Unsupervised Detection of Background and Moving Cars

Yongning Wu, Xiao Ling

Problem Statement

Background subtraction, also known as foreground detection is typically the first step before more in-depth object recognition. In the unsupervised setting, we generally assume that still objects are part of background, with possibility of being occasionally covered or blurred by moving objects in the pictures.

In this simplified setting, we are particularly interesting in the case that

1. A fixed camera facing some traffic
2. Car is the main moving part.

This scenario may happen in some traffic monitoring camera. Our algorithm can help identify the moving cars, blocking object in the high way or intersections. Our goal is to provide an extendable extraction framework that separates background and moving objects. Given a coarse object detector, we believe that this detection framework can provide an enhanced object detection, which is easily extended to other use cases.

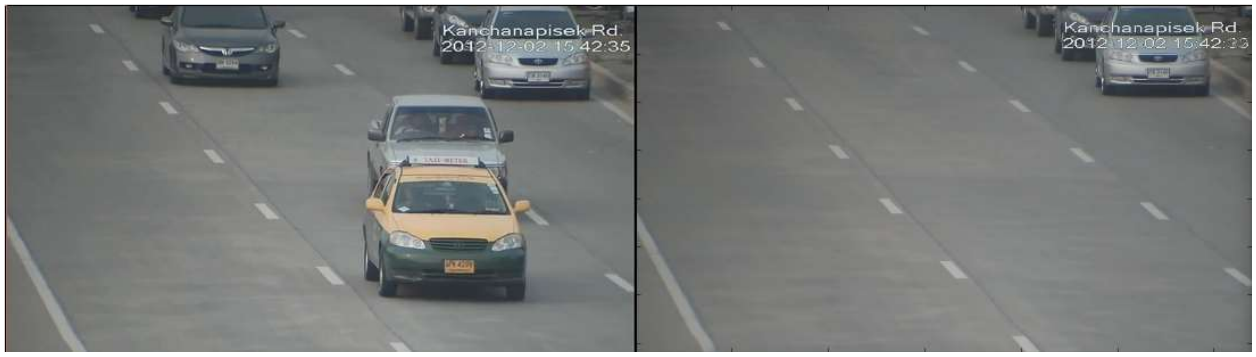
Introduction

Our detection framework can be divided into four main steps:

1. extract data file from video
2. [optional] apply a coarse object detector
3. converge on background
4. annotate frames

We expect coarse object detectors can help with tricky cases for better initialization and convergence. It does not need to very accurately identify objects. Below are sample input and background output:

FIGURE 1 (before and after)



Method

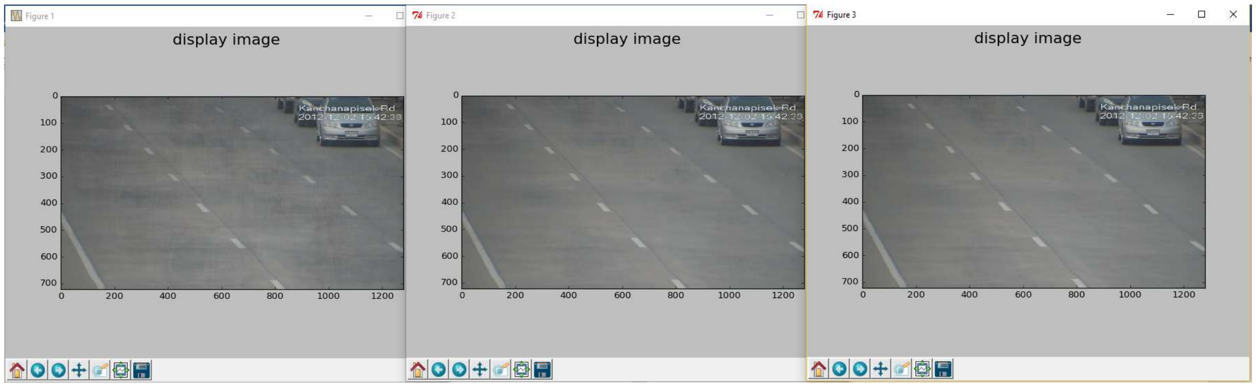
EM algorithm is the main method used to iteratively improve the estimation of unobserved background. Here, we assume there is a background predictor BP. BP relies on some prior knowledge about the background to make further prediction for pixels on frames. Then the likely background in the frames will help improve the estimation of the background. This chicken-egg problem can be solved by EM steps as below:

E-step: for each pixel on each frame, predict the likelihood of background with BP.

