COMP5421 Computer Vision

Homework Assignment 5
Lucas-Kanade Tracking and Correlation Filters

Hartanto Kwee Jeffrey
jhk@connect.ust.hk — SID: 20851871

1 Lucas-Kanade Tracking

Q1.1

We are given the wrap function

$$\mathcal{W}(x;p) = x + p$$

Hence,

$$\frac{\partial \mathcal{W}(x;p)}{\partial p^T} = \frac{\partial}{\partial p^T}(x+p) = I_{2\times 2}$$

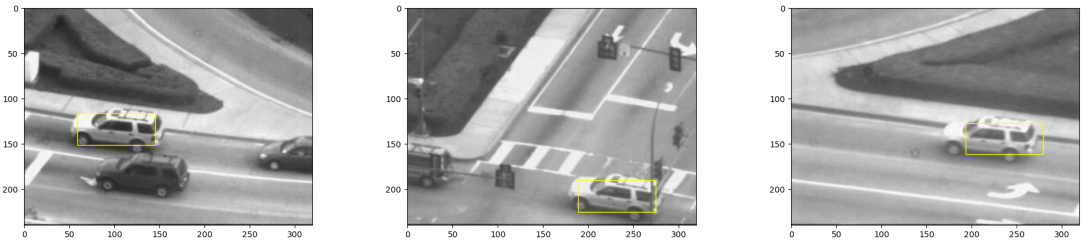
where $I_{2\times 2}$ is the identity matrix.

Our objective function can hence be approximated as

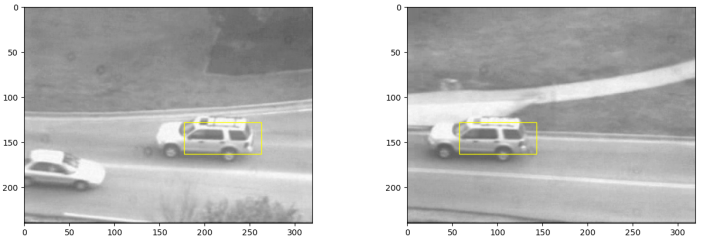
$$\begin{aligned}
&\arg\min_{\Delta p} \sum_x ||\mathcal{I}_{t+1}(x+p+\Delta p) - \mathcal{I}_t(x)||_2^2 \\
\approx &\arg\min_{\Delta p} \sum_x \left|\left|\mathcal{I}_{t+1}(x') + \frac{\partial \mathcal{I}_{t+1}(x')}{\partial x'^T}\Delta p - \mathcal{I}_t(x)\right|\right|_2^2 \\
= &\arg\min_{\Delta p} \sum_x \left|\left|\frac{\partial \mathcal{I}_{t+1}(x')}{\partial x'^T}\Delta p - (\mathcal{I}_t(x) - \mathcal{I}_{t+1}(x'))\right|\right|_2^2 \\
= &\arg\min_{\Delta p} ||A\Delta p - b||_2^2
\end{aligned}$$

where $A = \begin{bmatrix} \frac{\partial \mathcal{I}_{t+1}(x_1')}{\partial x_1'^T} \\ \frac{\partial \mathcal{I}_{t+1}(x_2')}{\partial x_2'^T} \\ \vdots \\ \frac{\partial \mathcal{I}_{t+1}(x_N')}{\partial x_N'^T} \end{bmatrix}$ and $b = \begin{bmatrix} \mathcal{I}_t(x_1) - \mathcal{I}_{t+1}(x_1') \\ \mathcal{I}_t(x_2) - \mathcal{I}_{t+1}(x_2') \\ \vdots \\ \mathcal{I}_t(x_N) - \mathcal{I}_{t+1}(x_N') \end{bmatrix}$. Note that $A \in \mathbb{R}^{N\times 2}$, $\Delta p \in \mathbb{R}^2$ and $b \in \mathbb{R}^N$. For a unique solution of Δp to be found, $A^T A$ should be non-singular.

