Computer Vision I

Project 4
Target Tracking

Given November 19 , 2018 Due: December 12, 2018

The Circulant Matrix (CM) tracker that we discussed in class is very efficient finding a translated copy of the target template (from the previous frame) by computing many convolutions in a single shot. This is accomplished by finding the peak response of a filter applied to a region of the current frame that is expected to include the target. This filter changes from frame to frame and it is computed based on the FFT of a larger region which contains the target in the current frame. (Efficiency is obtained by applying this filter in the frequency domain). The papers describing the algorithm and the code for this tracker is available at:

`http://www.robots.ox.ac.uk/~joao/#publications`

and

`http://www.robots.ox.ac.uk/~joao/circulant/`

The goal of this project is to improve the Circulant-Matrix tracker by:

1. Implementing an occlusion detection test;
2. Trying to recover from occlusion.

Detecting Occlusion The code provided by the authors of the CM tracker does not check for occlusion. You can do this by including a test that measures the response of the filter against the rest of the search window through the use of the "Peak to Sidelobe Ratio" (PSR). To do this, you should split the response of the filter into the maximum value and the "sidelobe" consisting of the rest of the pixels in the region, excluding a small window (i.e. 11×11) around the peak. Then the PSR is defined as:

$$\frac{g_{max} - \mu}{\sigma}$$

where g_{max} is the value of the peak, and μ and σ are the mean value and the standard deviation of the sidelobe, respectively. A low PSR indicates a poor match and a possible occlusion. If occlusion is detected, the tracker should stop or attempt to hallucinate the target until it can detect it again.

Recovery from Occlusion When occlusion is detected, your tracker could use the locations of the target in the past to predict where the target is now (even if it is behind some occluding

object) and predict where it should be in the next frame. In this way, rather than giving up, the tracker can attempt to find the target in the next frame at this predicted location. The prediction of the location of the target based on previous measurements can be done using one of many possible methods that we will discuss in class. Among them, you should consider using the Hankel matrix of the target locations or a simple dynamic model such as constant velocity.

1. Implement the above extensions.
2. Use your program with the video sequences from:
   `http://vision.ucsd.edu/~bbabenko/project_miltrack.html`.
3. Use the show_precision function to compare your results against the original CM tracker and the original Multiple Instance Learning (MIL) tracker (the results for the MIL tracker are available with the testing data). Add a visualization to indicate when you detected occlusion and when you recovered the target.
4. The output of your program should be a file with the image coordinates of the center of the bounding box, its width and height, for each frame or an indicator of occlusion. Print out a few key frames (4 or so) showing the bounding boxes generated by MIL, CM and your tracker.
5. Write a report discussing your results and showing the sample images.