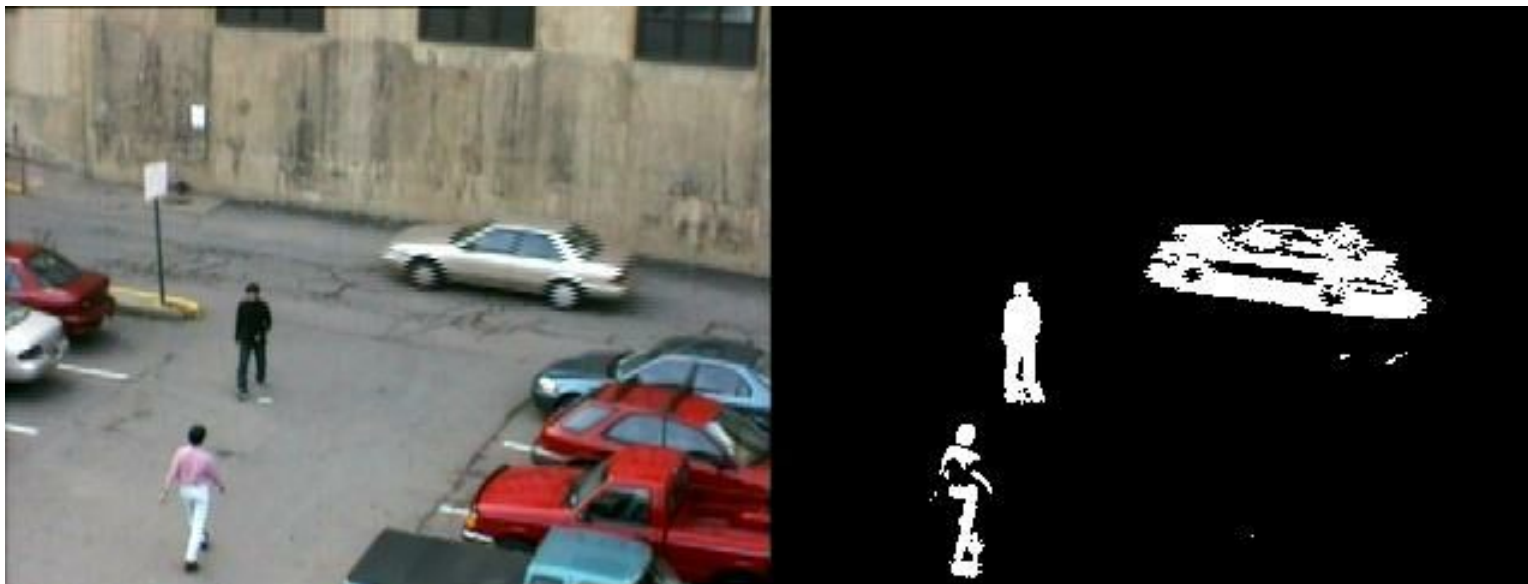
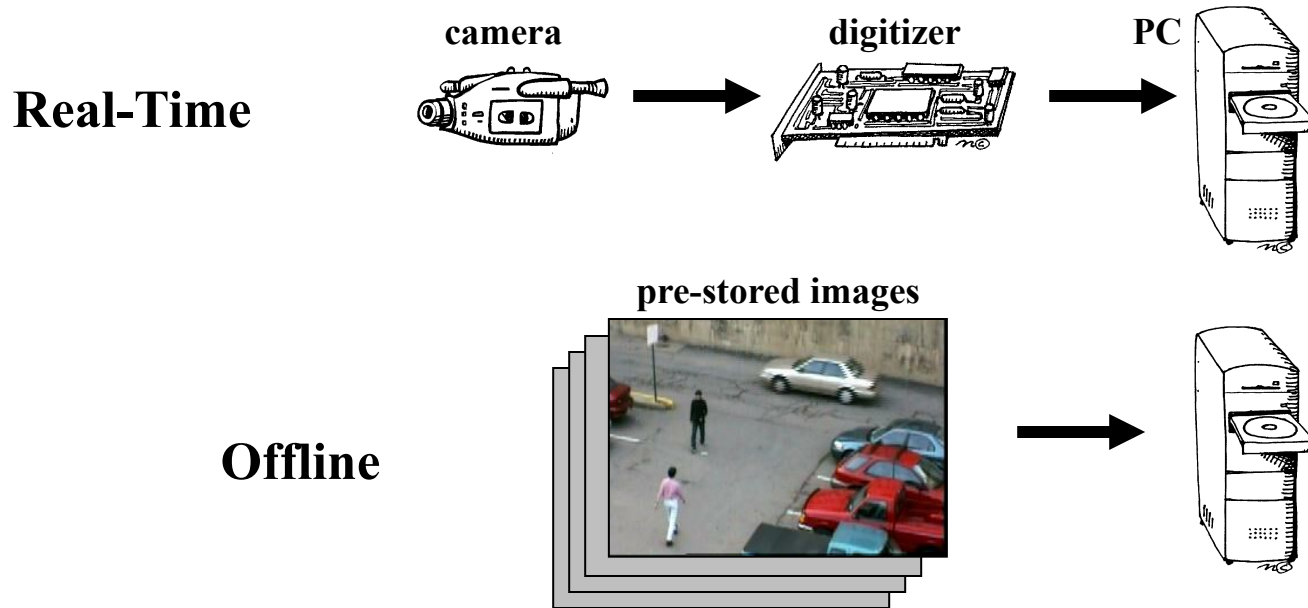


Video Change Detection



Basics of Video



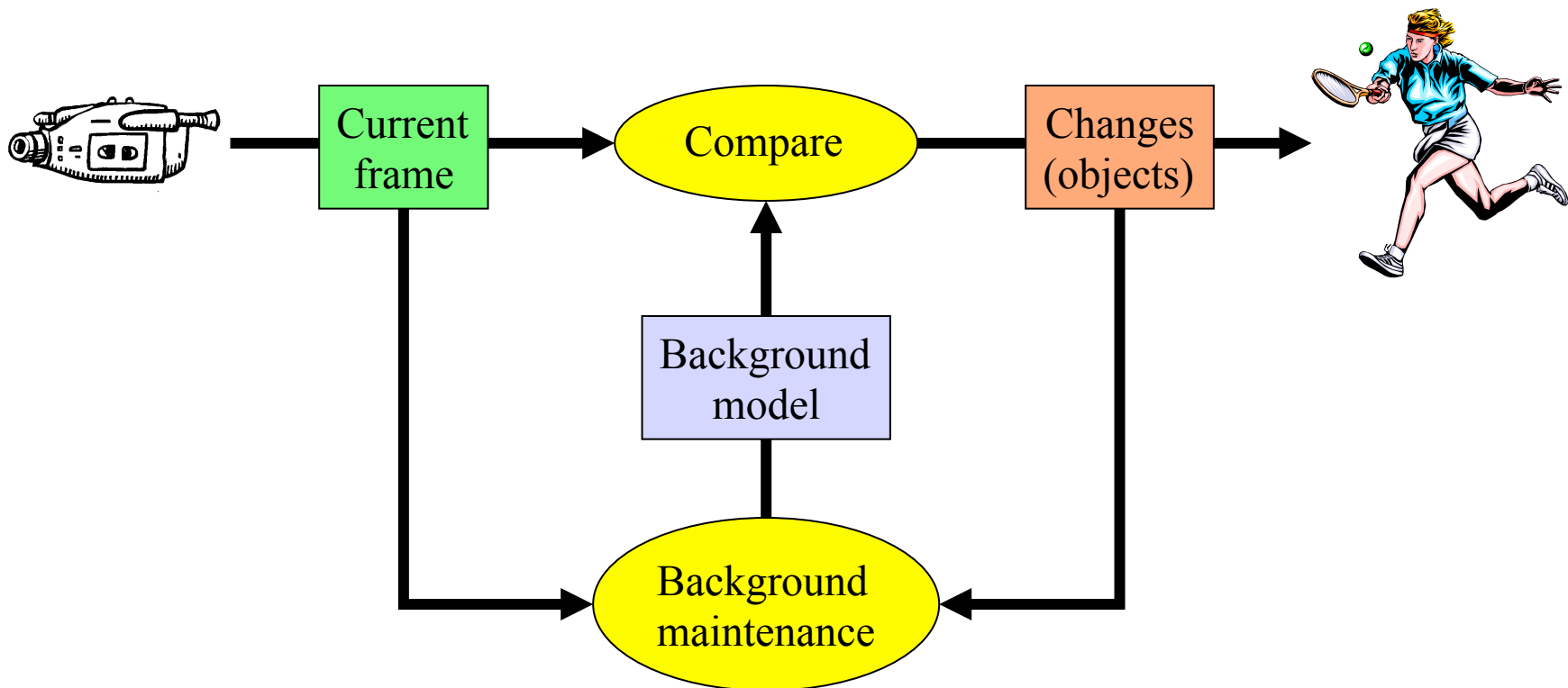
Frames come in 30 times per second. This is not much time to process each image. Real-time algorithms therefore tend to be very simple.

One of the main features of video imagery is the temporal consistency from frame to frame. Not much changes during $1/30$ of a second!

