The Value of Temporal Data in Moving Object Removal

Benjamin Coles

1

Abstract. *Removing moving objects from the intended focus of a video is a frequently met problem for videographers. Many conventional approaches use simple filtering techniques that use the frequency of pixels in each frame to attempt to isolate the background of the image; there has been very little exploration of the use of machine learning or temporal data for the task. In this paper we explore how the use of convolutional nets and temporal data can improve the clarity of extracted backgrounds.*

1. Introduction

Moving object removal is a frequent challenge found in image processing. When a constant background is the intended interest, it is often the case that moving objects, in the form of people, cars, etc., obscure the image. Our intent in this piece is to showcase a machine learning pipeline that takes a string of video frames and outputs the background image, removing all moving objects in the process.

Conventional approaches surprisingly lack the implementation of machine learning and use filtering approaches, such as median stack filtering, which we use here as a performance baseline. Median stack filtering merely takes the median RGB values of a pixel across all frames of a video to try and estimate the background. Furthermore, where conventional nets have been applied to this task, most fail to incorporate the temporal aspect of video, treating the task like foreground segmentation on a static image. Machine learning uses for this task are mainly focused on object segmentation, and dynamically create bounding areas for the movement of a specific object through the frames of a video. Once these bounding areas are created, then the area is removed from each still frame via inpainting. As such, most techniques fail particularly in the case of limited frames or a slow moving object, where some object blocks a component of the background for the entirety of the video. Here, we have implemented a pipeline that incorporates the untapped utility of temporal data and thrives when some component of the background remains unseen.

2. Related Work

Work related to this field segments our task into two, with few working to specifically remove images from video, focusing on object segmentation. The current fastest method for movement detection in an image is described by SiamMask (WANG et al., 2019). The approach uses a Siamese architecture network to create a bounds for a selected object in video, creating a mask over the selected object as it moves through frames. The network trains by placing as input the original image, and a smaller image centered on the bounding area. The mask is created online after a user draws out a rectangular bounding area for the moving object in question. Once the mask is created for each frame, the individual frames are put through an image inpainting algorithm, replacing the parts of the original

video as determined by the mask. While SiamMask performs extremely quickly, current attempts at using it for background retrieval fall short. The bounding box must be manually created around the object in the video you wish to remove, and removal is done on a frame-by-frame basis, with each frame treated independently of others.

External painting algorithms have existed for decades, and the one we employ in this investigation is the Telea 2004 inpainting algorithm (TELEA, 2004). The algorithm works by creating an estimation of a pixel's RGB values based on a weighted sum of its neighbors features, and performs quickly. This algorithm and others like it are commonly used in conjunction with segmentation to remove objects from backgrounds. While these methods are very good, they are by no means perfect, so the goal of our algorithm is to rely on their use at a minimum.

3. The Machine Learning Pipeline

The baseline approach of median stack filtering has the capacity to work extremely well in the case of non-dynamic backgrounds and a large amount of frames. The approach simply forms a stack of all frames and for every pixel and colour channel, takes the median value across these frames, to produce one background output image. This computational simplicity also enables it to return results rapidly for even high resolution, high frame rate video.

The pipeline we created maintains the benefits presented by median stack filtering, however, by introducing a two-stage pipeline, we significantly improve upon results in the case of few frames, dynamic backgrounds, and slow moving objects. The first step is a deviation of a foreground segmentation step, such that for each frame we create a mask, highlighting all moving objects. We then filter through these frames and apply image inpainting where necessary to produce a singular clear background image. Each step is described in more detail below.

3.1. Creation of Segmentation Masks

Segmentation masks were created for each frame of the inputted video. The role of these masks was to identify in each frame exactly which pixels of the image correspond to moving objects. This allowed for the isolation of the non-moving background in each individual frame, showing when and where the background was exposed. For this step we implemented a modification of the CNN architecture UNet, initially defined in 2015 for biomedical image scanning, but now used broadly across computer vision tasks (RONNEBERGER; FISCHER; BROX, 2015). Our adjustments to the network included the introduction of 2 dropout layers to prevent over fitting, and the reduction in the number of outputted feature maps, such that only the segmentation mask was produced.

We trained our model using the CDNet dataset, which features handmade segmentation masks for a variety of video types (Figure 1). We trained the model with 1000 images from each relevant category. However, we trained the model on three separate adaptations of the input to better understand the impact of temporal data on the creation of the masks. The three models were individually trained using the following feature maps as input:

(1) The grayscale difference between the image and the next image.

