gradient vectors of the region boundary. Then the rest of the region is filled by ensuring the minimum possible variance in the region.

LGTSM Deep Video Inpainting: Chang et al. developed an inpainting technique designed for recovering arbitrary mission regions in frames of videos [4]. Previous deep learning methods use 3D CNNs to model the spatial-temporal features inherent in videos to fill in unseen masked areas, but this method has many parameters and is difficult to train. 2D convolutions have been proven to be effective in image inpainting but lack the ability to capture temporal information. This technique takes inspiration from the Temporal Shift Module (TSM) which was used for activity recognition, enhances it, and applies it to 2D convolutions. Doing so enables the model to make use of features from temporally near and far neighboring frames; their results are similar to state of the art techniques using 3D convolutions but with fewer parameters and less training time. We made use of their model which was trained on the FaceForensics and Free-form Video Inpainting datasets.

Although our pipeline does not include free-form masks but rather masks of moving objects, we believed this video inpainter to be more relevant than other image inpainters be-

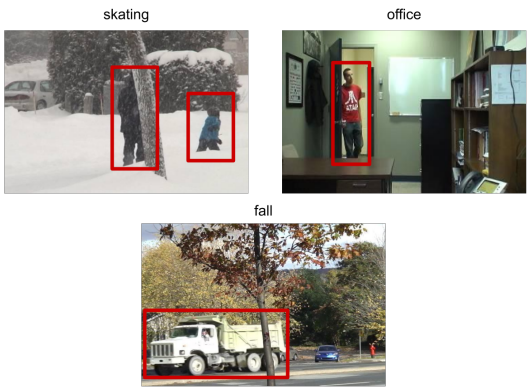

Fig. 5: Example Inputs with highlighted moving objects in red boxes

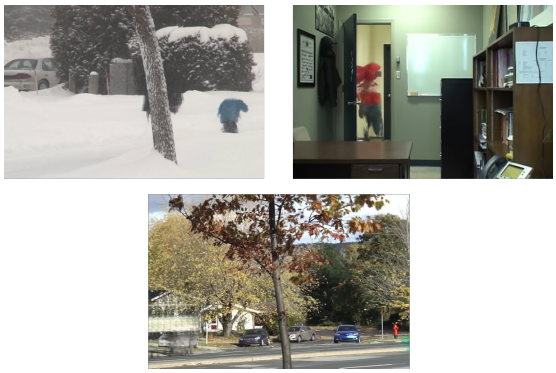

Fig. 6: Results for MSF given 40 consecutive images

cause our problem involves sequences of images. We expected the information gained from having images related by time to lead to better inpainting results.

IV. Experimental Results and Analysis

We evaluated MSF, our unsupervised approach, and different combinations of the pipeline on three different test image sequences of 40 images, each from the CDNET 2014 dataset containing 11 different video categories of 70,000 frames [1]. The code for the implemented models can be found here: https://github.com/vchen36/cs229-movingobjects. The three image sequences presented here are: people skating in the snow (*skating*), a man walking into an office and briefly reading a book before leaving (*office*), and cars and a large truck driving down the road with a dynamic background of leaves rustling in the trees (*fall*). Figure 5 illustrates the inputs and highlights the moving objects of each example.

Figure 6 illustrates the output of MSF baseline method given the example input sequences.

For the three sequences, we tested our unsupervised approach with MPC, LVC, and LVC with inpainting denoising and the results are shown in Figure 7. When observing the results of MPC, we see that it yield results similar to the baseline MSF, which is to be expected as both techniques determine the output by RGB value frequency at each pixel. LVC removes most of the moving objects but there remains to be noticeable noise. Further, the third column in Figure 7 shows the mask generated by the LVC with inpainting denoising approach. Navier-Stokes inpainting with these masks is applied to the output of the LVC to reduce the noise in the output. Notice that for each example the output of the LVC with inpainting denoising has less noise than the standard LVC output.

When testing our pipeline, for each of the image sequences, we ran our implementation of a CNN autoencoder and the FgSegNet_v2 to obtain masks of the moving objects for each frame. In Figure 8, we see the final output of using a MSF filter and the Navier-Stokes inpainter to complete the pipeline on each of the produced sets of masks. When looking at the results with our masks, we see that when using our CNN autoencoder masks, the pipeline does not perform very well, due to the segmentation step's inability to produce accurate masks. When using FgSegNet_v2 where the masks are much better, the objects are removed reasonably well, but visual artifacts are obvious from using Navier-Stokes inpainting. In comparison, if we use a state of the art deep learning inpainter, the results are much better. Despite inacurrate masks provided from the CNN autoencoder, the LGTSM video inpainter manages to produce visually appealing results. The results when using both FgSegNet_v2 for segmentation and the LGTSM deep video inpainting technique are the most visually appealing of our results in which the moving objects are completely removed and only a small amount of artifacts remain. This pipeline configuration shows strong improvement over the baseline MSF and shows that deep learning is a promising technique for moving object removal in consecutive images.

