EECE 5639 Project 1

Neel Bhalla, Eric Grimaldi

October 12 2021

1 Abstract

In this project we explore a simple technique for motion detection in sequences captured with a stationary camera where most of the pixels belong to the stationary background. Since the intensity values observed at each pixel over time are nearly constant, except when a moving object passes in front of the background, we can detect such a moving object using simple methods for evaluating temporal gradients. In particular we investigate the performance a simple temporal difference vs a 1D derivative of Gaussian filter method, as well as the effects of various degrees of spatial smoothing. Finally we design a method for setting a threshold value for defining a mask our filter.

2 Approach

By the definition of our problem, we know that the majority of pixels in each frame will be a nearly constant value representing the background of our image, and that as a result an object which moves in front of our camera will appear as a relatively large change in pixel value over a short period of time. Knowing this, we can detect movement using some variety of filter which evaluates the temporal gradient of each pixel in each frame. With these temporal gradient values, we can then create a mask for each frame that obliterates any pixel with a low temporal gradient value. When applying these masks to their respective frames, this should leave us with a "cut out" of the moving objects within each frame.

As part of our investigation we also compare the performance of various levels of spatial smoothing on the performance of our temporal gradient methods. In these cases, spatial smoothing will be applied to each frame prior to temporal difference.

Temporal Difference

The temporal difference (TD) filter is our simplest method for approximating the temporal gradient. We define a 1D filter with the kernel

$$0.5\begin{bmatrix}-1 & 0 & 1\end{bmatrix}$$

We apply this kernel to the gray level of each pixel over the entire time domain. In simple terms, for each image in our sequence of images, we find the difference in gray level between

the preceding and following image. If this gray level difference is above the chosen threshold, then we decide that the pixel is seeing movement, and add it to our mask for that time step.

1D Derivative of Gaussian

We also use a derivative of Gaussian (DOG) method for approximating the temporal gradient. This method is similar, except that we calculate our 1D kernel using the equation for the derivative of the 1D Gaussian distribution a chosen σ_t value.

$$f(x) = K\frac{-x}{\sigma_t^2}exp(\frac{-x^2}{2\sigma_t^2})$$

First, we choose the kernel to have n elements such that

$$n = ceil(5\sigma_t) + n_0, \text{ where}$$

$$n_0 = \begin{cases} 0 & \text{if } ceil(5\sigma_t) \text{ is odd} \\ 1 & \text{if } ceil(5\sigma_t) \text{ is even} \end{cases}$$

This ensures that our kernel has enough time to approach zero for a given σ_t, and has an odd number of elements.

We then the calculate value of each element in the kernel such that the ith element is $f(x_i)$ where x_i is the ith element in $[\frac{-n+1}{2}, ..., 0, ..., \frac{n-1}{2}]$. We set $K = 1$ and instead normalize the full kernel at the end. This results in negative values before the center element, a center element of zero, and positive values after the center element. One can easily see a similarity in sign to the TD kernel.

Further, one can see that this method constructs a kernel where σ_t controls the number of time steps seen by the kernel, as well as the weight of each time step, where weight decays exponentially with temporal distance from the present.

Finally, its worth noting that a DOG with $\sigma_t \in (0.2, 0.6]$ would simplify to the TD kernel.

Box Filter

For spatial smoothing, we initially use box filters with 3x3 and 5x5 kernels.

Box filtering is an incredibly simplistic method. For each pixel in each image, instead of considering the pixel's raw value we consider the average value of it and its neighbors within some nxn neighborhood. We achieve this by defining an nxn kernel

$$\frac{1}{n^2}\begin{bmatrix} 1 & ... & 1 \\ ... & 1 & ... \\ 1 & ... & 1 \end{bmatrix}_{n\mathbf{x}n}, \text{ where } n \text{ is odd}$$

We then convolve this kernel with the image before applying our temporal gradient filtering for motion detection.

2D Gaussian Filter

We also use a Gaussian method for spatial smoothing. Somewhat similarly to our DOG method for temporal gradient, we calculate our kernel using the 2D Gaussian distribution and a chosen σ_s value.

$$f(x) = Kexp(\frac{x^2 + y^2}{2\sigma_s^2})$$

Again, for similar reasoning, we choose the kernel to have nxn elements such that

$$n = ceil(5\sigma_t) + n_0, \text{ where}$$

$$n_0 = \begin{cases} 0 & \text{if } ceil(5\sigma_s) \text{ is odd} \\ 1 & \text{if } ceil(5\sigma_s) \text{ is even} \end{cases}$$

We then calculate the value of each element in the kernel such that the (i, j) element is $f(x_i, y_j)$ where x_i and y_j are the ith and jth elements in $[\frac{-n+1}{2}, ..., 0, ..., \frac{n-1}{2}]$. We set $K = 1$ and instead normalize the full kernel at the end. This gives us a positive kernel with a maximum element in the center pixel.

Further, one can see that this is effectively a weighted average of some neighborhood of pixels, where the weight of elements decays exponentially with distance from the center.