Detecting Moving Objects

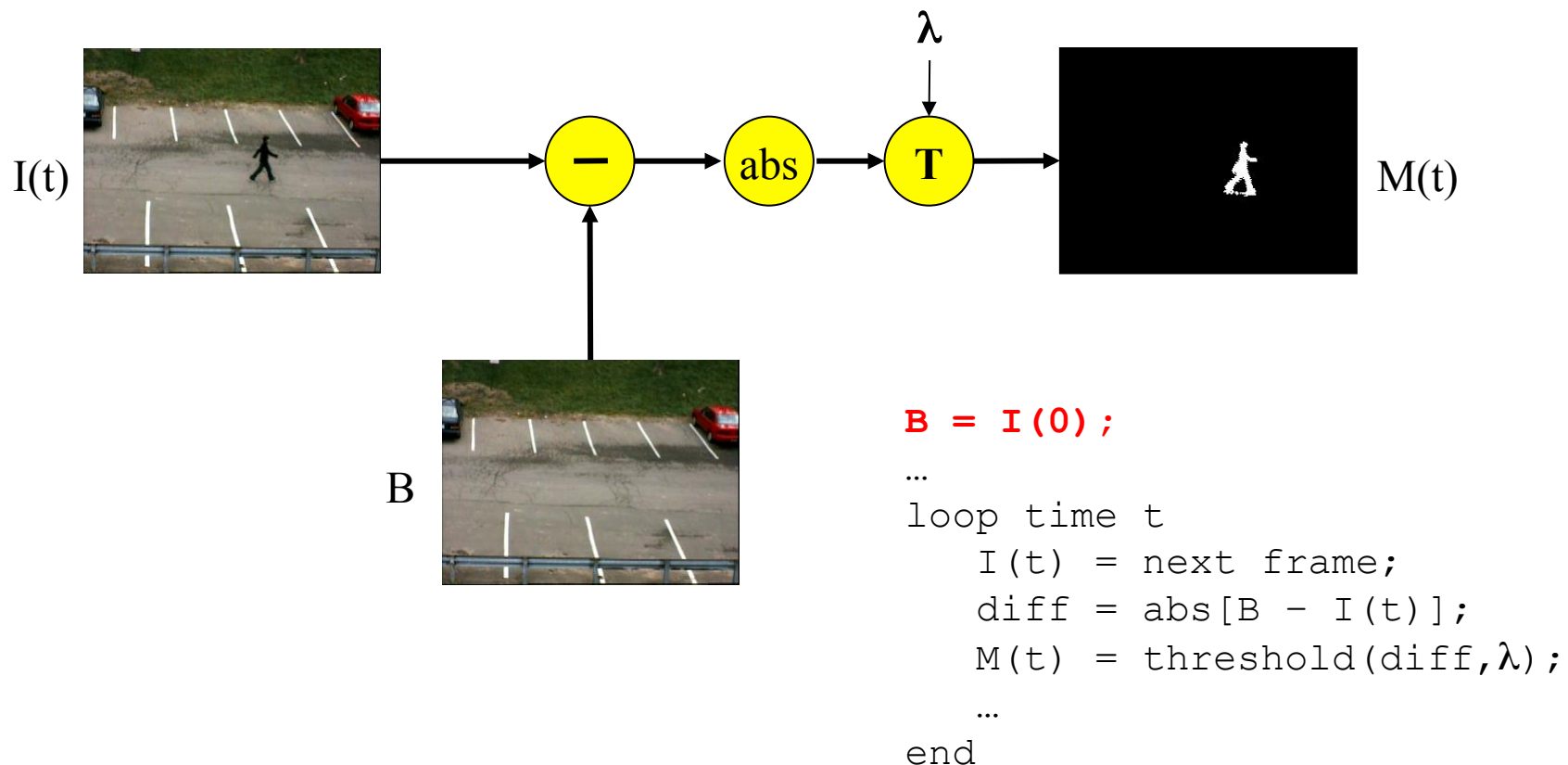
Assumption: objects that move are important (e.g. people and vehicles)

Basic approach: maintain a model of the static background. Compare the current frame with the background to locate moving foreground objects.

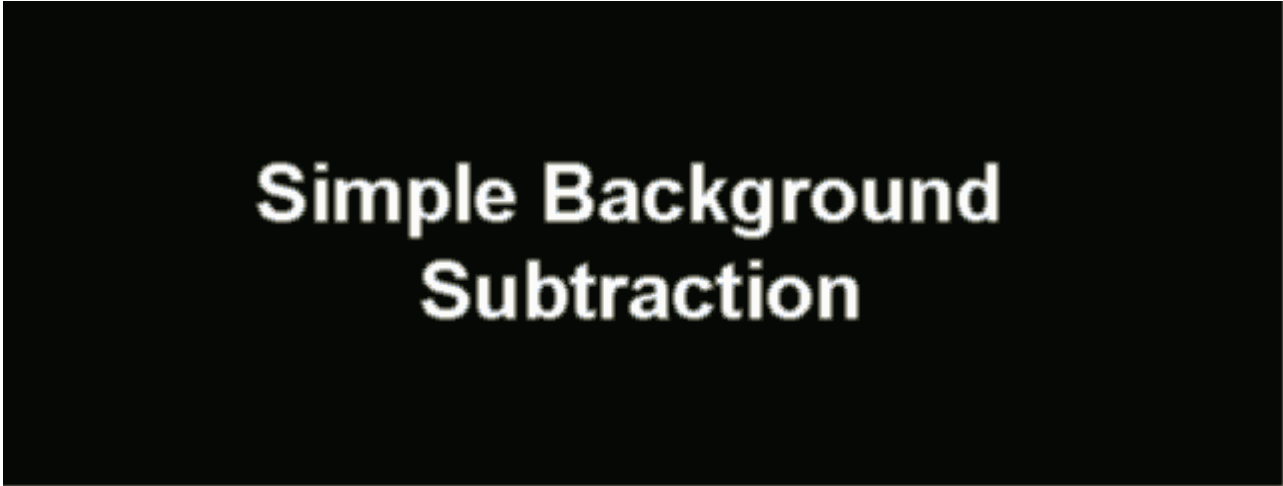


Simple Background Subtraction

- Background model is a static image (assumed to have no objects present).
- Pixels are labeled as object (1) or not object (0) based on thresholding the absolute intensity difference between current frame and background.



Background Subtraction Results



**Simple Background
Subtraction**

movie

BG Observations

Background subtraction does a reasonable job of extracting the shape of an object, provided the object intensity/color is sufficiently different from the background.



BG Observations



Objects that enter the scene and stop continue to be detected, making it difficult to detect new objects that pass in front of them.

If part of the assumed static background starts moving, both the object and its negative ghost (the revealed background) are detected



BG Observations



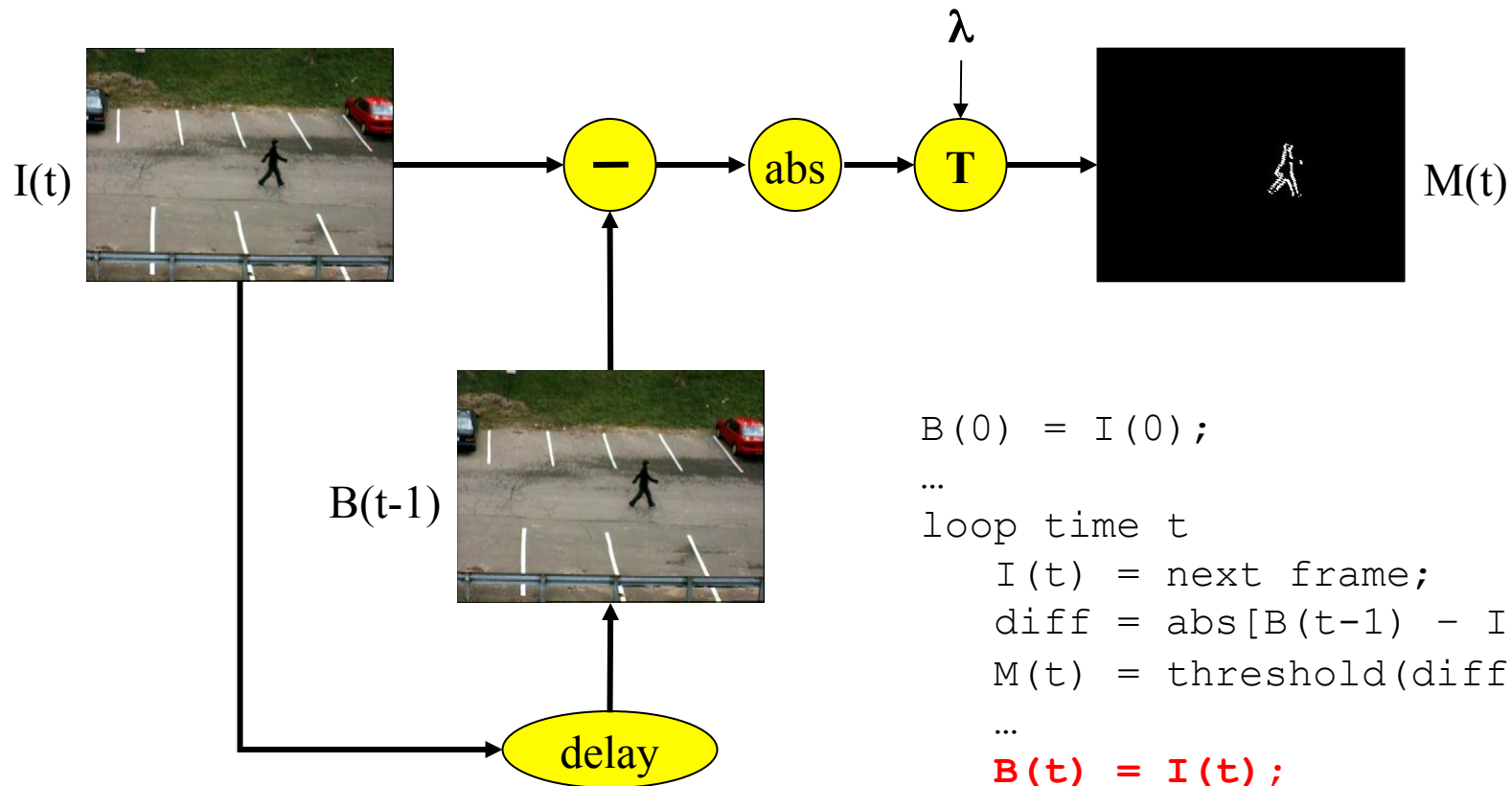
Background subtraction is sensitive to changing illumination and unimportant movement of the background (for example, trees blowing in the wind, reflections of sunlight off of cars or water).