Q1.3



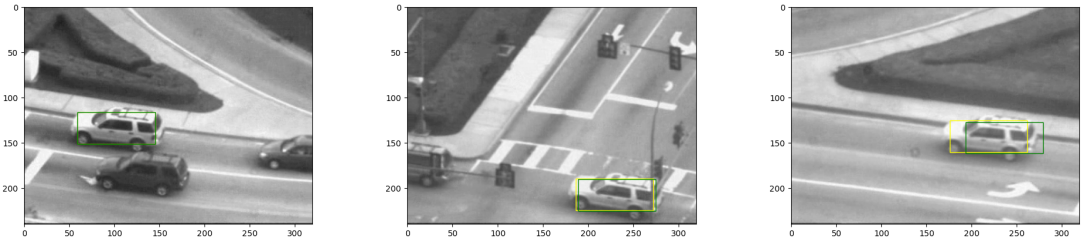
(a) Frame 0 (b) Frame 100 (c) Frame 200



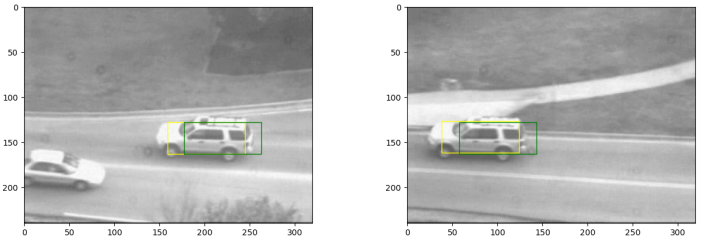
(d) Frame 300 (e) Frame 400

Figure 1: Lucas-Kanade Tracking with One Single Template

Q1.4



(a) Frame 0 (b) Frame 100 (c) Frame 200



(d) Frame 300 (e) Frame 400

Figure 2: Lucas-Kanade Tracking with Template Correction. The yellow bounding box is the result with template correction, and green one is the result without template correction.

2 Appearance Basis

Q2.1

The previous and current frame are related by the following relationship:

$$\mathcal{I}_{t+1}(x) = \mathcal{I}_t(x) + \sum_{k=1}^{K} w_k \mathcal{B}_k(x)$$
$$\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x) = \sum_{k=1}^{K} w_k \mathcal{B}_k(x)$$

Since the basis for B_k are orthogonal to each other, $\mathcal{B}_i(x) \cdot \mathcal{B}_k(x) = 0$ if $i \neq k$. Therefore, to find w_i for some i, we dot both sides with $\mathcal{B}_i(x)$:

$$\begin{aligned}\mathcal{B}_i(x) \cdot (\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x)) &= \mathcal{B}_i(x) \cdot \sum_{k=1}^{K} w_k \mathcal{B}_k(x) \\ &= w_i (\mathcal{B}_i(x) \cdot \mathcal{B}_i(x)) \\ w_i &= \frac{\mathcal{B}_i(x) \cdot (\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x))}{\mathcal{B}_i(x) \cdot \mathcal{B}_i(x)}\end{aligned}$$

Hence, $w = \begin{bmatrix} \frac{\mathcal{B}_1(x) \cdot (\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x))}{\mathcal{B}_1(x) \cdot \mathcal{B}_1(x)} & \frac{\mathcal{B}_2(x) \cdot (\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x))}{\mathcal{B}_2(x) \cdot \mathcal{B}_2(x)} & \dots & \frac{\mathcal{B}_K(x) \cdot (\mathcal{I}_{t+1}(x) - \mathcal{I}_t(x))}{\mathcal{B}_K(x) \cdot \mathcal{B}_K(x)} \end{bmatrix}^T$.

Q2.2

This section provides a derivation of the least squares form of the objective function for the gradient descent.