3 Experiments

In this project, we conduct three main experiments. In essence, each experiment is a sweep of parameters controlling some subject of interest.

Temporal Gradient

First, we compare the performance of TD, and DOG with $\sigma_t \in [1.0, 1.4, 1.8, 2.2]$. We chose σ_t in this way to guarantee each value tested was a new kernel size. We then apply these motion detection methods to the Office data set. We analyze side by side video playback of the bit mask, original footage, and footage masked for motion detection, as well as still frames. A selection of results can be seen in Figure (1).

Spatial Smoothing

Second, we compare the impact of spatial smoothing from a 3x3 box filter, a 5x5 box filter, and Gaussian filters with $\sigma_s \in [1.0, 1.4, 1.8, 2.2]$. We chose σ_t in this way to guarantee each value tested was a new kernel size. We then smooth the Office data set using each of these spatial smoothing methods. In order to clearly see the effect of smoothing, we use the simple TD algorithm for motion detection in all cases. We analyze side by side video playback of the bit mask, original footage, and footage masked for motion detection, as well as still frames. A selection of results can be seen in Figure (2).

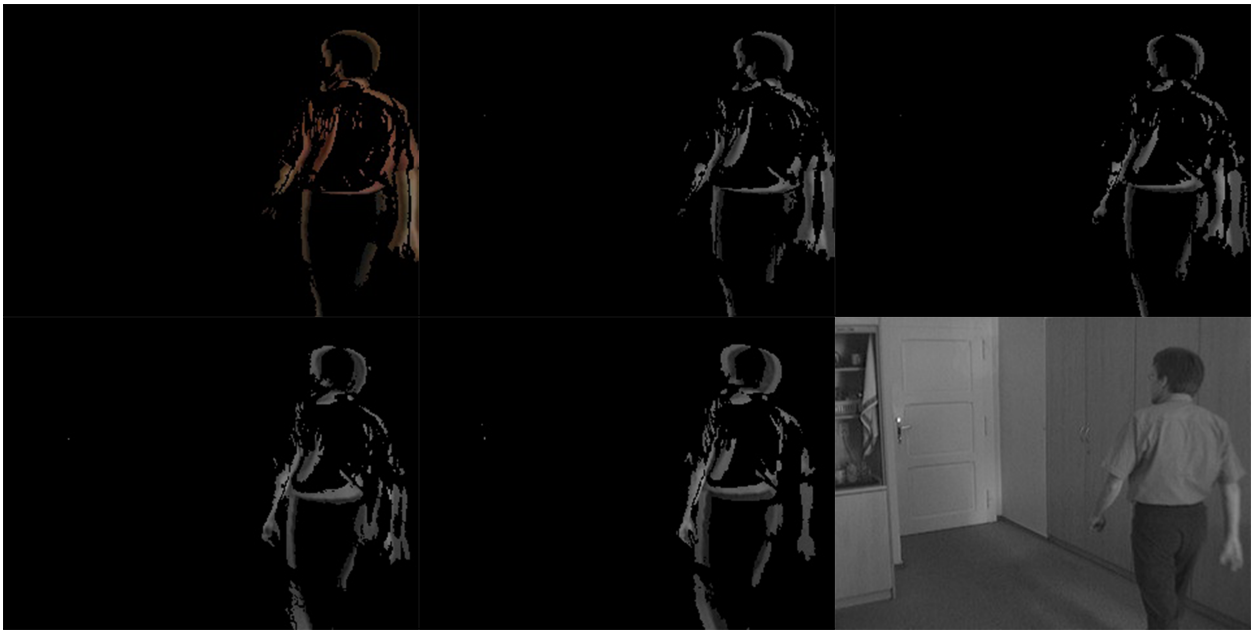

Figure 1: Results of our temporal gradient study. From top left to bottom right: TD, DOG with $\sigma_t = 1.0$, $\sigma_t = 1.4$, $\sigma_t = 1.8$, $\sigma_t = 2.2$, and the original frame.

Thresholding

Finally, we look at the impact of the threshold setting across a large range of thresholds $\in [0, ..., 20]$. We chose the range of thresholds as such in order to illustrate the failure modes at both low and high threshold values. Again, in order to clearly see the effect of the threshold, we use the simple TD algorithm for motion detection on the Office data set for all cases. We analyze side by side video playback of the bit mask, original footage, and footage masked for motion detection, as well as still frames. A selection of results can be seen in Figures (3) and (4).

4 Discussion

Temporal Gradient

In the simple case of TD, so few samples are considered in our estimate of the temporal gradient, that the impact of noise is high. This often leads to "speckled" and "jagged" masks of detected motion due brief variations in the background, or brief lack of variations in the moving object. The mask produced by this method is subject to somewhat sudden variation from frame to frame when watching video playback of the full sequence.

