OpenCV Object Tracking Notes

Colin Burdine

1 Existing Methods

1.1 Minimum Output Sum of Squares Error (MOSSE) Filters [1]

- *Filter-Based* trackers work by initially selecting a target that is centered within a trackign window in the first frame.
- The target is then tracked by correlating the filter over a search window in the next frame, and the location that maximizes the correlation is considered the location in the next frame.
- The principle virtue of the MOSSE Filter is that it is fast, and can perform real-time tracking at high frame rates, making it suitable for edge computing purposes.
- The correlation is computed using the *Fast Fourier Transform* (FFT; denoted by $\mathcal{F}$). The correlation σ is given by the element-wise complex inner product of the filter matrix h and the frame image f:

$$\sigma = \mathcal{F}^{-1}(F \odot H^*) \quad \text{where} \quad F = \mathcal{F}(f),\ H = \mathcal{F}(h)$$

- The most computationally expensive operation in this process is the 2D FFT and inverse 2D FFT, which can be performed in $\mathcal{O}(n^2 \log(n))$ for an $n \times n$ filter.
- We shall break down the operation of MOSSE filters into some critical sections:

Preprocessing

- For simplicity, the MOSSE algorithm first reduces the image to a log grayscale. Then, the image is normalized to have pixel intensity values with mean 0 and std. dev. 1.
- Since the 2D FFT projects a rectangular region onto a torus shape, connecting the top with the bottom and the left side with the right side, we want to avoid a filter "wrapping around" the edges of the image. This is remedied by multiplying the image by a cosine window.

Applying the MOSSE Filter

Ideally, a desired MOSSE filter H satisfies the condition $G = F \odot H^*$, however, we can find the filter H that is closest to this equality in the L^2 sense by solving the optimization problem that minimizes the residue.

$$\min_{H^*} \sum_i |F_i \odot H_i^* - G_i|^2$$

Since the set of possible filters forms a Hilbert space, the solution (given by the Hilbert projection Theorem) is simply the projection of G onto the basis of F, given by:

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^*}$$

Regularization

- A Mosse filter can fit a single image (e.g. f_1) perfectly, which is not desirable in general. Thus, we use the average of the filter trained across all of the images to produce a filter that generalizes better. This averaging is motivated by the idea of *Bootstrap Aggregation*:

$$H^* = \frac{1}{N} \sum_i \frac{G_i \odot F_i^*}{F_i \odot F_i^*}$$

- MOSSE filters can sometimes be unstable for a low number of images; however, we can apply regularization by replacing the denominator $(F_i \odot F_i^*)$ with $(F_i \odot F_i^*) + (\epsilon \mathbf{1})$, where ϵ is the regularization parameter and $\mathbf{1}$ is the matrix of ones.
- (Note: Figure 4 in [1] seems to indicate that regularization is not necessary, though it is unclear what the number of images N is.)

Initializing and Updating the MOSSE Filter

- The MOSSE filters are initialized as described in the previous section, with an initial value of ϵ. According to Figure 4 in [1], $\epsilon = 0.1$ seems to be a good choice.
- To update the filter as tracking progresses, a moving average with "forgetfulness" parameter $\eta \in [0, 1]$ is used in both the numerator and denominator. This results in the formula:

$$H_i^* = \frac{A_i}{B_i}, \qquad \text{where} \qquad A_i = \eta(G_i \odot F_i^*) + (1 - \eta)A_{i-1},\ B_i = \eta(F_i \odot F_i^*) + (1 - \eta)B_{i-1}$$

- [1] recommends selecting $\eta = 0.125$.

Failure Detection

- In order to detect if a tracked object has been lost, we determine a threshold value for the *Peak to Sidelobe Ratio* (PSR). The PSR is computed for a correlation output G as

$$PSR(G) = \frac{g_{max} - \mu_{sl}}{\sigma_{sl}}$$

where g_{max} is the point where the correlation is maximized (the object's estimated location), and (μ_{sl}, σ_{sl}) are the mean and std. dev. of the sidelobe region, which is the region of all pixels excluding those within some window centered about g_{max}. Typically, this window size is 11×11 pixels.

- [1] recommends that when the PSR drops below the threshold of 7.0, the object is lost.

1.2 Generic Object Tracking Using Regression Networks (GO-TURN)

1.3 MedianFlow Tracking

1.4 Discriminative Correlation Filters with Channel and Spatial Reliability (CSRT)

1.5 Kernelized Correlation Filters (KCF)

References

[1] David Bolme, J. Beveridge, Bruce Draper, and Yui Lui. Visual object tracking using adaptive correlation filters. pages 2544–2550, 06 2010.