Background subtraction cannot handle movement of the camera.


Simple Frame Differencing

- Background model is replaced with the previous image.



```
B(0) = I(0);  
...  
loop time t  
    I(t) = next frame;  
    diff = abs[B(t-1) - I(t)];  
    M(t) = threshold(diff, λ);  
    ...  
    B(t) = I(t);  
end
```

Frame Differencing Results



**Simple Frame
Differencing**

movie

FD Observations

Frame differencing is very quick to adapt to changes in lighting or camera motion.

Objects that stop are no longer detected. Objects that start up do not leave behind ghosts.

However, frame differencing only detects the leading and trailing edge of a uniformly colored object. As a result very few pixels on the object are labeled, and it is very hard to detect an object moving towards or away from the camera.



Differencing and Temporal Scale

Note what happens when we adjust the temporal scale (frame rate) at which we perform two-frame differencing ...

$$\text{Define } D(N) = \| I(t) - I(t+N) \|$$



$I(t)$



$D(-1)$



$D(-3)$



$D(-5)$



$D(-9)$



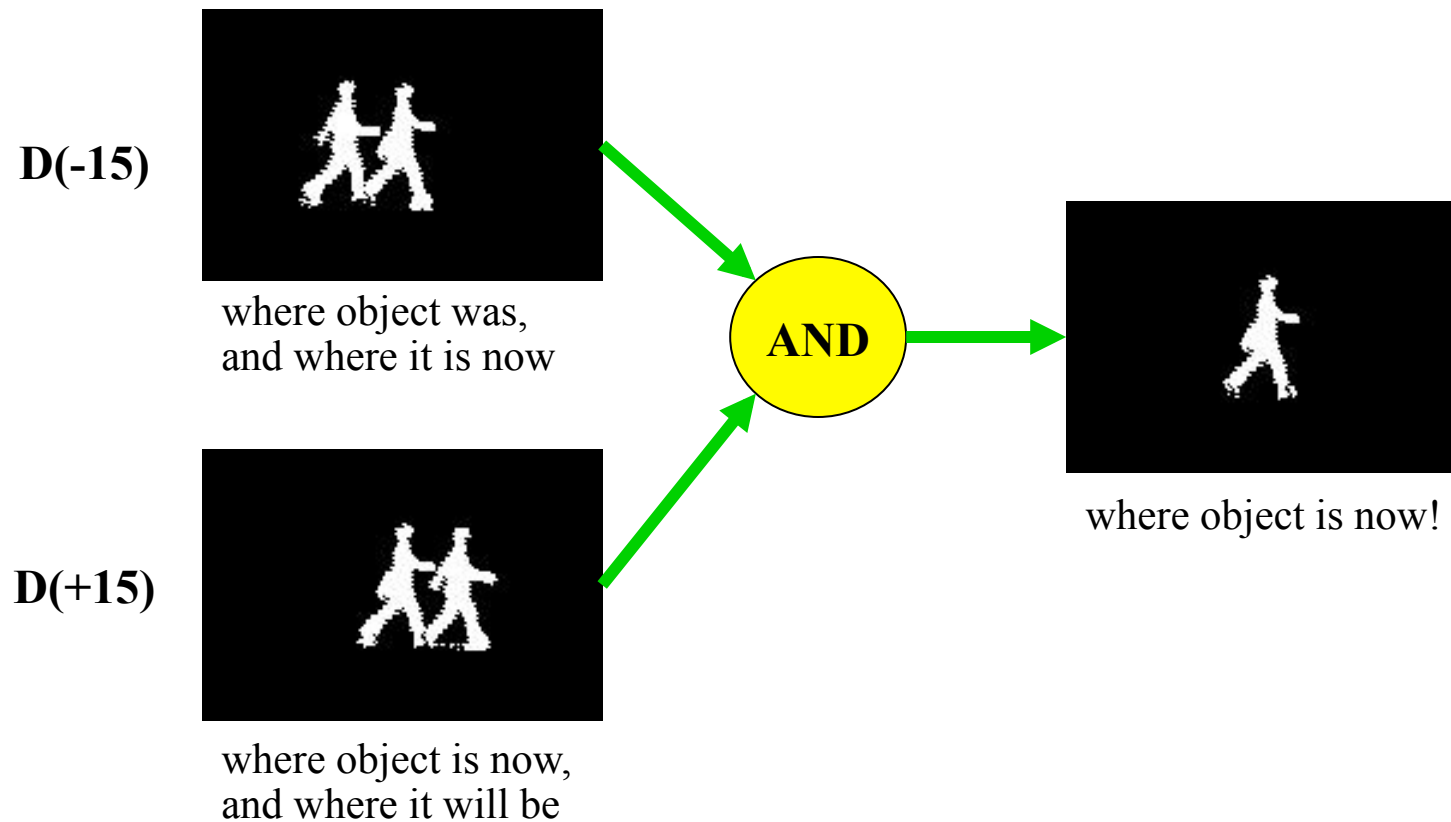
$D(-15)$



more complete object silhouette, but two copies
(one where object used to be, one where it is now).