We consider the new image as $\mathcal{I}_{t+1}(x+p)$. If $\{\mathcal{B}_k\}_{k=1}^K$ is an orthonormal basis, then $\mathcal{B}_k(x) \cdot \mathcal{B}_k(x) = 1$. w reduces to

$$\begin{aligned} w &= \begin{bmatrix} \mathcal{B}_1(x) & \mathcal{B}_2(x) & \dots & \mathcal{B}_K(x) \end{bmatrix}^T (\mathcal{I}_{t+1}(x+p) - \mathcal{I}_t(x)) \\ &= \begin{bmatrix} \mathcal{B}_1(x) & \mathcal{B}_2(x) & \dots & \mathcal{B}_K(x) \end{bmatrix}^T (A\Delta p - b) \end{aligned}$$

We let $B = \begin{bmatrix} \mathcal{B}_1(x) & \mathcal{B}_2(x) & \dots & \mathcal{B}_K(x) \end{bmatrix} \in \mathbb{R}^{N \times K}$. If we let $z = A\Delta p - b$, then it becomes clear that

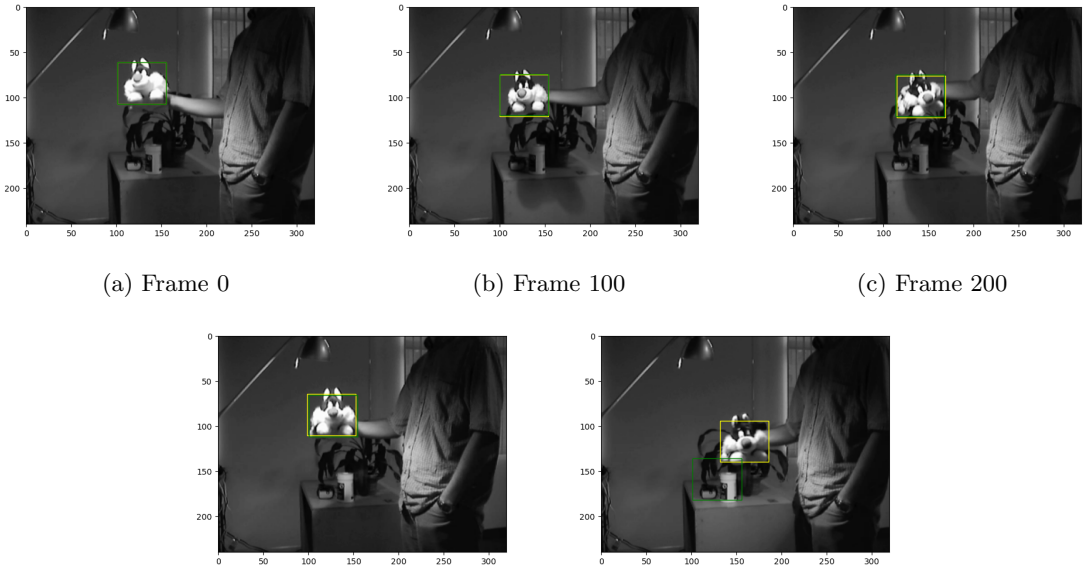
$$\sum_x \left|\left| \mathcal{I}_{t+1}(x+p) - \mathcal{I}_t(x) - \sum_{k=1}^{K} w_k \mathcal{B}_k(x) \right|\right|_2^2 = ||z - Bw||_2^2 = \left|\left|z - BB^T z\right|\right|_2^2$$

Note by orthogonal decomposition, $z = proj_B(z) + proj_{B^\perp}(z) = BB^T z + B^\perp z$. Note that $z \in \mathbb{R}^N$ and $B^\perp \in \mathbb{R}^{N \times N}$. We can rewrite $z - BB^T z$ in terms of Δp:

$$\begin{aligned} z - BB^T z &= A\Delta p - b - BB^T (A\Delta p - b) \\ &= \left(A - BB^T A\right) \Delta p - (b - BB^T b) \\ &= \left(I - BB^T\right) A\Delta p - \left(I - BB^T\right) b \end{aligned}$$

This is the form we will pass into `np.linalg.lstsq`. We also see that $z - BB^T z = \left(I - BB^T\right) z$, so $B^\perp = I - BB^T$.