M-step: update background estimation by weighted average of all frame pixels.

Below, we should observe that after each iteration, the background becomes cleaner and stabilizes after a while. In the left frame, we can see some big cloudy grey stuff on the road. On the right, it becomes clear which color should be the road.

FIGURE 2 (show backgrounds after consecutive iterations)



Three important components we want to highlight and discuss more below.

Background Initialization

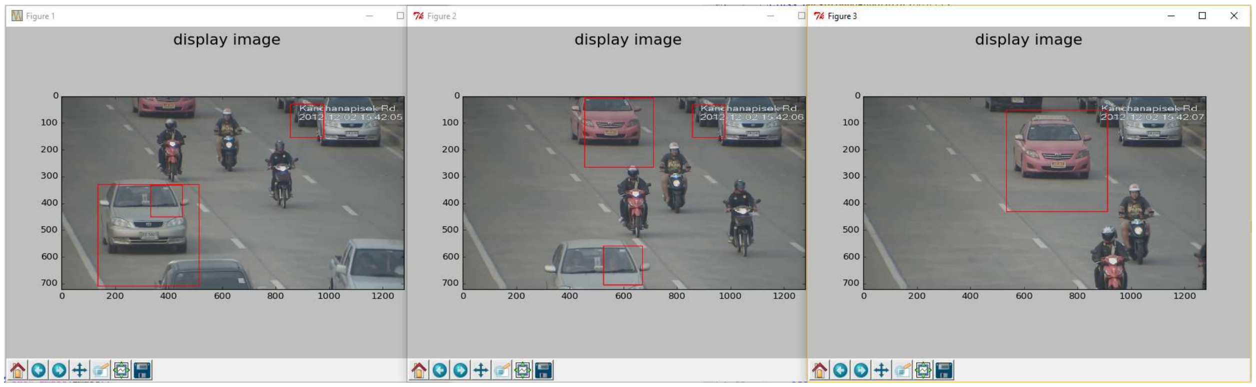
1. A simple version is to take unweighted average of original frames

   It can handle most of cases well in our experiment. We believe that some theoretic analysis can show that for a given time window, a pixel is occupied by background >50% of the time, then an unsupervised background detection is possible.

2. A sophisticated version is to apply an external car predictor which only gives a rough mask of the car

   This specialized predictor can overcome the previous limitation with better initialization and provide more guarantee that the converged background is the true background. This predictor does not need to accurately outline the moving object as long as it can provide a rough box shape that covers most of the objects. See figure 3 for example.

FIGURE 3 (show external predictor with box in figure)