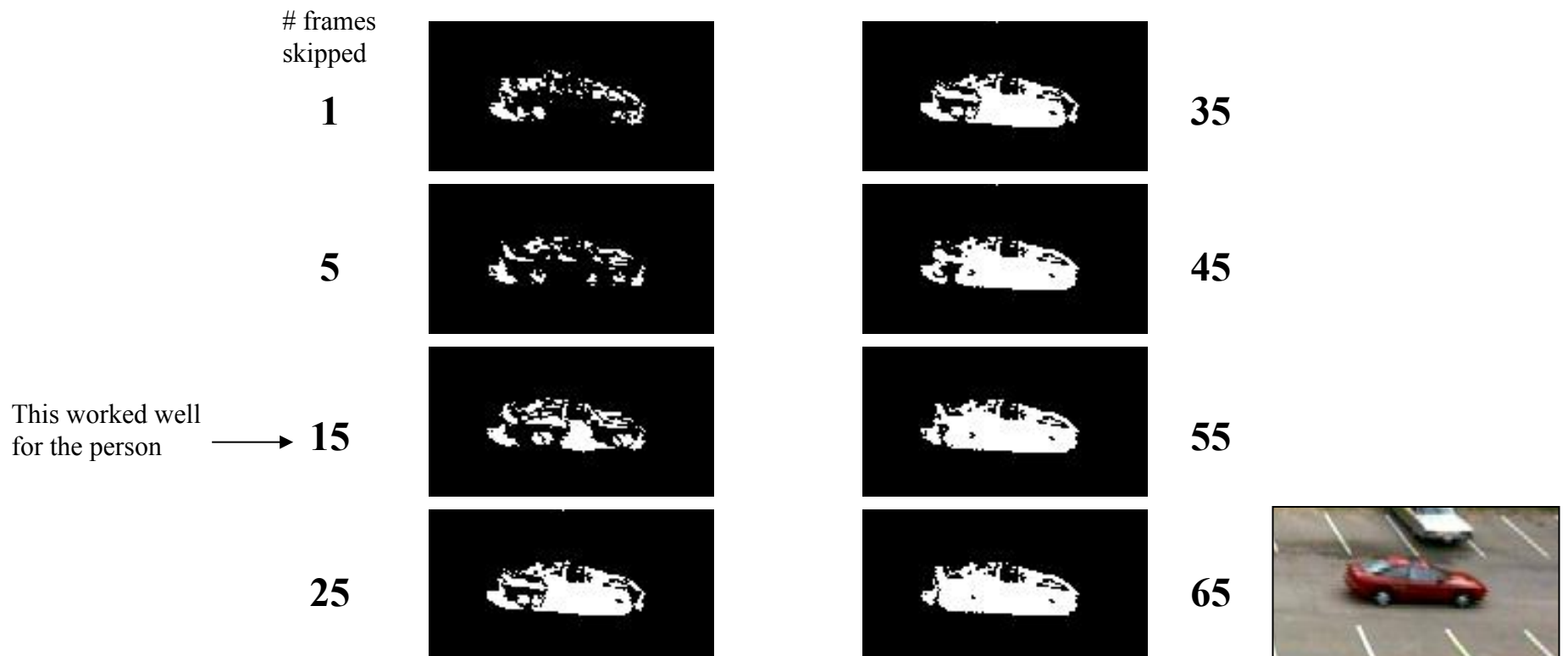
Three-Frame Differencing

The previous observation is the motivation behind three-frame differencing



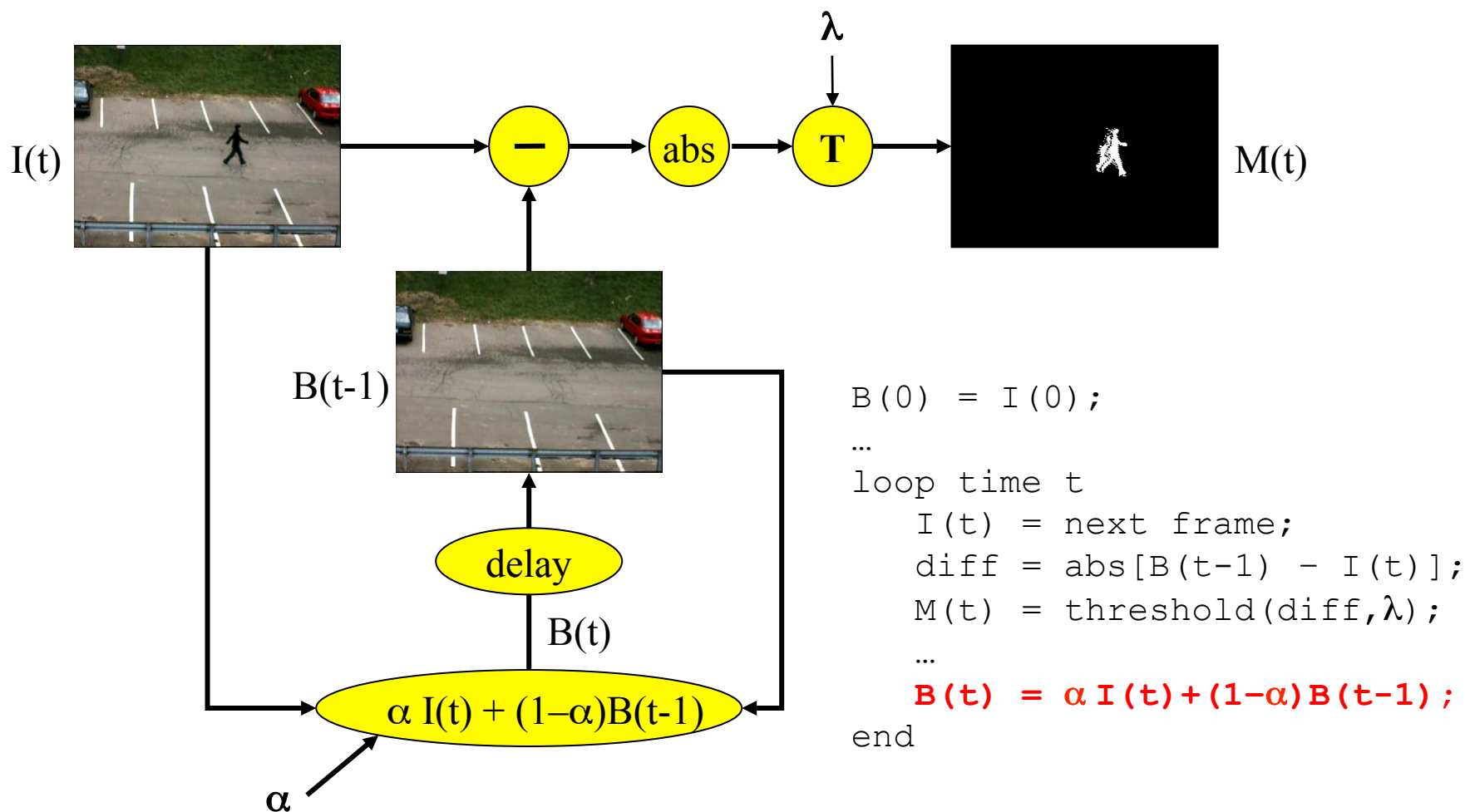
Three-Frame Differencing

Choice of good frame-rate for three-frame differencing depends on the size and speed of the object

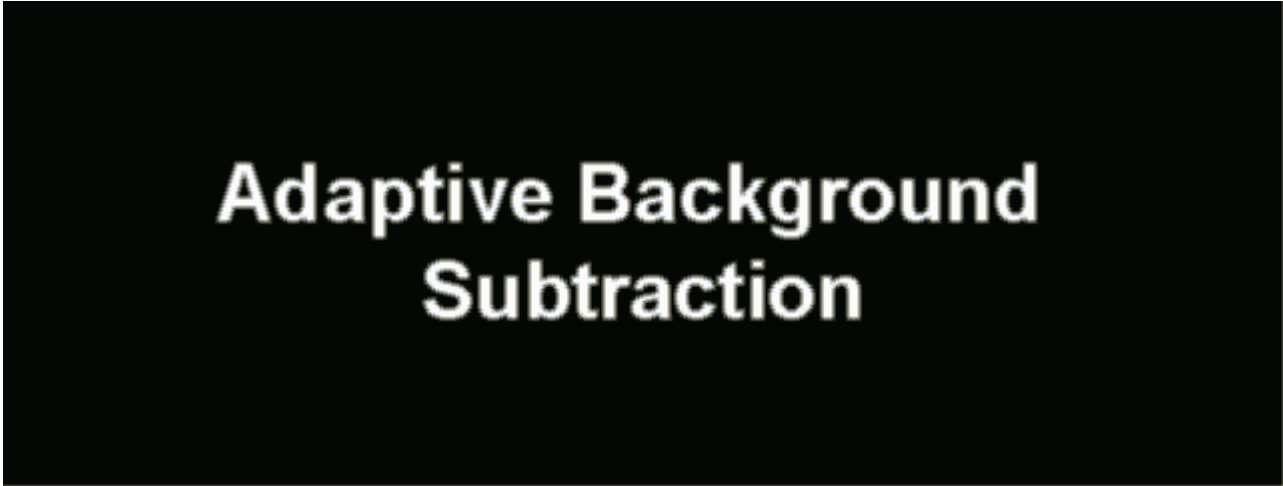


Adaptive Background Subtraction

- Current image is “blended” into the background model with parameter α
- $\alpha = 0$ yields simple background subtraction, $\alpha = 1$ yields frame differencing



Adaptive BG Subtraction Results



**Adaptive Background
Subtraction**

movie

Adaptive BG Observations

Adaptive background subtraction is more responsive to changes in illumination and camera motion.

Fast small moving objects are well segmented, but they leave behind short “trails” of pixels.

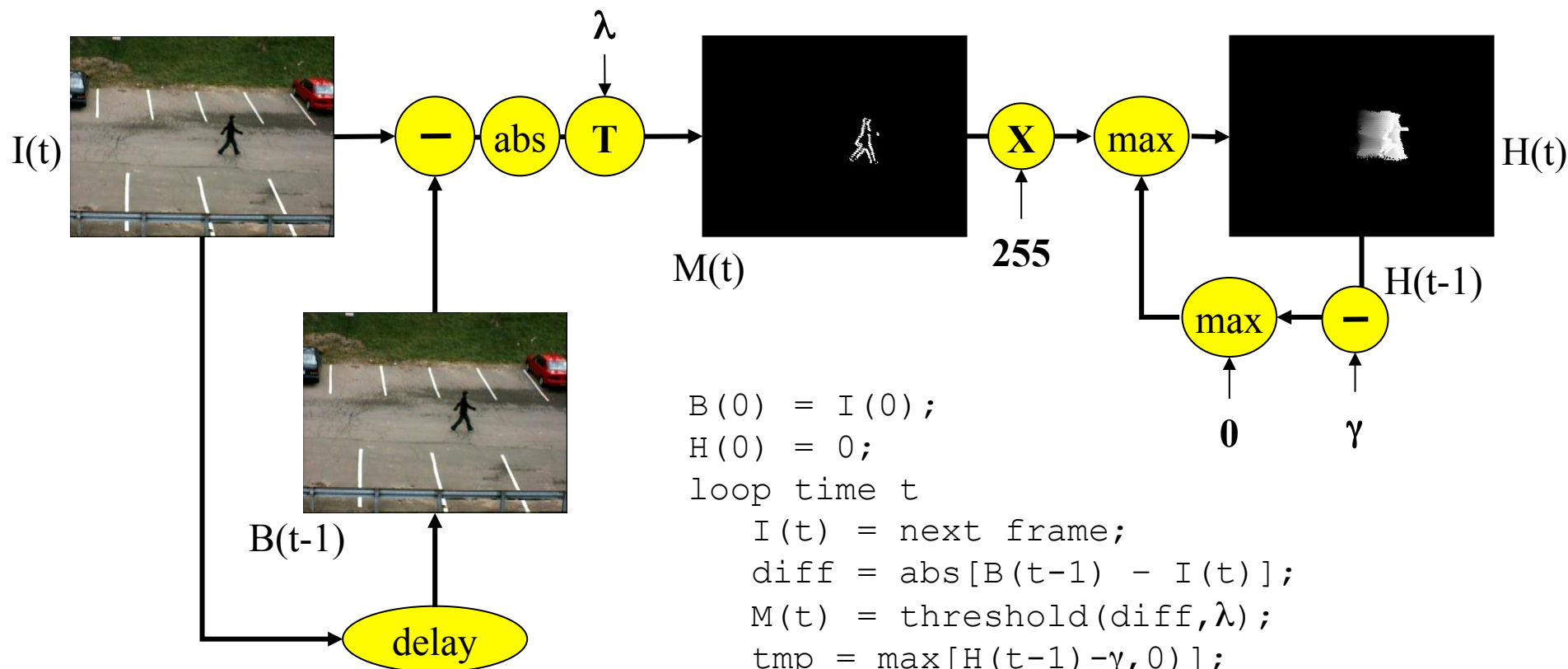
Objects that stop, and ghosts left behind by objects that start, gradually fade into the background.

