banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

1. Lucas-Kanade Tracking

Q 1.1.

- It is the Jacobian matrix which varies based on the type of warping (Translation, affine, etc).
  J for translation is $\begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}$. for affine, it is $\begin{bmatrix}x & y & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & x & y & 1\end{bmatrix}$ .p is the warping parameter and x is the pixel in the image.

- b = $T(x_n) - I\big(\omega(x_n; p)\big)$
  where $T(x_n)$ is the template and $I\big(\omega(x_n; p)\big)$ is the image being warped

  A = $\frac{\partial I\big(w(x_n;p)\big)}{\partial P^T}$

- $A^T A$ should be Invertible for a unique solution of Δp can be found.

banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

Q1.2 LK Script

Q 1.3 Execution of the LK Algorithm.

The following images are the outputs for 1, 100, 200,300 and 400th frame respectively.

banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

Q 1.4 Template drift Correction

The following is the Template drifting corrected execution of the LK algorithm

banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

Q 2.1

Express w as a function of the image and the template and bases.

$$I_{t+1} = I_t + \sum_{k=1}^{K} w_k B_k$$

Given:

The bases are orthogonal which makes the dot product with another basis 0 and an inner dot product with itself is a constant (Not 1, since they are not orthonormal)

$\boldsymbol{B_i . B_o}$ **= 0 (1)**

$\boldsymbol{B_i . B_i} = \boldsymbol{constant}$ **(2)**

Bringing I_t to the other side, we get

$$I_{t+1} - I_t = \sum_{k=1}^{K} \omega_k B_k$$

By multiplying Bi on both sides, we get

$$B_i(I_{t+1} - I_{t)} = B_i \sum_{k=1}^{K} \omega_k B_k$$

Taking only the RHS for now, we get

$$= B_i(\omega_1 B_1 + \omega_2 B_2 \ldots \omega_k B_k)$$

By multiplying Bi to the elements inside, we get

$$= (\omega_1 B_1 B_i + \omega_2 B_2 B_i \ldots + \omega_i B_i B_i \ \ldots \omega_k B_k B_i)$$

By given 1 and 2,

$$\omega_i \|B_i\|^2 = B_i(I_{t+1} - I_t)$$

Thus,

$$\boldsymbol{\omega_i = B_i(I_{t+1} - I_t) \,/\, \|B_i\|^2}$$

If the bases are orthonormal,

$$\boldsymbol{\omega_i = B_i(I_{t+1} - I_t)}$$

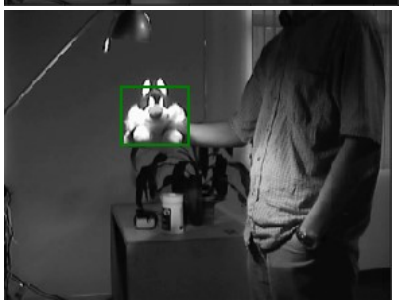
banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

Q 2.2 LucasKanadeBasis - script

Q 2.3 LucasKanadeBasis execution and output

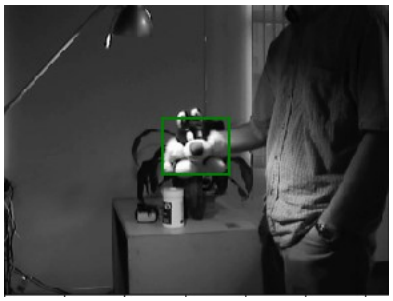
1st Frame

200th Frame



300th Frame

350th Frame



400thFrame

GREEN – LK with Appearance Basis RED – classic LK tracker

The difference in the tracking between the classicLucasKanade Tracker and the one with appearance basis is not so different from one another.

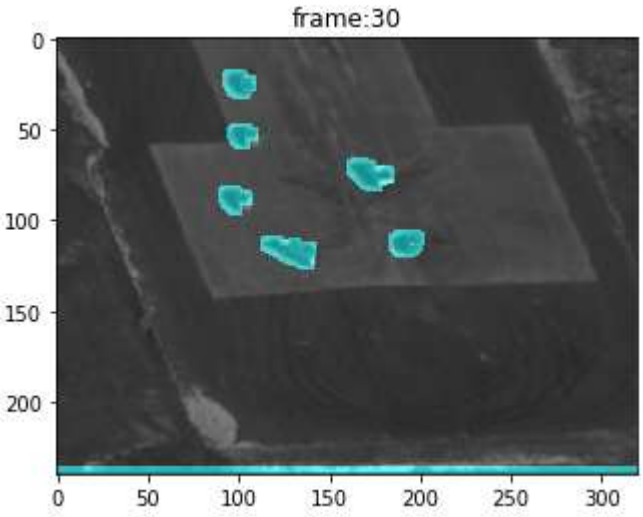
banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