V. Conclusion

We developed unsupervised techniques for moving objects removal in consecutive images and showed the the LVC with denoising inpainting can remove objects more effectively than the baseline MSF approach. Further, we developed a pipeline approach containing a foreground segmentation stage, filtering stage, and inpainting stage. The components of our pipeline algorithm can be swapped out for different algorithms, and depending on the techniques used, we produce reasonable outputs. Knowing regions of foreground can produce more visually appealing results via pipeline algorithm and the output of the FgSegNet_v2 segmentation algorithm and LGTSM inpainting technique show strong improvement over the baseline MSF. This highlights the promise of deep neural networks for moving object removal techniques.

VI. Future Work

We plan to investigate different techniques for segmentation and inpainting, as the performance of these components determines the visual appeal of our results. Further, we plan to evaluate the performance after modifying the order of the stages in the pipeline algorithm, such as reversing the order of steps.

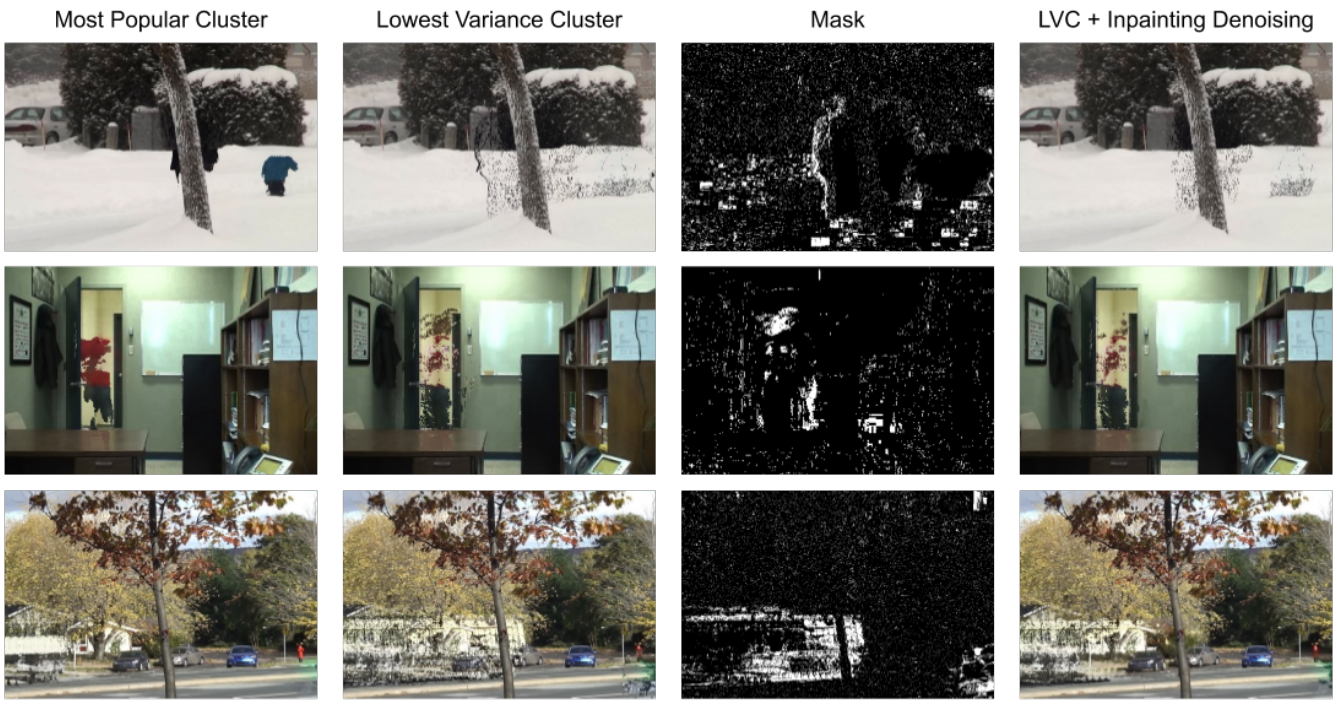

Fig. 7: Results for unsupervised methods given 40 consecutive images

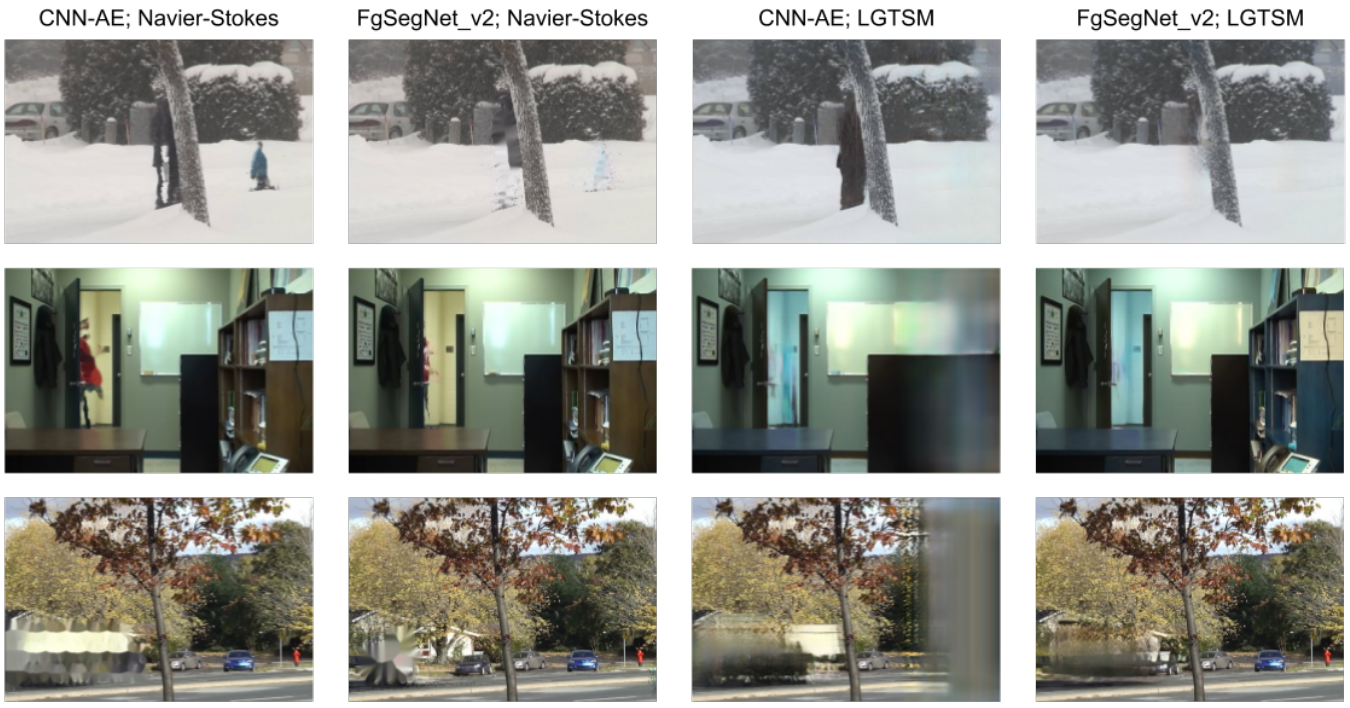

Fig. 8: Results for different configuration of pipeline given 40 consecutive images

VII. Contributions

Victor: In the early stages of the project, Victor worked on the evaluation and exploration of our baseline MSF. Later, he worked on researching different inpainting methods and implemented promising techniques for our specific problem using the test image sequences we had chosen. LGTSM

experimentation.

Daniel: MPC, LVC, LVC+inpainting implementation and experimentation. Vanilla CNN Autoencoder implementation, training, and experimentation. Compiling final results and a portion of writing for the final paper.