The centers of large slow moving objects start to fade into the background too! This can be “fixed” by decreasing the blend parameter α , but then it takes longer for stopped/ghost objects to disappear.



Persistent Frame Differencing

- Motion images are combined with a linear decay term
- also known as motion history images (Davis and Bobick)




```

B(0) = I(0);
H(0) = 0;
loop time t
  I(t) = next frame;
  diff = abs[B(t-1) - I(t)];
  M(t) = threshold(diff,  $\lambda$ );
  tmp = max[H(t-1) -  $\gamma$ , 0];
  H(t) = max[255 * M(t), tmp];
  ...
  B(t) = I(t);
end

```

Persistant FD Results



**Persistent Frame
Differencing**

movie

Persistent FD Observations

Persistent frame differencing is also responsive to changes in illumination and camera motion, and stopped objects / ghosts also fade away.

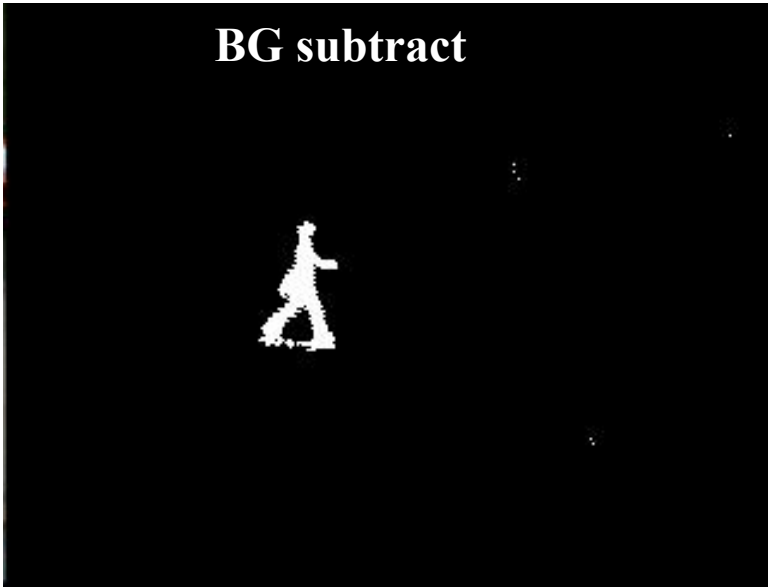
Objects leave behind gradually fading trails of pixels. The gradient of this trail indicates the apparent direction of object motion in the image.

Although the centers of uniformly colored objects are still not detected, the leading and trailing edges are made wider by the linear decay, so that perceptually (to a person) it is easier to see the whole object.