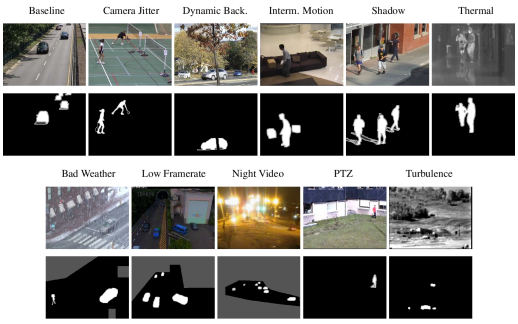


Figure 1. Samples of frames and their corresponding segmentation masks from 11 categories of CDNet 2014

(2) All of (1) in addition to the 3-channel RGB representation of the input image.

(3) All of (2) in addition to the grayscale difference between the image and the previous image.

We hoped that the inclusion of a very of input types would allow us to see what contributes the most information to the segmentation task, in order to observe the true value of temporal data.

3.2. Mask Processing and Background Creation

Once the segmentation masks for each frame of the video were created, they were used to build two separate images: a semi-completed background image and the inpainting mask, a mask that represents the region of the image never uncovered in the video. The background image was formed pixel-by-pixel. Each frame of the video and its corresponding segmentation mask were analyzed so that coordinates in the image frame that were exposed to the background of that region were filled in. This was accomplished by calculating the median RGB values for an individual pixel across all the frames where the background was exposed in that area (Figure 2 (a)). The inpainting mask was created similarly: all segmentation masks from each frame were stacked, and any location which was not (or rarely, to account for error) uncovered by a moving object was placed into the inpainting mask (Figure 2 (b)).

(a) Semi-completed background image

(b) Inpainting mask

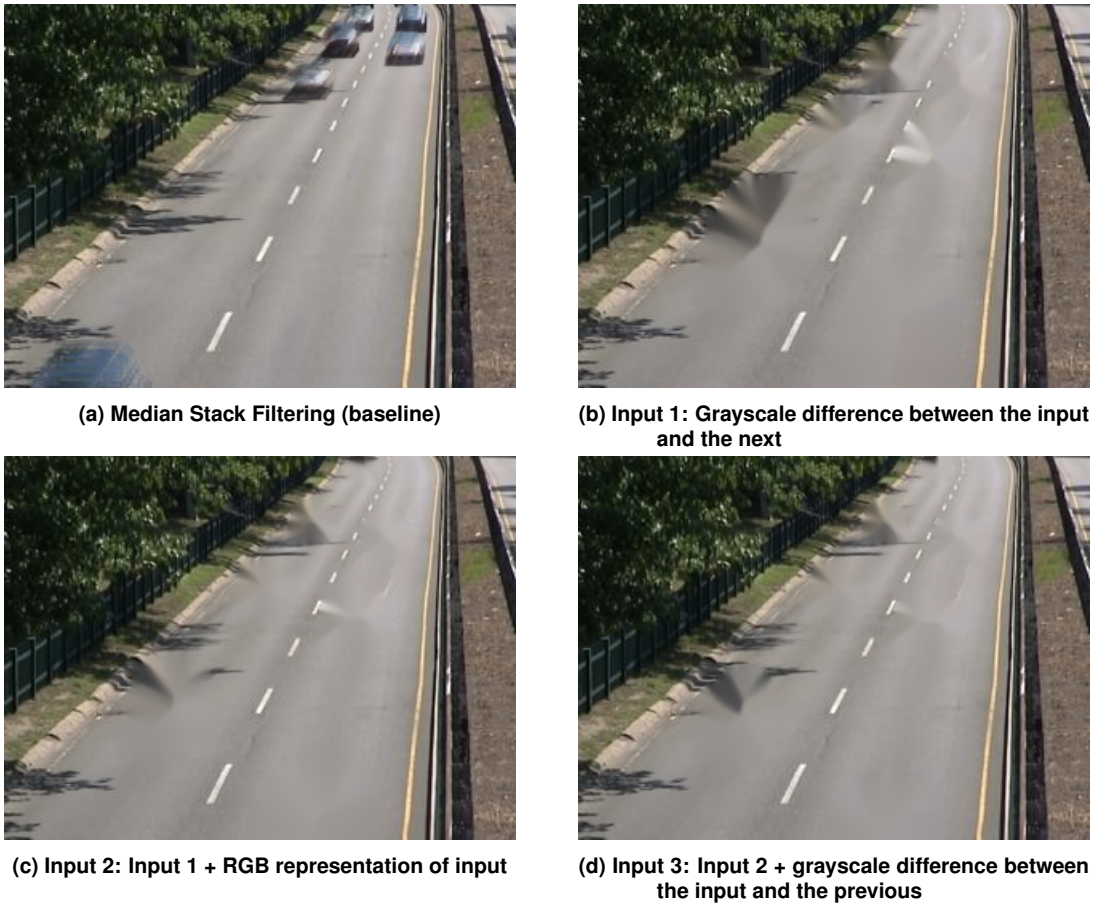
Figure 2. Outputs of network and inputs to inpainting algorithm