The DOG minimizes this effect by looking at a larger number of time steps, but weighting them according to their closeness to the actual current time step. This expanded range reduces the impact of individual noisy samples. It is hard to quantify, but when watching the video footage masked by the DOG motion detector, the mask seems to "flow" better from frame to frame - it seems to improves the overall consistency of the mask.

Figure 2: Results of our spatial smoothing study. From top left to bottom right: No smoothing, 3x3 box filter, 5x5 box filter, the original frame, Gaussian filter with $\sigma_s = 1.0$, $\sigma_s = 1.4$, $\sigma_s = 1.8$, and $\sigma_s = 2.2$.

However, the DOG experiences something of a failure mode at high σ_t values. As σ_t increases, the size of the kernel increases, and the filter considers a wider and wider period of time when detecting motion in each pixel. A negative consequence of this is that pixels which recently *but not currently* observed motion will still be considered by this method to have seen motion. As a result, the mask gains something of a "light trail" around moving objects in the frame. Additionally, since the kernel is symmetric, with the time step under consideration in the center, we get a symmetric effect *going forward in time* as well.

Additionally, at very high values of σ_t, many constant background pixels before and after the actual motion begin to enter the estimate of the temporal gradient. In this case, motion might be obliterated by the large sample size of background pixels.

As noted earlier, If $\sigma_t \in (0.2, 0.6]$ then the DOG filter will simplify to the TD filter. This is because the filter creates a three element kernel, inversely symmetric, with center element zero, which normalizes to $0.5 \begin{bmatrix} -1 & 0 & 1 \end{bmatrix}$.

Finally, both methods have trouble detecting the middle of moving objects. In short, if the object is slow enough, or the frame rate is fast enough, then the pixels in the middle of a moving object of relatively uniform color will appear to not change within the window of time where the moving object occupies those pixels.

Spatial Smoothing

Spatial smoothing of all kinds has an immediate benefit: the dilution of noise. As more samples are considered, our estimate of the pixel converges, and the impact of noise is obliterated. This is particularly valuable for reducing number of the "speckled" and "jagged" elements in the masks produced by out motion detectors.

However, in the case of temporal gradient motion detection, its important to remember that we are detecting motion through observation of unusually high change in pixel value relative to some constant background pixel value. This means that if our spatial smoothing

Figure 3: Results of our threshold study for high values. From top left to bottom right: threshold = 1, 2, 3, 4, 5, and 6.

uses too large a window and takes in many nearby background pixels, it will dilute the unusually high change in pixel value that we are looking for, causing a failure to detect motion.

Additionally, there is a cosmetic effect of uncertain value: as the window of spatial smoothing increases, the mask produced by out motion detectors will acquire something like a "halo". In more specific terms, the unusually high change in pixel value present at the edge of a moving object will bleed out into the surrounding background pixels due to the smoothing effect; as a result, the temporal gradient estimate based on these smoothed values will find change in pixel value at some distance away from the actual moving object dictated by the smoothing window.

In the case of box filtering, the effects of window size become apparent quickly. The 3x3 smoothing sees virtually only improvement over the raw data. However, the 5x5 smoothing is already starting to cause the motion detectors to fail to see significant portions of the moving object.

The 2D Gaussian filtering, however, does not run into the same issues until higher window sizes. It seems 2D Gaussian filter with $\sigma_s = 1.8$, or a 9x9 window, has roughly similar levels of performance to the 5x5 box filter. This behavior is due to the exponential weighting of the Gaussian filter. Even though the filter is looking at a larger region, pixels farther away from the pixel under consideration are weighted much lower than itself or its immediate neighbors.

Figure 4: Results of our threshold study for high values. From top left to bottom right: threshold = 10, 11, 12, 13, 14, and 15.

Thresholding

Finally, we consider thresholding, the effect of which looms over all our other studies. From our experiments, it is readily apparent that the threshold for detection has two failure states. If the threshold is too low, then random fluctuations due to noise in environmental light and in the camera electronics will register as movement. If the threshold is too high, then sections of actual moving objects will fail to register as movement. Additionally, if smoothing is applied, a lower or higher threshold may increase or decrease the halo effect.

In our experiments, we empirically found best performance with threshold values between 6 and 10 for the TD case with no smoothing.

A more methodical strategy for determining a good threshold is to find the average of the standard deviation of temporal gradient of each pixel. Since the background is relatively constant, and the moving objects take up a relatively small part of each frame, we can treat the variation of these pixels as additive white Gaussian noise. In this case, the proposed calculation should produce a reasonable estimate the standard deviation of this noise distribution. Setting the threshold to twice the standard deviation should be high enough to ignore 95% of gray level fluctuations from noise; three times the standard deviation should be similarly high enough to ignore 99.7% of noise. To our thinking, the number of errors actually removed by increasing the threshold about three times the standard deviation is infinitesimally small, and the actual motion detection may begin to suffer due to the high threshold. We find that this method echoes our empirical finding that threshold values between 6 and 10 are appropriate.

5 Appendix: Code

Please see our git repository for copies of our code.

$$https://github.com/EAGrimaldi/EECE5639\_Project1$$