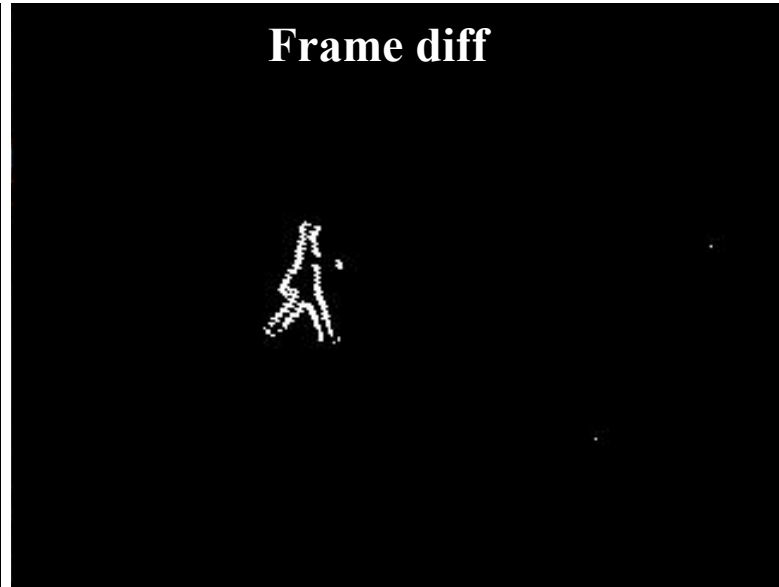


Comparisons

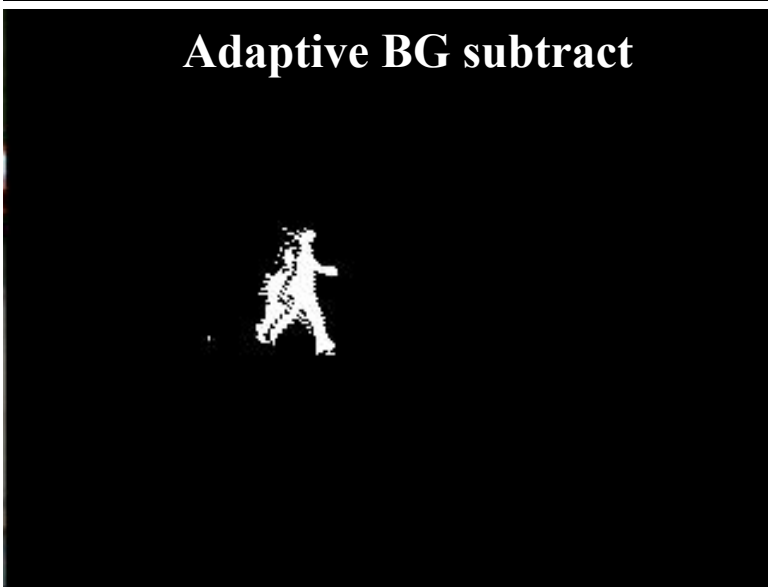
BG subtract



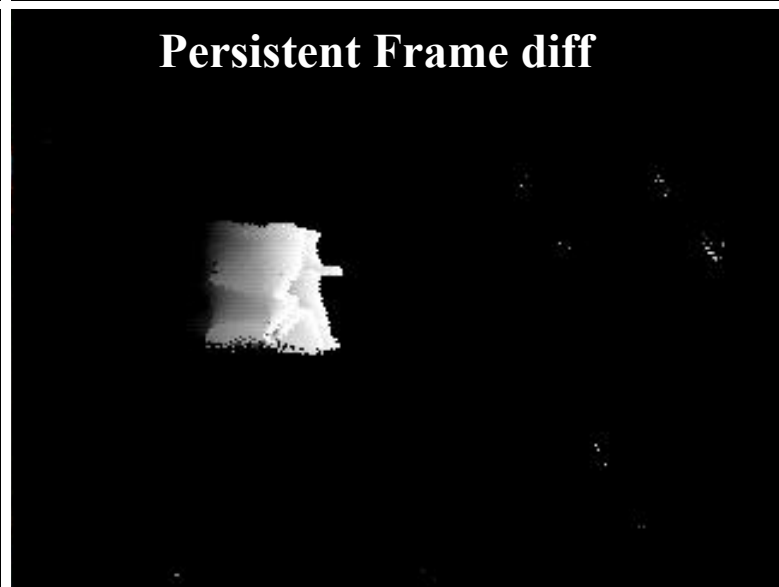
Frame diff



Adaptive BG subtract

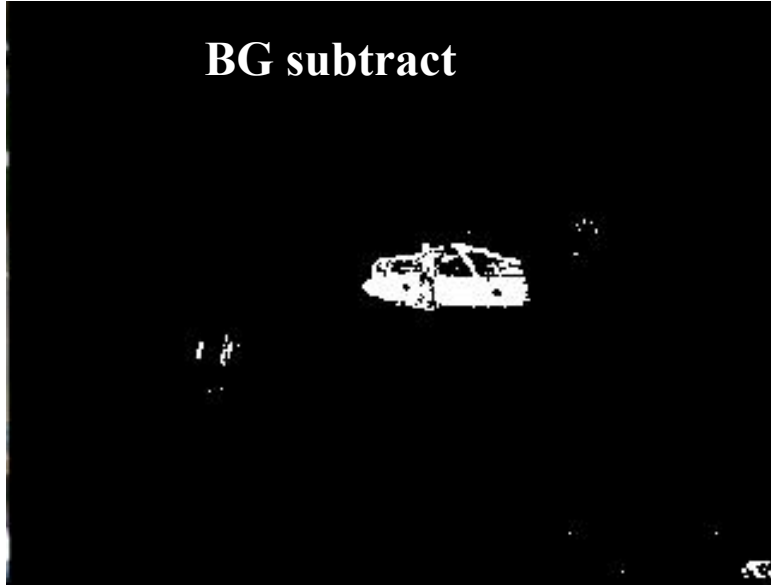


Persistent Frame diff

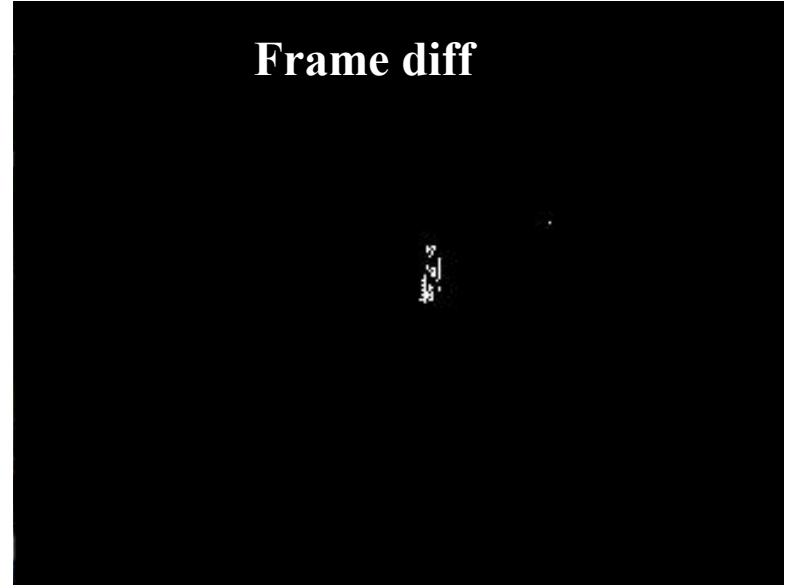


Comparisons

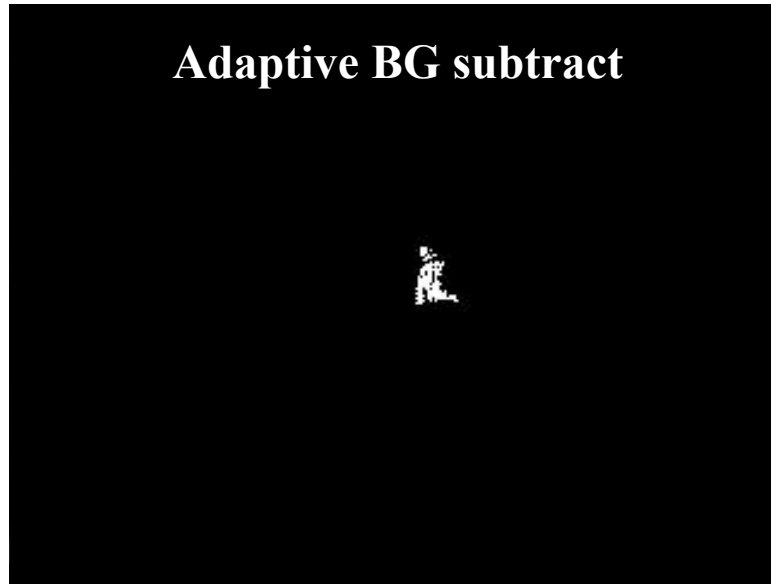
BG subtract



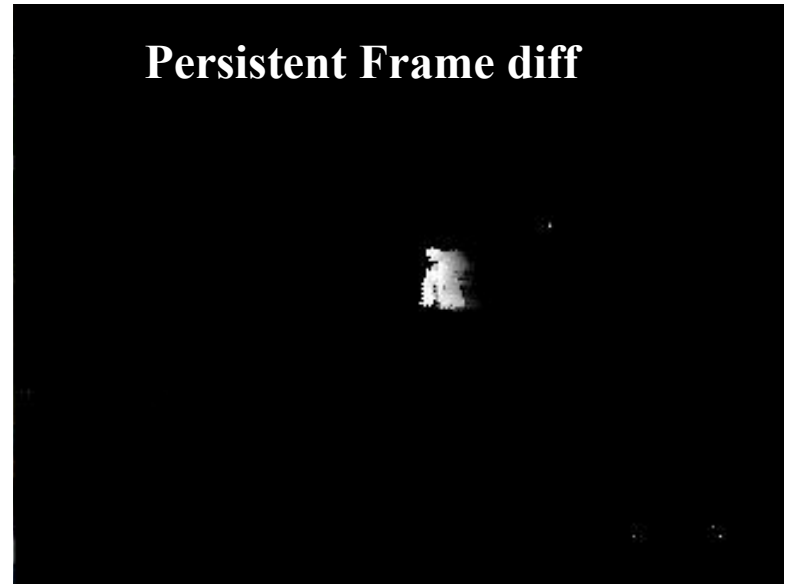