Yechao: Implementation and evaluation of MSF. Implementation of FgSegNet_v2, training models for all image categories of CDnet 2014 dataset. Also writing a portion for the final paper.

REFERENCES

[1] Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, CDnet 2014: An Expanded Change Detection Benchmark Dataset, in Proc. IEEE Workshop on Change Detection (CDW-2014) at CVPR-2014, pp. 387-394. 2014

[2] M. Hori, H. Takahashi, M. Kanbara, and N. Yokoya. (2010). Removal of Moving Objects and Inconsistencies in Color Tone for an Omnidirectional Image Database. 6469. 62-71. 10.1007/978-3-642-22819-3_7.

[3] M. Bertalmio, A. L. Bertozzi and G. Sapiro, "Navier-stokes, fluid dynamics, and image and video inpainting," Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001, Kauai, HI, USA, 2001, pp. I-I. doi: 10.1109/CVPR.2001.990497

[4] Chang, Ya-Liang, et al. "Learnable Gated Temporal Shift Module for Deep Video Inpainting." ArXiv:1907.01131 [Cs], July 2019. arXiv.org, http://arxiv.org/abs/1907.01131.

[5] L. A. Lim, and H. Y. Keles. "Learning Multi-Scale Features for Foreground Segmentation." Pattern Analysis and Applications, Aug. 2019. arXiv.org, doi:10.1007/s10044-019-00845-9.

[6] R. Wang, F. Bunyak, G. Seetharaman, and K. Palaniappan, "Static and Moving Object Detection Using Flux Tensor with Split Gaussian Models," 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Columbus, OH, 2014, pp. 420-424. doi: 10.1109/CVPRW.2014.68

[7] R. Shetty, M. Fritz, and B. Schiele, "Adversarial Scene Editing: Automatic Object Removal from Weak Supervision." Computer Vision and Pattern Recognition, Jun. 2018. arXiv:1806.01911

[8] T. Azumi, Y. S. Syahkal, Y. H. Azumi, H. Oyama, and R. Dömer (2013). TECSCE: HW/SW codesign framework for data parallelism based on software component. 403. 1-13. 10.1007/978-3-642-38853-81.

[9] M. Gabbouj, E. Coyle, N. Gallagher (1992). "An overview of median and stack filtering. Circuits Systems and Signal Processing". 11. 7-45. 10.1007/BF01189220.