CSci 5561 Assignment 2

Luis Guzman

Friday October 16, 2020

In this assignment, I used SIFT Feature extraction, RANSAC, and inverse compositional image alignment to track a face through multiple frames of a video. The method in this assignment consisted of using a template face image, and aligning this image with a larger target in order to track the face. In order to extract features from the target and template images, I used OpenCV's implementation of the SIFT algorithm with the contrastThreshold parameter set to 0.02. I also used the Lowe's ratio test to determine which features were good candidates for passing to my alignment algorithm. The criterion is:

$$\frac{||\lambda_1 - \lambda_2||}{||\lambda_2 - \lambda_i||} > 0.7 \; \forall \; i$$

where λ_i is a vector containing the local (16x16) pixel information around the keypoint. We're essintially comparing the how similar each detected keypoint is, and making sure that the best match isn't also similar to another keypoint.

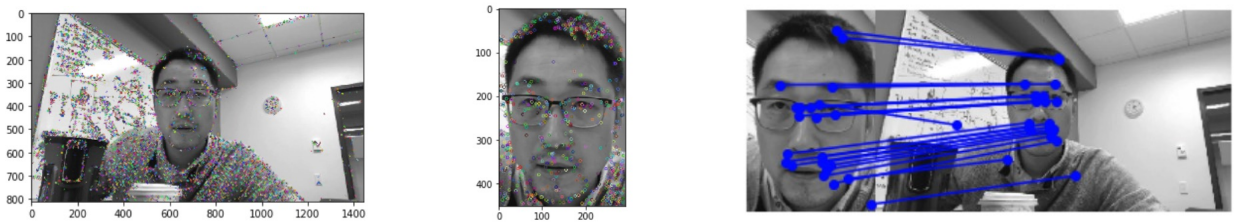

Figure 1: Images showing the SIFT feature extraction and feature matching of the target and template images. Mismatched features were filtered out using RANSAC

Once we have our keypoints, I used RANSAC to iterate through them and generate an affine transform for each set of 3 points. The affine transform can be calculated using least squares from

$$\boldsymbol{x'} = \boldsymbol{X}\boldsymbol{a}$$

$$\begin{pmatrix} x'_1 \\ y'_1 \\ x'_2 \\ y'_2 \\ x'_3 \\ y'_3 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_1 & y_1 & 1 \\ x_2 & y_2 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_2 & y_2 & 1 \\ x_3 & y_3 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_3 & y_3 & 1 \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{pmatrix}$$

where x_i, y_i are the target points, x'_i, y'_i are the template points, and a_i is the 6 affine transform coefficients. The solution is then $a = (X^T X)^{-1} X^T x'$. I used inverse warping to ensure that there were no empty pixels in my image once after it had been warped.

The next step was to use inverse compositional image alignment to improve the errors that are inherent to a direct affine warp. This algorithm uses the Jacobian to compute steepest descent images, and then uses the Hessian matrix to decrease the error rates, similar to gradient descent. The algorithm is described below:

Algorithm 1 Inverse Compositional Image Alignment

Initialize $p = p_0$ from input `A`.
Compute the gradient of template image, ∇I_{tpl}
Compute the Jacobian $\frac{\partial W}{\partial p}$ at $(x; 0)$.
Compute the steepest decent images $\nabla I_{\text{tpl}} \frac{\partial W}{\partial p}$
Compute the 6×6 Hessian $H = \sum_x \left[\nabla I_{\text{tpl}} \frac{\partial W}{\partial p}\right]^{\top} \left[\nabla I_{\text{tpl}} \frac{\partial W}{\partial p}\right]$
while $\|p\| > \epsilon$ **do**
    Warp the target to the template domain $I_{\text{tgt}}(W(x; p))$.
    Compute the error image $I_{\text{err}} = I_{\text{tgt}}(W(x; p)) - I_{\text{tpl}}$.
    Compute $F = \sum_x \left[\nabla I_{\text{tpl}} \frac{\partial W}{\partial p}\right]^{\top} I_{\text{err}}$.
    Compute $\Delta p = H^{-1} F$.
    Update $W(x; p) \leftarrow W(x; p) \circ W^{-1}(x; \Delta p) = W(W^{-1}(x; \Delta p); p)$.
end while
Return `A_refined` made of p.