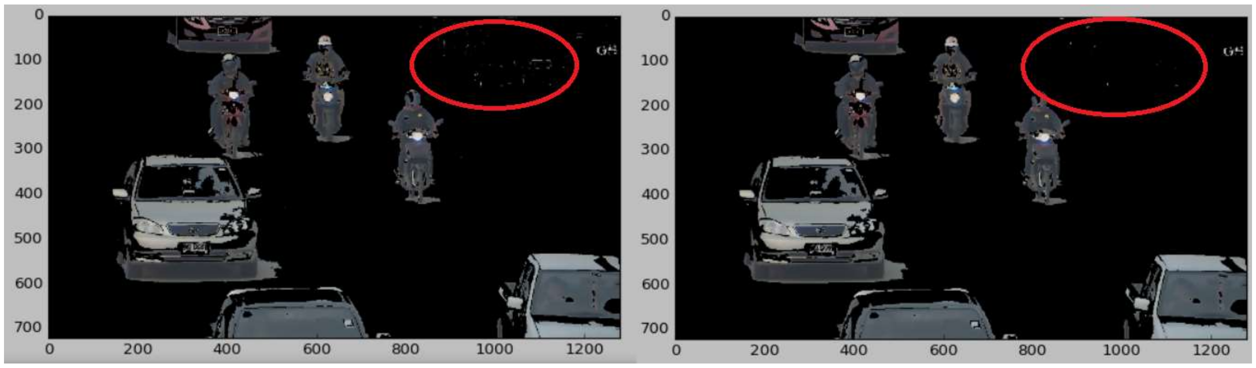
Background Predictor: BP

In our implementation, we have the predictor with the following characteristics:

1. Use Gaussian density function to map the closeness to background to probability [0,1]
2. Mean is the weighted average of background
3. Standard deviation starts with high value and tends to decrease with each iteration.
4. The prediction is smoothed in (x, y) dimension with Gaussian filter.
5. [TODO] further smoothed in time dimension

The reason to start with high standard deviation is to tolerate incorrect initialization. Then decrease the standard deviation to make the predictor tighter in the correct background color range. The smoothing can be done in both time and space dimension. Due to time constraint, we only implemented space dimension smoother. It will help suppress some local outliers either due to camera noise or other minor fluctuation. The effect is adjustable by the diameter of Gaussian filter.

FIGURE 4 (compare non-smooth [left] vs smooth [right])



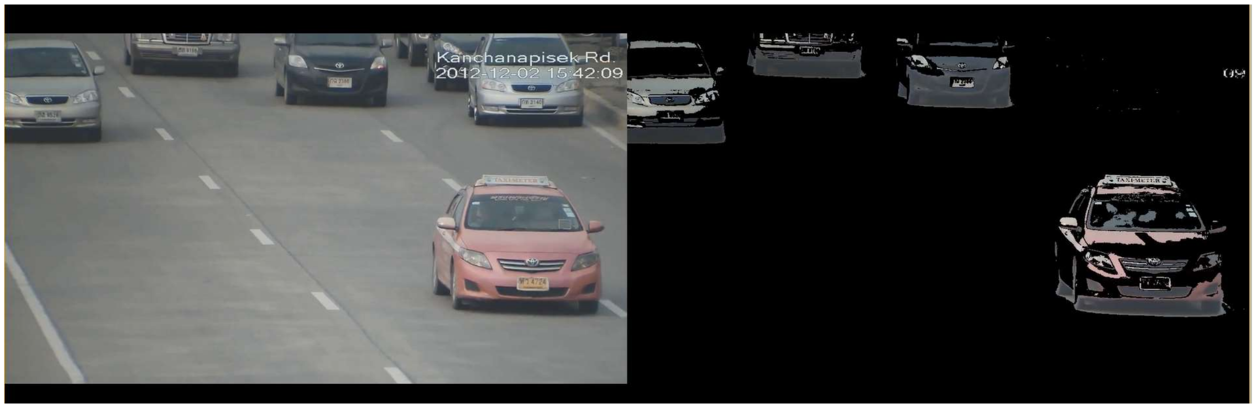
Foreground Annotation

Foreground annotation is not limited to input frames used in background extraction step. It tries to enforce a binary classification on any given frame. Its quality depends the following factors:

1. predictor (e.g. sigma in Gaussian), threshold
2. smoothing
3. object model

In the following annotated figure, we can see with proper threshold and smoothing, it can correctly identify the moving cars. However, lacking a car object model, it may classify some part of car as background if the color is close to background. We need to apply a Geometric package to construct a convex contour to mark the internal part of a car as non-background. Also the object model will help differentiate the shadow caused by the object vs the true object.

FIGURE 5 (show annotation)



Discussion

There are several directions to further improve this work.

1. Pixel level alignment to handle camera shaking
2. Better object model to disambiguate the similar background color on the object, shadow handling.
3. Space-temporal smoothing and long time range background update.

Appendix:

The code and instruction are available at

https://github.com/lingxiao1989/BackgroundDetectionfromSurveillanceVideos