Frame diff



Adaptive BG subtract

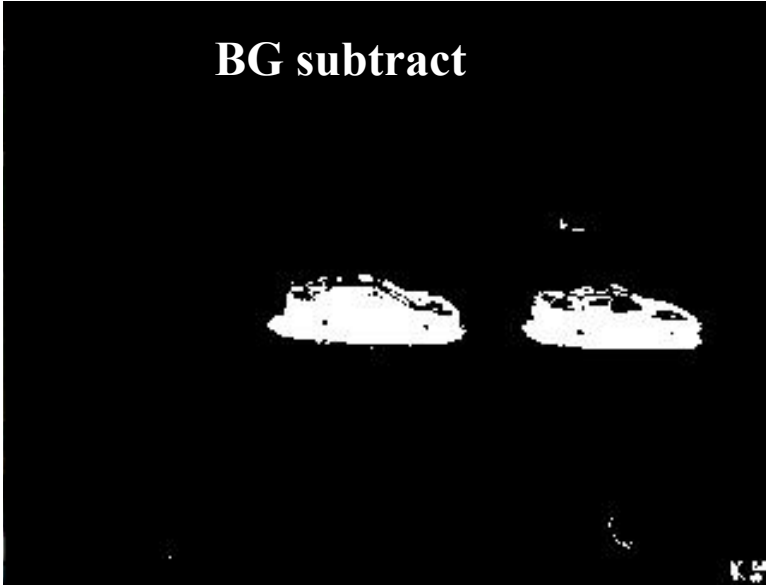


Persistent Frame diff

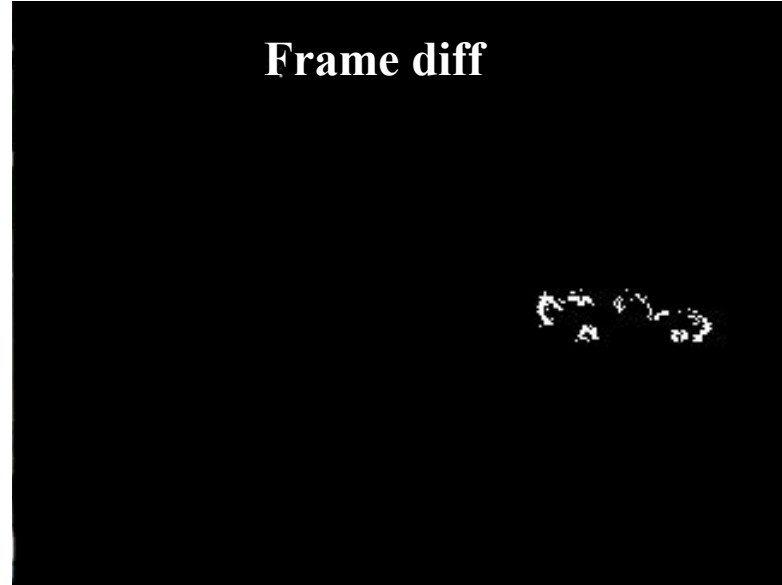


Comparisons

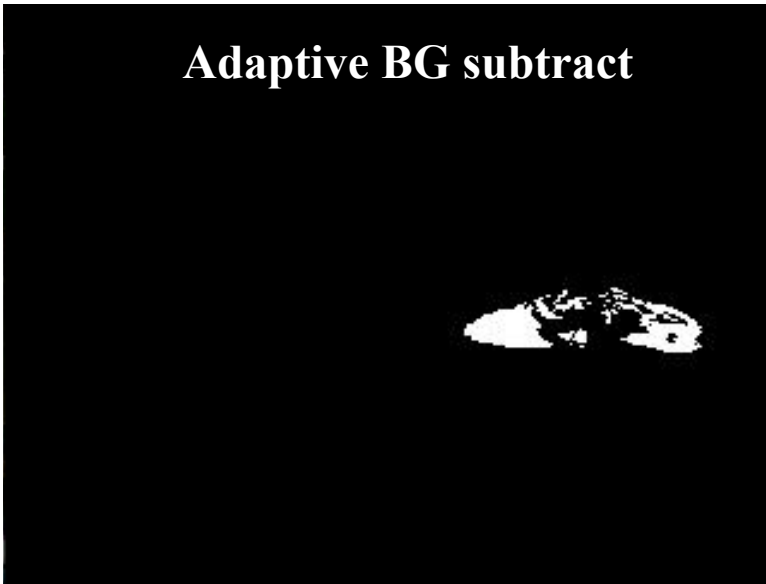
BG subtract



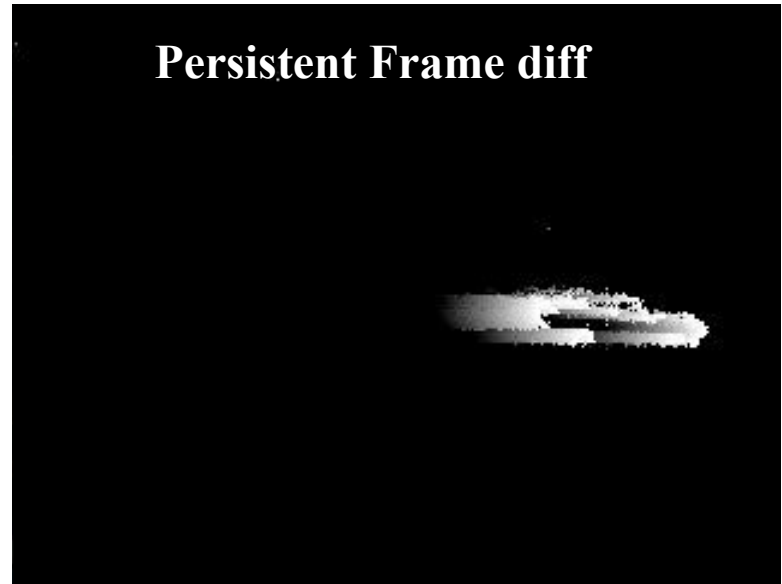
Frame diff



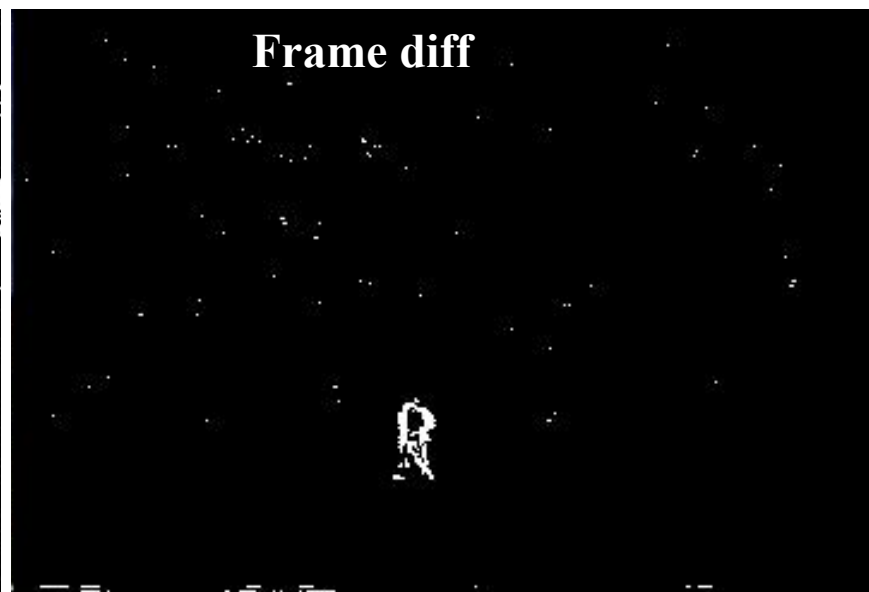
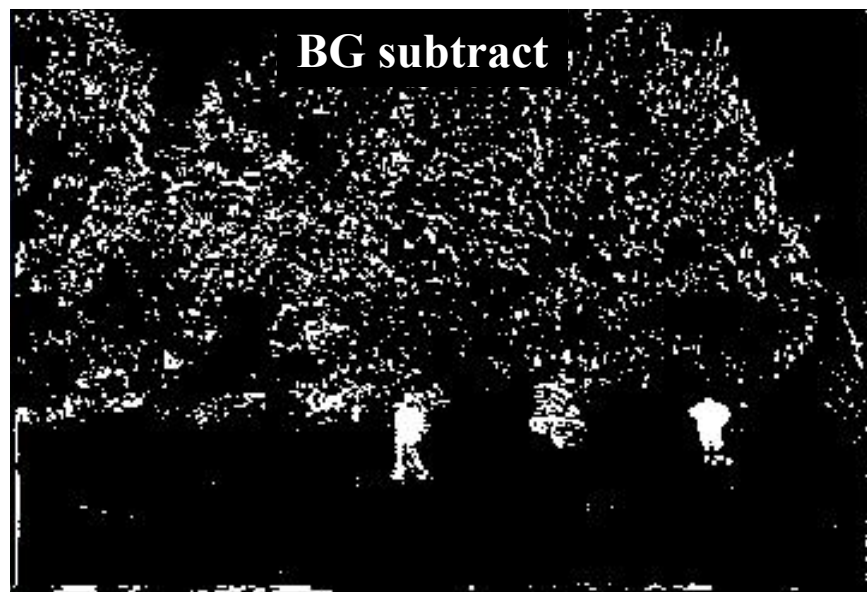
Adaptive BG subtract



