Analyzing an Intersection with Motion Detection from a Fixed Camera

I. Goal

The objective of this project was to take stock footage that viewed a 4-way intersection with stop signs from a top-down view and analyze it to determine whether a stop light should be put in based on the amount of turns over time. Left turns specifically are necessary in the calculation of this so the recognition of left turns in particular was necessary.

Stock:

Notice, this is a special intersection. Cars only go from left to right because it is a one way intersection. 2 different types of each turns can still be made. Left turns into and out of the parking lot, and right turns into and out of the building from which the footage was taken.
Notice the pedestrians. Any sort of motion detection should feasibly detect them very well, and it’s important that while implementing this not to detect the pedestrians, even if they mimic the path of a turning vehicle. The overhead view is also designed to prevent occlusion, but on more than one occasion a larger vehicle (i.e. a bus or full sized van) partly obscured the view of another vehicle on the screen.

With normal-vehicles, occlusion is avoided due to the height of the vantage point.

Busses can frequently merge with or block out entirely other vehicles and pedestrians.
II: Methods

Motion Detection was accomplished with a frame-by-frame comparison of stills. The frame used to compare had no moving vehicles on it, but the cars can still be seen above the road, so a bag-of-words comparison would probably not be the best in this situation, as it would detect all of the above cars as well as any moving cars. The method decided on was to ignore the cars above by using binary images.

On the first try to create a binary image by just comparing to one frame and a certain cutoff point at which a difference in the value of a pixel indicated motion, most of the screen was deemed to be moving. It was determined that this is the result of the equipment used. The camera used to record the video has very poor quality, and the pixels values change frequently from still to still, even if nothing is on the screen.

Therefore, it was then necessary to create a matrix of per-pixel standard deviation for comparison. 30 different stills of nothing moving on the screen were analyzed and the highs and lows of each pixel were recorded. The value of the range between high and low divided by two was set as the cutoff for the difference to decide if a pixel was part of a moving object or not. From this vantage point, as can be seen above there were no trees with leaves or anything that would move continually, so this algorithm should be fairly effective in this case to work at eliminating background for a binary image.

A snippet of code used for calculating the standard deviation matrix:

```matlab
for r = 1:h3
    for s = 1:w3/3
        if image(r,s,1) ~= blank(r,s,1)
            q=q + abs(image(r,s,1) - blank(r,s,1)); //red
        end
        if image(r,s,2) ~= blank(r,s,2)
            q=q + abs(image(r,s,2) - blank(r,s,2)); //green
        end
        if image(r,s,3) ~= blank(r,s,3)
            q=q + abs(image(r,s,3) - blank(r,s,3)); //blue
```
end
if q>std_dev(r,s)//q is total pixel difference
    std_dev(r,s)=q;
end
q=0;
end

std_dev was set as a matrix of zeroes so any difference would have been recorded. A very similar method was used to compare the actual images for the construction of the binary image. Instead of resetting the std_dev variable, another matrix of zeroes was used and if the value was above that of standard deviation, and it was set to a 1. This, read as a binary image, would produce a binary image with anything radically different as white and everything else as white.
As is obvious, the camera’s values were very inordinate on the white parts visible other than the moving van and pedestrians.

This would prevent a problem for the process of tracking a moving object with binary images: along with any pedestrians, which stand out in the picture, some of the more complex surfaces in the picture turned up as moving permanently. To counteract this, the blob detection of the binary image had to be modified. White regions (decided by another cutoff value, at least .5 of the pixels need to be white) with area of 40*40 were inspected and it was determined if they were next to more of these regions. This exhaustive search with arbitrary values worked well with a little bit of tweaking of said cutoff values. The results can be seen below:

![Binary Image Of A Random Frame](image)

Here, it can be seen that the algorithm picked up the heavy concentration of white on the van and ignored the pedestrians, edges, and crosswalk.
Using this matrices, from this point the sequences of 1’s would be analyzed every few (5-10) frames to determine the turn being made. Obviously going straight in either was very easy to track, being a sequence of 1’s moving left to right across a row or up a column, but tracking and detecting turns proved to be harder. The differences in car size and turning styles (some take the corner tight, others loose) resulted in a wide discrepancy of sequences in changes to the matrix during a turn sequence. It also was apparent that based on this model the algorithm would likely fail to be relevant at any other intersection, or from any angle.

III: Results

The project failed to yield any full results in the form of video, as it was still in the stages trying to detect turns when work stopped on it. The possible applications of the software that was developed are as follows:

Traffic Lights: Image analysis to determine what sequence of stop lights to use, and real time light-changing based on the number of cars around the stop light. Already widely in used.

Traffic tracking: A helicopter over a highway (or a stationary camera on the highway) could remain stationary for a few seconds over a given area to determine how fast traffic is moving on a given road. This method would be more automated than the current model, which involves people in the chopper reporting back on whether certain areas are congested.