Q2.3



(a) Frame 0 (b) Frame 100 (c) Frame 200

(d) Frame 300 (e) Frame 400

Figure 3: Lucas-Kanade Tracking with Appearance Basis. The yellow bounding box is the result using appearance basis, and green one is the result with template correction.

3 Affine Motion Subtraction

Q3.1

Here we present a derivation of the update step Δp.

We are given the following wrap function:

$$\mathcal{W}(x;p) = \begin{bmatrix} 1+p_1 & p_2 \\ p_4 & 1+p_5 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} p_3 \\ p_6 \end{bmatrix} = \begin{bmatrix} (1+p_1)\,x + p_2 y + p_3 \\ p_4 x + (1+p_5)\,y + p_6 \end{bmatrix}$$

Hence,

$$\frac{\partial \mathcal{W}(x;p)}{\partial p^T} = \begin{bmatrix} x & y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x & y & 1 \end{bmatrix}$$

The objective function can be approixmated using first-order Taylor expansion:

$$\begin{aligned}
&\arg\min_{\Delta p} \sum_x ||\mathcal{I}_{t+1}(\mathcal{W}(x;p+\Delta p)) - \mathcal{I}_t(x)||_2^2 \\
&= \arg\min_{\Delta p} \sum_x \left|\left| \mathcal{I}_{t+1}(x') + \frac{\partial \mathcal{I}_{t+1}(x')}{\partial x'^T} \frac{\partial \mathcal{W}(x;p)}{\partial p^T} \Delta p - \mathcal{I}_t(x) \right|\right|_2^2 \\
&= \arg\min_{\Delta p} \sum_x \left|\left| \frac{\partial \mathcal{I}_{t+1}(x')}{\partial x'^T} \begin{bmatrix} x_x & x_y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_x & x_y & 1 \end{bmatrix} \Delta p - (\mathcal{I}_t(x) - \mathcal{I}_{t+1}(x')) \right|\right|_2^2 \\
&= \arg\min_{\Delta p} \sum_x \left|\left| \begin{bmatrix} (\mathcal{I}_{t+1})_x(x') & (\mathcal{I}_{t+1})_y(x') \end{bmatrix} \begin{bmatrix} x_x & x_y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_x & x_y & 1 \end{bmatrix} \Delta p - (\mathcal{I}_t(x) - \mathcal{I}_{t+1}(x')) \right|\right|_2^2 \\
&= \arg\min_{\Delta p} \sum_x ||D\Delta p - (\mathcal{I}_t(x) - \mathcal{I}_{t+1}(x'))||_2^2 \\
&= \arg\min_{\Delta p} ||A\Delta p - b||_2^2
\end{aligned}$$

where $D = \begin{bmatrix} (\mathcal{I}_{t+1})_x(x')\,x_x & (\mathcal{I}_{t+1})_x(x')\,x_y & (\mathcal{I}_{t+1})_x(x') & (\mathcal{I}_{t+1})_y(x')\,x_x & (\mathcal{I}_{t+1})_y(x')\,x_y & (\mathcal{I}_{t+1})_y(x') \end{bmatrix}$. Now $A \in \mathbb{R}^{N\times 6}$.