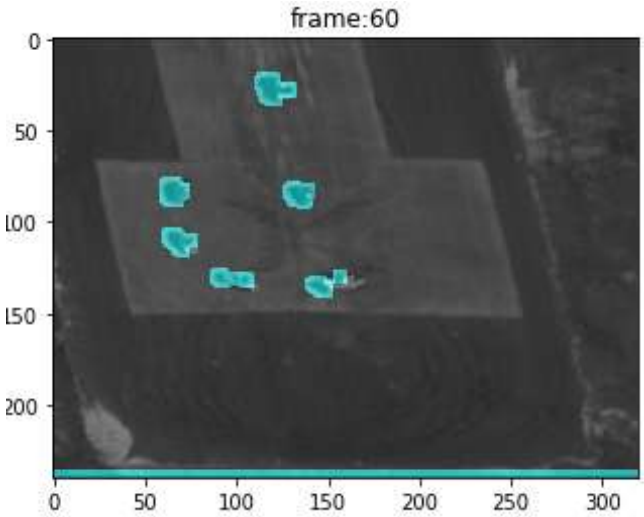
3.1

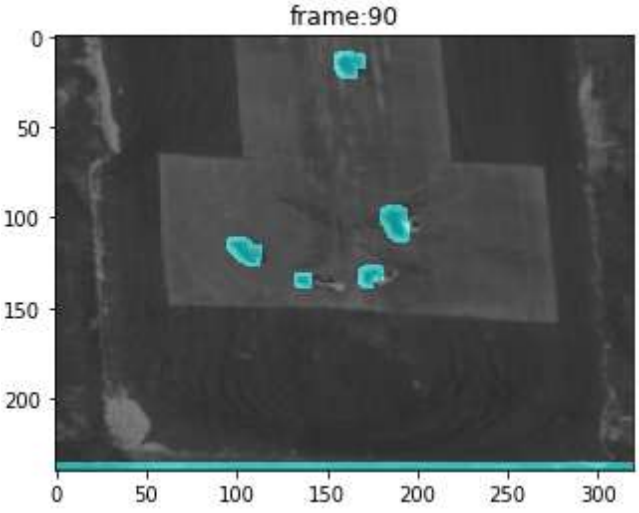
3.2

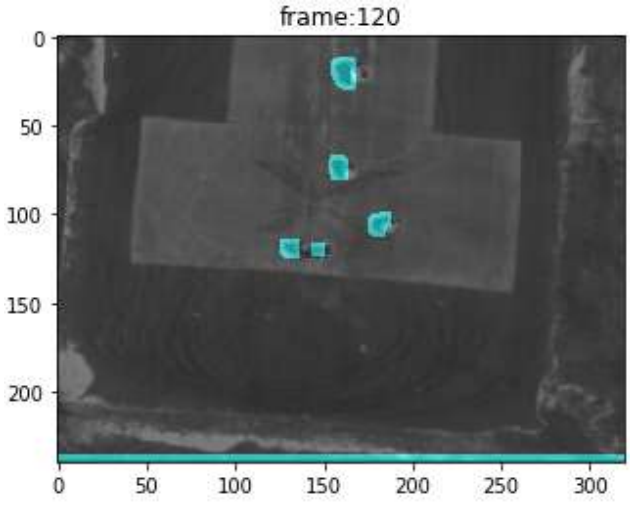
3.3 Moving Object detection:

The following are the 30, 60, 90 and 120th frame of the 'Aerialseq.npy'