To produce our background image we input the semi-completed image and the

inpainting mask into the Telea inpainting algorithm. The algorithm predicted the content of the background never uncovered in the video to produce a completed image. It is likely the case, however, this step will not be necessary in a high proportion of use cases, when the entirity of the background is exposed at some time.

4. Pipeline and Analysis

We decided to keep one category of video (highway) out of our training set to use for validation. The validation set consisted of a single video made up of 35 frames, depicting cars moving down a highway. This footage was selected because it is a relatively small sample, and some portions of the background are never exposed. In theory, this should showcase how our pipeline performs in the cases where median stack filtering struggles. We present the result when these images were input into the 4 models mentioned in this report (Figure 3); we first have the baseline median stack filtering approach and then our 3 different methods of input into the machine learning pipeline.



(a) Median Stack Filtering (baseline)

(b) Input 1: Grayscale difference between the input and the next

(c) Input 2: Input 1 + RGB representation of input

(d) Input 3: Input 2 + grayscale difference between the input and the previous

Figure 3. Results on 'highway' validation set

(Figure 3 (a)) Median stack filtering performed predictably poor. The median pixel is rarely part of the true background, especially in the region around the distant cars that moved less distance across the frames. This leaves 'fuzzy' cars that have clearly not been removed. In fact, the space each car takes up has increased with the filtering technique - we actually see less of the background towards the back of the image than in any individual frame.

(Figure 3 (b)) Our first input immediately shows the success of our approach. No cars at all can be seen in the front or back of the image. Segmentation was inconsistent, as evident from the regions of distortion across the image. This input struggled especially with shadows near the trees on the left side.

(Figure 3 (c)) The second input sees clear improvement on the first. Including the original RGB input offered significant improvement to segmentation and consequently the task as a whole. The size of the distorted regions are far smaller and less noticeable: the shadows near the trees are much cleaner and segmented than those in input 1. As suggested by a significant decrease in loss during training, it is clear that an improvement in the performance of the network has allowed for a more successful result.

(Figure 3 (d)) The third input seems to perform on par with the second, improving in some areas and degrading others. Shadow quality around the trees degraded, but the roads look slightly more consistent with the baseline.

Perhaps most importantly, we can see that there is a very strong relationship between the quality of the mask creation and the quality of the outputted background. As we suspected, the initial mask creation is a limiting factor in the output image. Whilst the inclusion of temporal data allows us to easily identify moving objects, it also brings about the miss-classification of dynamic backgrounds, as seen by the distortion in the shadows of the trees moving in the wind. Generally, however, we see vast improvements across all inputs when compared to our baseline approach.

5. Conclusion and Future Work

Inclusion of convolutional nets and temporal data clearly improve upon our baseline method of median stack filtering. Indeed, we have shown that the inclusion of temporal data allowed for strong performance in the mask creation, and a robust pipeline enabled us to utilise this strength of the network to reform a background image. Unlike common filtering techniques, our pipeline performs well even in the case of limited input and slow moving objects.

Whilst our pipeline clearly showed promise, we were at times hindered by a significant flaw which is difficult to overcome: output images are qualitative, so it is difficult to assign unbiased metrics of success to each application of the method. We hope to mitigate this in the future by improving the individual aspects of the network: object segmentation and image inpainting.

Further work should focus on the consistency of generated output masks. Whilst the pipeline defined in this paper works rather well in most applications, its output is heavily dependant on the quality of mask creation. Furthermore, there is a lot of scope for experimentation with both model input and architecture; any improvements in this area of work will undoubtedly see the results of this pipeline improve even further.

References

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015. Disponível em: ⟨http://arxiv.org/abs/1505.04597⟩. 2

TELEA, A. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, v. 9, 01 2004. 2

WANG, Q. et al. Fast online object tracking and segmentation: A unifying approach. p. 1328–1338, 06 2019. 1