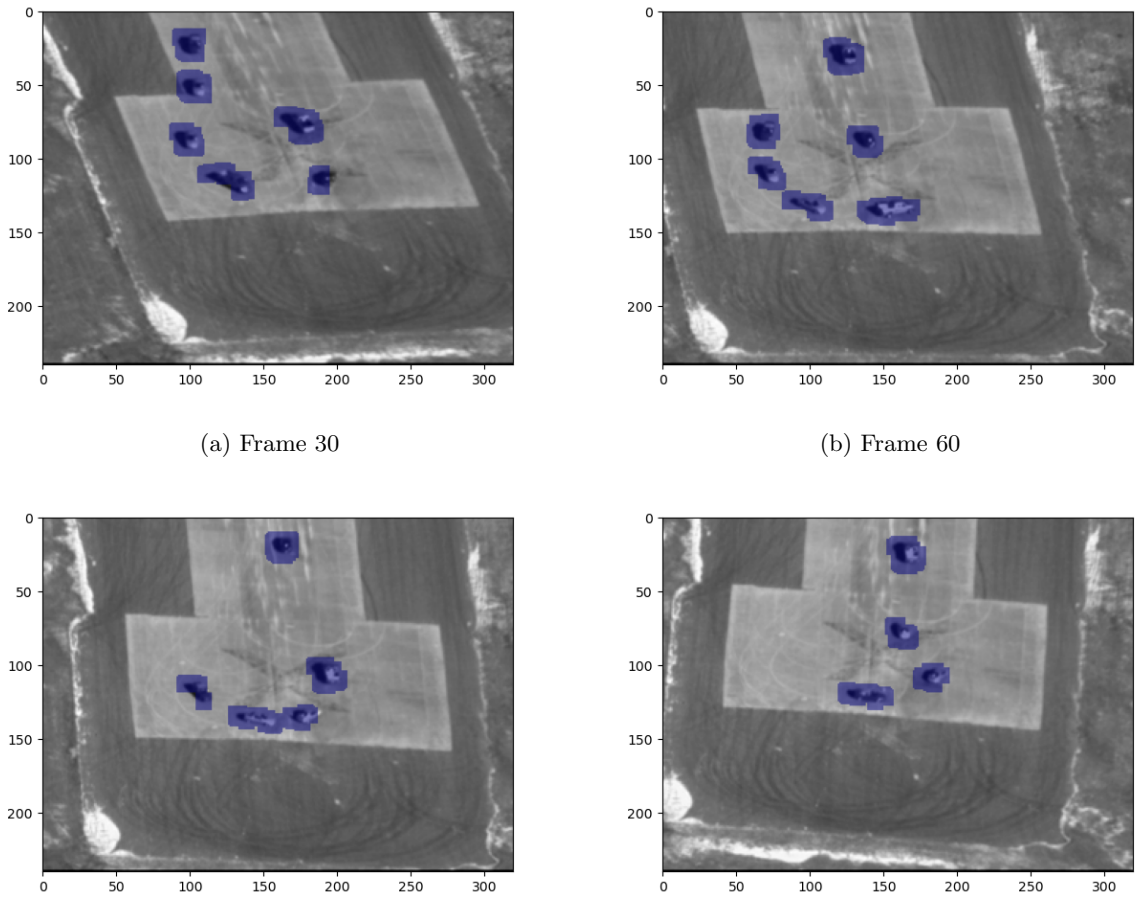
Important Note: The final affine transformation matrix outputted by the function will follow the conventions of x being the horizontal axis and y being the vertical axis, instead of the system used in `scipy` where x and y are the first and second axis of the `numpy` image array.

This is the convention we are adopting: $\begin{bmatrix} 1+p_1 & p_2 & p_3 \\ p_4 & 1+p_5 & p_6 \end{bmatrix}$.

This is the convention for `scipy`: $\begin{bmatrix} 1+p_5 & p_4 & p_6 \\ p_2 & 1+p_1 & p_3 \end{bmatrix}$, which we are **NOT** using.

However, in `SubtractDominantMotion`, we will convert the matrix using our convention to `scipy`'s format. Refer to the code for more information.

Q3.3



(a) Frame 30

(b) Frame 60

(c) Frame 90

(d) Frame 120

Figure 4: Moving Object Detection using Dominant Motion Subtraction.