Figure 2: Inverse compositional alignment algorithm

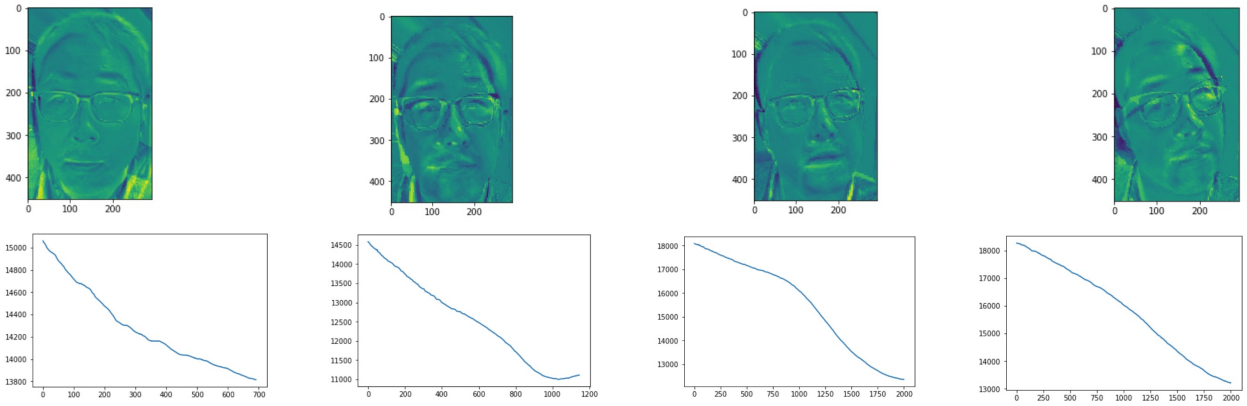

Figure 3: Error images before inverse compositional alignment and the error rate per iteration

Lastly, I just implemented the above procedure for multiple frames of a video, and drew bounding boxes at each step. The resulting images show how this method can be implemented to accurately track objects in video.

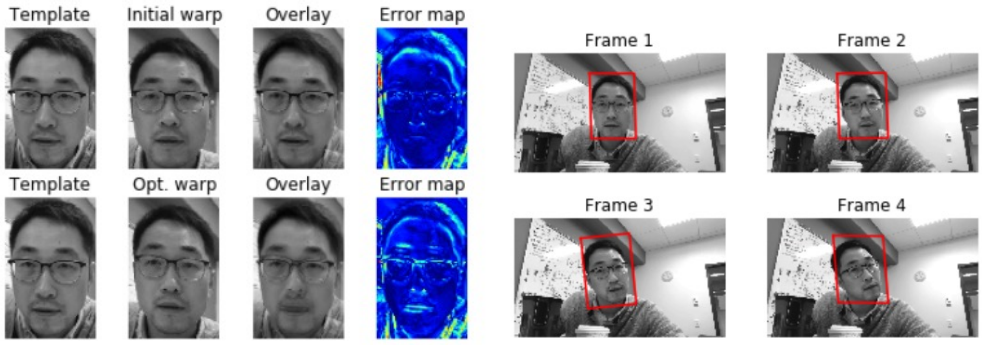

Figure 4: Images the improved error image after inverse compositional alignment and the bounding boxes tracking a face across multiple frames

One downside to this algorithm is the time that it takes for SIFT and inverse compositional alignment to run. In my implementation, it took close to 1000 iterations for the errors to converge. This algorithm would require further optimizations in order to run on real-time video.