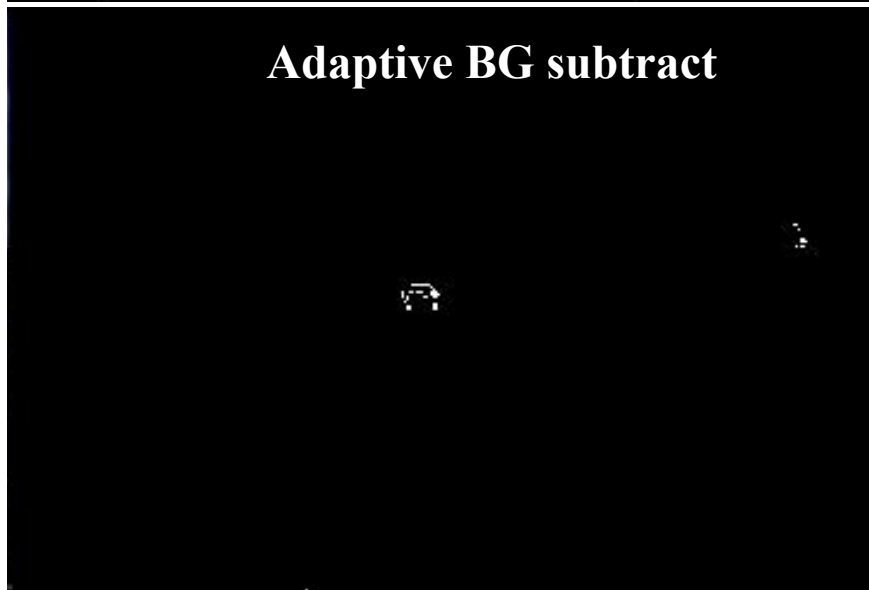
Persistent Frame diff



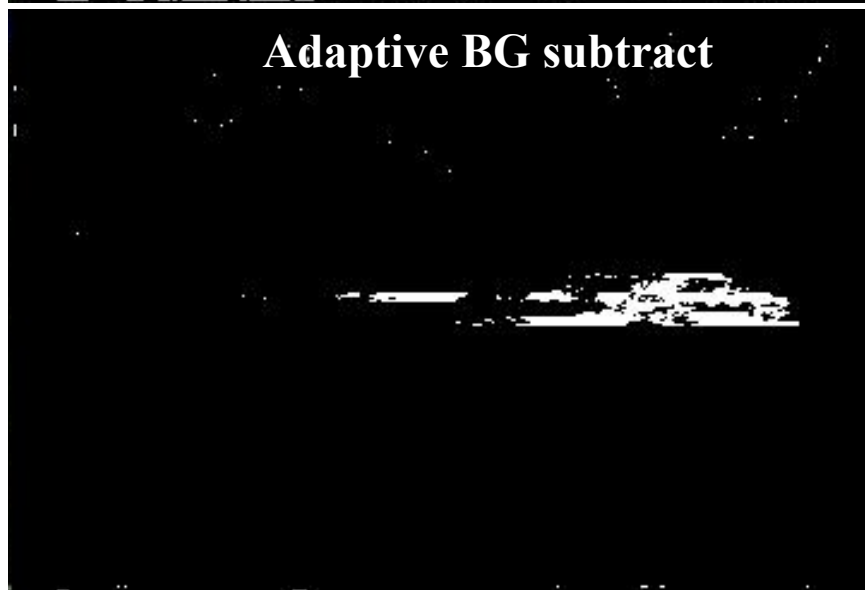
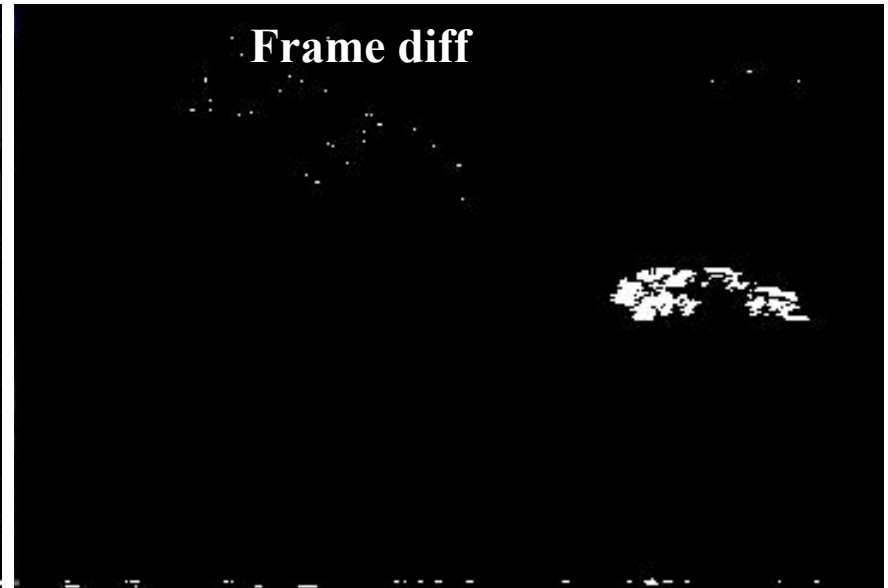
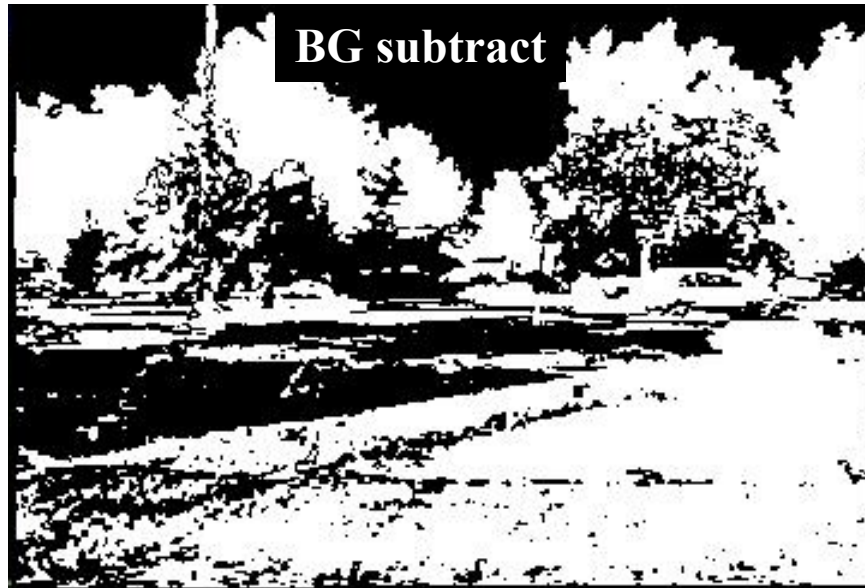
Comparisons



Comparisons



Comparisons



Project 3

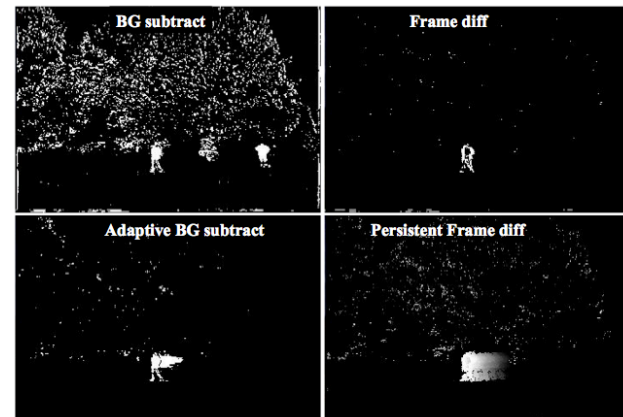
The goal of this project is to implement the four simple motion detection algorithms described in Lecture 20, run them on short videos, and compare the results.

As described (and pseudocoded) in Lecture 20, the four algorithms are:

- Simple Background Subtraction
- Simple Frame Differencing
- Adaptive Background Subtraction
- Persistent Frame Differencing

We'd like you to implement all four in one program, and generate as output a four panel frame showing the results of each of them, on each frame of video (see Figure 1), and generate a video of the results.

Figure 1: example output frame showing results of the four motion detection algorithms run on the same input sequence. Note: you don't have to label them with text, but do put them in the same order as shown here.



Project 3 (continued)

Detailed description

The steps of your program should be roughly the following:

- 1) For a given video sequence, read in each image in a loop and convert it to greyscale using whatever method you are familiar with (for example by taking the green channel).
- 2) After reading each image, compute a grayscale output frame using each of the four motion detection algorithms, and concatenate them together into a single four-panel frame. This last is easy to do in matlab, since images are arrays. For example, if the four motion detection results image arrays are A, B, C and D (you will want to make them more descriptive names), the output four-panel image can be generated as

`outimage = [A B; C D];`

Note that, since the persistent frame differencing image will have a different grayscale range (0-255) than the other three methods (0-1), when generating the output four-panel image you want to convert it to the range 0-1 by dividing by 255, so all output brightness ranges are comparable. (alternatively you could scale up the intensity range of the other three by multiplying by 255).

Project 3 (continued)

- 3) Generate a numbered filename for your output image (e.g. if you are currently processing frame 0036 of the input sequence your output image will be numbered 0036) and save the image, either as a jpg or png. If you save as a png you will have better resolution in the final movie, since it does not compress the image while writing.
- 4) Continue the loop until all input images have been processed.
- 5) Generate a video of the results. How you do this is totally up to you. One program I like is “ffmpeg”, which is open-source software with versions that work on windows, linux or mac. Or you could use windows moviemaker; or apple iMovie, or whatever you want. It's even possible to generate movies from matlab, if you prefer to do that. Make your movies some obviously playable format, like mp4. If we can't read what you produce we'll have to ask you to regenerate them, which will be a pain for both us and you.

Pro Tip: Although you are running all four algorithms at the same time, it is a good idea to keep their data structures independent from each other. For example, in the lecture pseudocode, variable B or B(t) is used to denote the current background frame. However, this image will be different for each algorithm, so you really should be keeping four current background frames, one computed by each algorithm.

Project 3 (continued)

Datasets

We've uploaded a DataSets.zip file containing several sequences to test on. Five of them are the test sequences that were used in Lecture 20. These are included so that you can compare your results against ours, to see if your algorithms are working correctly.

Also included are three new datasets you haven't seen before, called ArenaA, ArenaN and ArenaW¹. We want you to submit results videos for each of these.

What to hand in

In addition to the three results videos for the Arena datasets and your runnable code, the other half of your grade is based on a written report discussing your program, design decisions, and experimental observations. The report should include at least the following:

a) Summarize *in your own words* what you think the project was about. What were the tasks you performed; what did you expect to achieve?

¹Video data courtesy of <https://motchallenge.net/data/PETS2017/>

Project 3 (continued)

b) Present an outline of the procedural approach along with a flowchart showing the flow of control and subroutine structure of your Matlab code. Explain any design decisions you had to make. In particular, did you keep the four motion detection algorithms completely independent from each other, or did you try to use the results for some of them when computing others? However you do it is up to you, but explain what you did.

c) Experimental observations. All methods have user-settable thresholds and parameters. One is the pixel difference threshold for determining what level of intensity change is needed to declare there has been a significant change at a pixel, not just sensor noise. What value did you decide to set it at, and how did you decide? What do you observe when you set this higher? What do you observe when you set it lower? An important parameter for adaptive background subtraction is the “alpha” parameter for merging new images into the background model. Try different values for this, and explain what you observed as you change it higher and lower. Can you verify that setting it to 0 and 1 yields results very similar to two of the other algorithms? Also note the linear decay parameter in the persistent frame differencing algorithm. How did you set that, and what happens if you make it larger or smaller?

Project 3 (continued)

d) Experimental Comparisons. Compare the results of all four algorithms on the three Arena videos (which should be easy to do since they are conveniently displayed as four panels in your output video). Do you notice any significant differences between them? Are there some situations where one or more of them produce terrible results? Identify some events in the videos where the algorithms produce very different quality results, and explain why the different algorithms are either succeeding or failing on these events (you can paste pictures in your report to illustrate this). Overall, which method would you prefer to use, if you were only able to choose one, and justify your answer (there is no right or wrong answer to this, as long as you have reasonable and sufficient justification).

e) Document what each team member did to contribute to the project. It is OK if you divide up the labor into different tasks. This is also where you let us know if any of your teammates were slacking off on this project.

Optional Additions

There are no extra credit additions for this project.