4 Efficient Tracking

Q4.1

Before discussing its efficiency, we provide a derivation of the updating formula for the inverse compositional gradient descent.
Using first-order Taylor expansion, we can approximate the image under an affine transformation with Δp:

$$\mathcal{I}_t\left(\mathcal{W}\left(x;0+\Delta p\right)\right) \approx \mathcal{I}_t\left(\mathcal{W}\left(x;0\right)\right) + \frac{\partial \mathcal{I}_t\left(x\right)}{\partial x^T}\frac{\partial \mathcal{W}}{\partial p^T}\Delta p$$

Basically, $\mathcal{I}_t\left(\mathcal{W}\left(x;0+\Delta p\right)\right) = \mathcal{I}_t\left(x\right)$. Hence, our objective function can be approximated as

$$
\begin{aligned}
&\sum_x ||\mathcal{I}_t\left(\mathcal{W}\left(x;\Delta p\right)\right) - \mathcal{I}_{t+1}\left(\mathcal{W}\left(x;p\right)\right)||_2^2 \\
\approx &\sum_x \left|\left|\mathcal{I}_t\left(\mathcal{W}\left(x;0\right)\right) + \frac{\partial \mathcal{I}_t\left(x\right)}{\partial x^T}\frac{\partial \mathcal{W}}{\partial p^T}\Delta p - \mathcal{I}_{t+1}\left(\mathcal{W}\left(x;p\right)\right)\right|\right|_2^2 \\
= &\sum_x \left|\left|\frac{\partial \mathcal{I}_t\left(x\right)}{\partial x^T}\begin{bmatrix} x_x & x_y & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & x_x & x_y & 1 \end{bmatrix}\Delta p - \left(\mathcal{I}_{t+1}\left(\mathcal{W}\left(x;p\right)\right) - \mathcal{I}_t\left(\mathcal{W}\left(x;0\right)\right)\right)\right|\right|_2^2 \\
= &||A'\Delta p - b'||_2^2
\end{aligned}
$$

and our update rule is $\mathcal{W}\left(x;p\right) \leftarrow \mathcal{W}\left(x;p\right) \circ \mathcal{W}\left(x;\Delta p\right)^{-1}$.

This approach is computationally more efficient since the pseudoinverse of the $A' = \left(\frac{\partial \mathcal{I}_t\left(x_i\right)}{\partial x_i^T}\frac{\partial \mathcal{W}}{\partial p^T}\right)_i$ term in the inverse compositional method can be precomputed, while the $A = \left(\frac{\partial \mathcal{I}_{t+1}\left(x_i'\right)}{\partial x_i'^T}\frac{\partial \mathcal{W}}{\partial p^T}\right)_i$ in the classical approach has to be recalculated every time the $x' = x + p$ is updated in each iteration. Calculating inverse takes a lot of time, especially when the As contains around 75k rows.

To truly utilize the efficiency of precomputation, instead of selecting all points common to both images' valid area, we assume that the motion between two images are minimal and we only select points from a rectangle centered on the image.

Both versions of the affine matrix alignment are written in `SubtractDominantMotion.py`. Uncomment the code to run each version.

Q4.2

The filter we want to obtain is given by $\arg\min_g \sum_{n=1}^N \frac{1}{2}||y_n - x_n^T g||_2^2$, or $\arg\min_g \frac{1}{2}||y - X^T g||_2^2 + \frac{\lambda}{2}||g||_2^2$. Differentiate the objective function with respective to g to find the minimum:

$$
\begin{aligned}
&\frac{\partial}{\partial g}\left(\frac{1}{2}||y - X^T g||_2^2 + \frac{\lambda}{2}||g||_2^2\right) \\
= &-X\left(y - X^T g\right) + \lambda g \\
= &\left(XX^T + \lambda I\right)g - Xy \\
= &\left(S + \lambda I\right)g - Xy
\end{aligned}
$$

The minimum is attained when the gradient is zero:

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

Q4.3

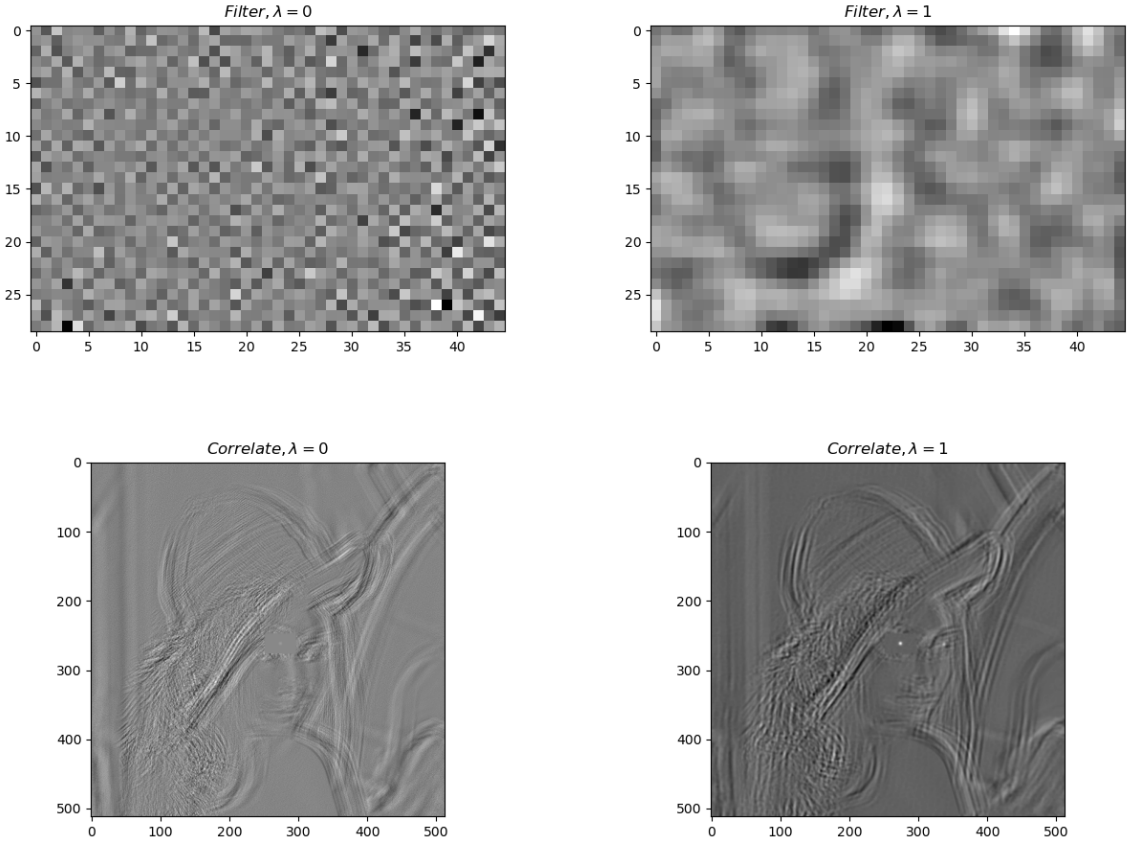


Figure 5: Linear discriminant weight vector and filter response under different regularization factors: $\lambda = 0$ on the left and $\lambda = 1$ on the right.

It should be emphasized that the motivation for correlation filters is to find a filter g, which will provides us with a desired response y from a series of discriminators x extracted from some part of the image. The performance we're trying to asses is how well the correlation filter gives us the desired response.

Viewing the code, the desired response is $\exp(-||dp||_2^2/5)$, where $||dp||_2^2 = dx^2 + dy^2$ is the distance away from the center of Lena's eye; the less the distance, the stronger the response. This means that we want to fit a filter which gives a clear, strong response for an eye. Comparing the responses generated by the two correlation filters with different regularization, we find that $\lambda = 1$ gives a strong response for the eye that we didn't use for fitting g (i.e. the eye on the right), while we only see random noise for $\lambda = 0$.

