16720 Computer Vision Assignment3

Yu Fang Chang, Robotics Institute
yufangc@andrew.cmu.edu
Octobor 20, 2015

1 Lucas-Kanade Tracking

1.1

The goal of the Lucas-Kanade algorithm is to minimize the sum of squared error between two images, the previous frame and the image warped back onto the coordinate frame of the previous image. While minimizing

$$\sum_{(x,y)\in R_t} (I_{t+1}(x+u, y+v) - I_t(x,y))^2$$

is a non-linear optimization, the Lucas-Kanade algorithm assumes that a current estimate of $\mathbf{p}$ is known and then iteratively solves for increments to the parameters $\Delta p = (u,v)^T$. The expression then approximately minimizing

$$\sum_{\mathbf{x}} (I_{t+1}(W(\mathbf{x}; p+\Delta p)) - I_t(\mathbf{x}))^2$$

and then performing a first order Taylor expansion

$$\sum_{\mathbf{x}} (I_{t+1}(W(\mathbf{x}; p) + \nabla I_{t+1}\frac{\partial W}{\partial p}\Delta p - I_t(\mathbf{x}))^2$$

The above expression is a least squares problem and has a closed from solution which can be solved as partial derivative with respect to Δp equals zero:

$$\sum_{\mathbf{x}} [\nabla I_{t+1}\frac{\partial W}{\partial p}]^T [I_{t+1}(W(\mathbf{x}; p)) + \nabla I_{t+1}\frac{\partial W}{\partial p}\Delta p - I_t(\mathbf{x})] = 0$$

$$\Delta p = H^{-1}\sum_{\mathbf{x}} [\nabla I_{t+1}\frac{\partial W}{\partial p}]^T [I_t(\mathbf{x}) - I_{t+1}(W(\mathbf{x}; p))]$$

where $H = \sum_{\mathbf{x}} [\nabla I_{t+1}\frac{\partial W}{\partial p}]^T [\nabla I_{t+1}\frac{\partial W}{\partial p}]$.
To derive $A\Delta p = b$, we get

$$A^T A = H = \sum_{\mathbf{x}} [\nabla I_{t+1}\frac{\partial W}{\partial p}]^T [\nabla I_{t+1}\frac{\partial W}{\partial p}]$$

Since we are computing the Jacobian $\frac{\partial W}{\partial p}$, the warping function $W(\mathbf{x}; \mathbf{p})$ has to be differentiable with respect to the warp parameters $\mathbf{p}$.

1.2

Please refer to LucasKanade.m. I implemented the inverse compositional version of the Lucas-Kanade tracker since it is far more computationally efficient than the Lucas-Kanade algorithm. We only pre-compute the most time-consuming steps once in the inverse version, which are

1. Evaluate the gradient ∇I_t

2. Evaluate the Jacobian $\frac{\partial W}{\partial p}$ at $(\mathbf{x}; 0)$

3. Compute the steepest decent $\nabla I_t \frac{\partial W}{\partial p}$

4. Compute the inverse Hessian matrix

1.3

Please refer to testCarSequence.m for demo and carseqrects.mat for tracking rectangle.

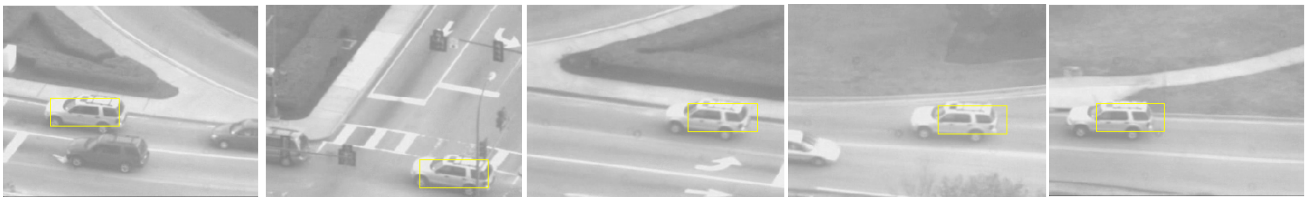

Figure 1: Tracking results at frames 1, 100, 200, 400 and 500.

1.4

2 Lucas-Kanade Tracking with Appearance Basis

2.1

The initial function can be rewrite as following:

$$\sum_{\mathbf{x}} [I_{t+1}(W(\mathbf{x}; p)) - I_t(\mathbf{x}) - \sum_{c=1}^{k} w_c B_c]^2$$

Since the bases are orthonormal, we can seperate the expression to the linear subspace spanned by a collection of vectors by B and its orthogonal complement vectors. Thus, we can derive w_c from minimizing the closed form solution in the bases.

$$w_c = \sum_{\mathbf{x}} B_c(\mathbf{x}) \cdot [I_{t+1}(W(\mathbf{x}; p)) - I_t(\mathbf{x})]$$

2.2

Please refer to LucasKanadeBasis.m.

2.3

Please refer to testSylvSequence.m for demo and sylvseqrects.mat and sylvseqextrects.mat for tracking rectangle. We can found that the initial result is quite good even with some distortion of the tracking object, so there is not much improvement in this approach. Figure 2,3 show the results comparing to question 1.

Figure 2: Tracking results of original sequence at frames 1, 200, 300, 350 and 400. Yellow rectangles are applying the initial approach as question1 while red ones are the improved approach with bases.

Figure 3: Tracking results of extend sequence at frames 1, 400, 800, 1200 and 1300. Yellow rectangles are applying the initial approach as question1 while red ones are the improved approach with bases.

3 Dominant Motion Estimation

3.1

Please refer to LucasKanadeAffine.m. Since I found that the warping function has not converged at the end of the iterations, I add the step size each time in the updated function.

3.2

Please refer to SubtractDominantMotion.m.

3.3

With the mask from the previous question, I applied some binary image function, like im2bw to set a threshold to the difference between warped image and the image, imdilate to dilate the difference and the bwfill to fill the holes and appied bwareaopen twice, setting lower and upper threshold to get a better result.

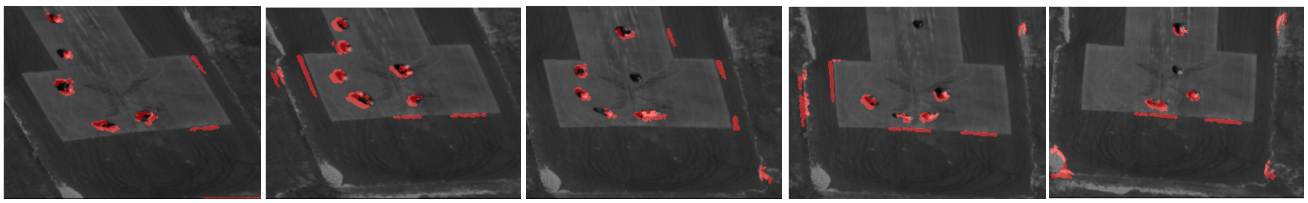

Figure 4: Tracking results at frames 1, 30, 60, 90 and 120. Though the result is not as accurate as the handout, the primary part seems detect correctly.