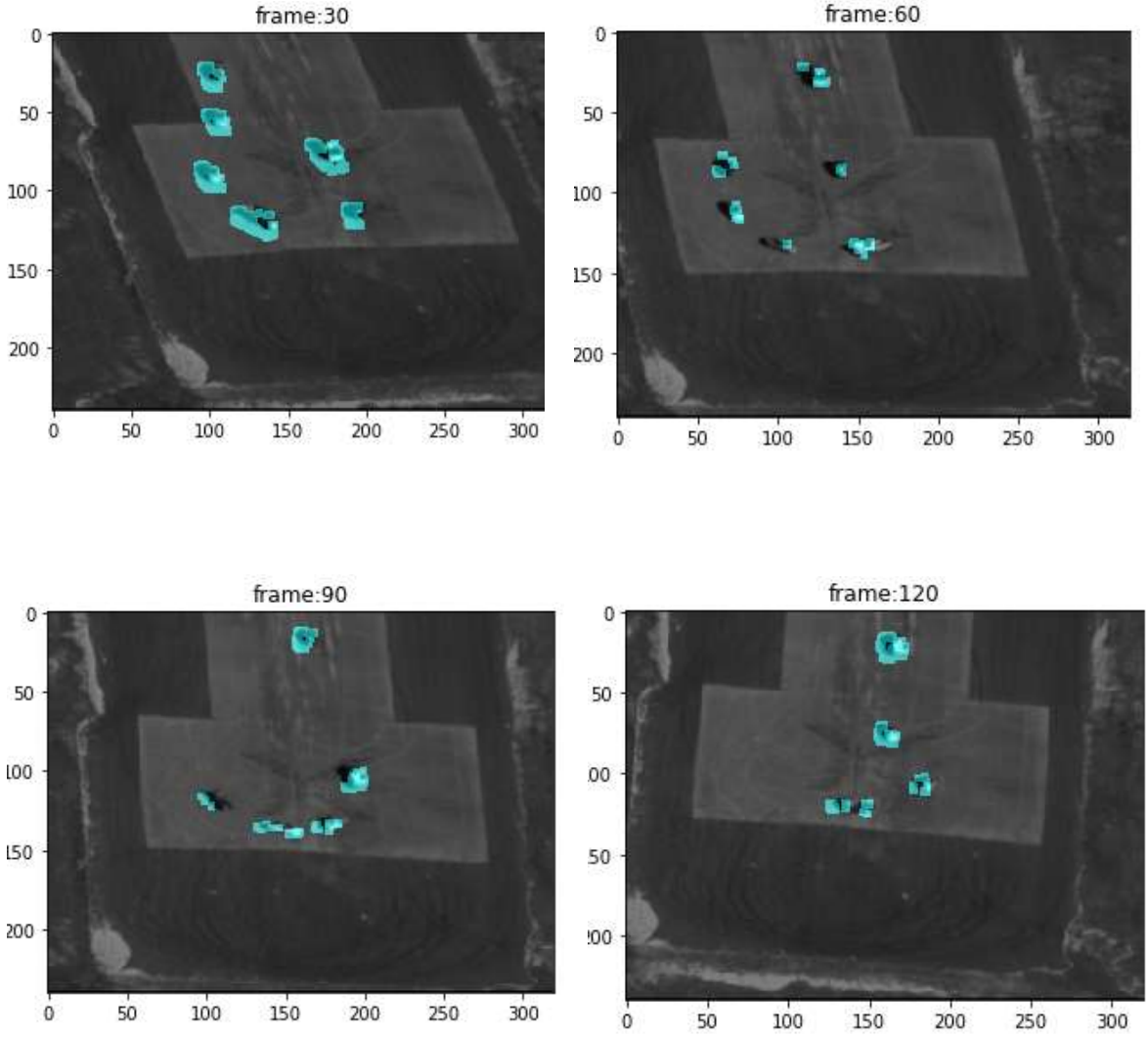


banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

4. Efficient Tracking

4.1 Inverse composition

The difference between the forward and the inverse method is that in the forward, the Hessian is calculated for every iteration (i.e.) The Matrix A is calculated for every iteration whereas in the Inverse composition the A matrix is calculated outside the loop, only once. This is made possible by inverting the roles of the template and the image. By doing that, the Hessian Matrix is calculated only one per frame and the computational efficiency is increased.



banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan

4.2 Correlation Filters.

$$E(g) = arg\ min\ 1/2\|y - X^T g\|_2^2 + \frac{\lambda}{2}\|g\|_2^2$$

In order to minimize 'g', we take the derivative of E with respect to 'g'

$$\frac{\partial E}{\partial g^T} = -X(y - X^T h) + \lambda I g$$

By equating the derivative to zero, we get

$$-X(y - X^T g) + \lambda I g = 0$$

Rearranging,

$$-X(y) + g(\lambda I + X^T X) = 0$$

Taking g to the other side,

$$g = (XX^T + \lambda I)^{-1} Xy$$

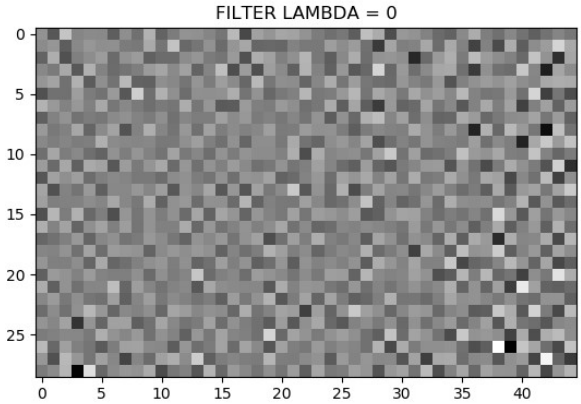
The correlation filter in terms of S, X and y is shown below

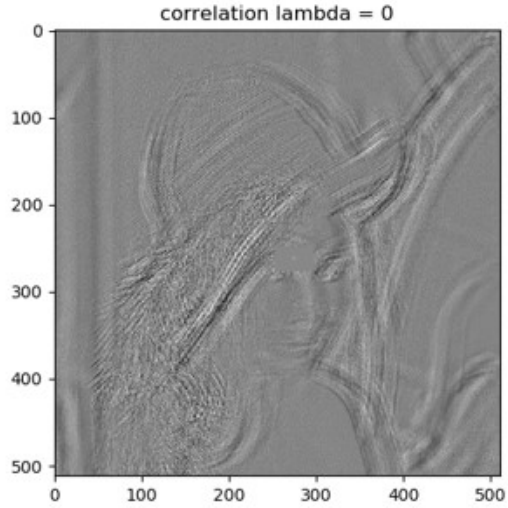
Since, $XX^T = S$

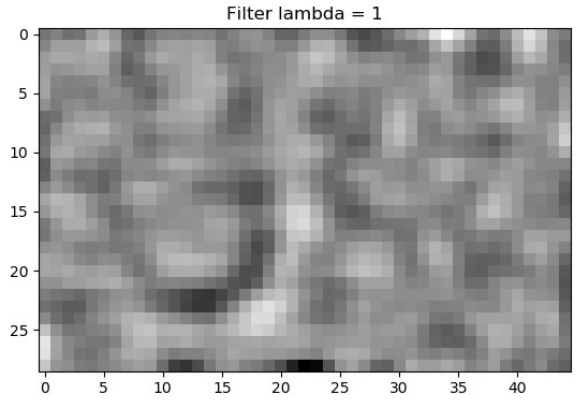
$$g = Xy\ o^{-1}\ (S + \lambda I)$$

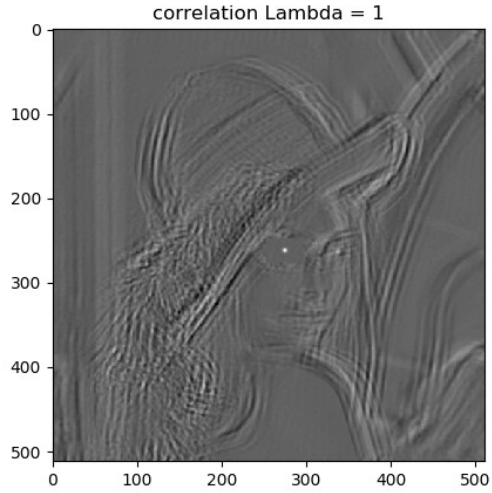
banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan

4.3 Correlation for different lambda values, 1 and 0:









The above figures are the output from the correlation filter for lambda values of 0 and 1 respectively.

The value of lambda when it is 1 performs better. It is due to the fact that it prevents overfitting, whereas the regularising parameter is not in the picture when it is 0 which is obvious and leads to overfitting.

banandan@andrew.cmu.edu
barathwaj Subramaniam Anandan **16720B- Introduction to CV – HW3**

4.4 Convolution for different values of Lambda, 1 and 0

The Kernels are flipped while doing convolution which is the difference between the computation of Correlation and Convolution.

The following are the difference between the output with and without flipping of kernels.

Flipping of kernels is doing by flipping the NumPy array values using 'np.flip'