The reason for this is $\lambda = 1$ gives more regularization than $\lambda = 0$, preventing g from straying away from the origin (the origin gives the default response, since the response to any image patch would evaluate to 0) and effectively preventing g from overfitting to specifically the eye on the left. In other terms, $\lambda = 1$ provides a trust region while $\lambda = 0$ doesn't.

Q4.4

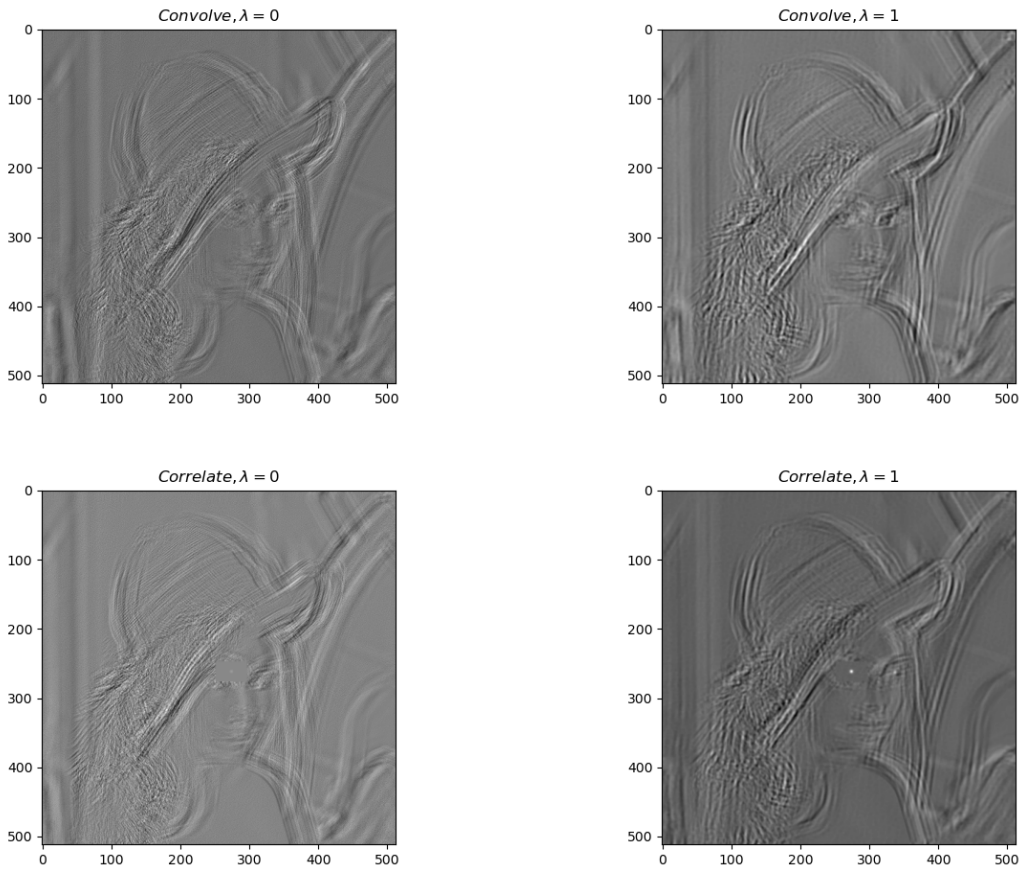


Figure 6: Image response under convolution (on the top) and correlation (on the bottom) using the same filter.

The response is different because convolution flips the filter horizontally and vertically before element-wise multiplication with the cropped image window, but correlation applies the given filter as it is. To get the same response as correlation using convolution, flip the filter before passing it into `scipy.ndimage.convolve`, like `filter[::-1,::-1]`. This means that the following two statements should give the same results:

```
scipy.ndimage.correlate(img, filter)
scipy.ndimage.convolve(img, filter[::-1,::-1])
```