Quiz 2 Solutions

Mithilaesh Jayakumar (G01206238)

1. A Hessian Matrix is a square matrix of second ordered partial derivatives for determining points of local maxima or minima. It gives the second ordered partial derivatives of image intensity and its variations around the selected window of pixels. The eigenvector decomposition from the resulting Hessian matrix extracts an orthonormal coordinate system that is aligned with the second order structure of the image. The eigenvector e1 points in the direction of the highest curvature with the magnitude $\lambda 1$. Similarly, e2 corresponds to the direction of lowest curvature with the strength of $\lambda 2$. From these eigenvalues and their resulting theoretical behavior, a decision can be made if the analyzed window of pixel belongs to the structure being searched.
   - If $\lambda 1 = 0$ and $\lambda 2 = 0$ then that pixel point has no features of interest which means it is a flat surface.
   - If $\lambda 1 = 0$ and $\lambda 2 > 0$ then that pixel point is an edge.
   - If $\lambda 1 > 0$ and $\lambda 2 > 0$ then that pixel point is a corner.

2. We can use the Hessian matrix to determine the concavity and the curvature of a surface in the image. This can be done by using the eigenvalues of the Hessian matrix and the Hessian determinant. Let $\lambda 1$ and $\lambda 2$ be the eigen values and D be the determinant of the Hessian matrix. Then the properties of concavity and the curvature based on the eigen value and determinant of Hessian matrix can be defined as follows.
   - If $\lambda 1 > 0$ and $\lambda 2 > 0$ then $D > 0$ so the concavity is consistent, and curvature becomes positive resulting the surface to concave up.
   - If $\lambda 1 < 0$ and $\lambda 2 > 0$ then $D < 0$ so the concavity is inconsistent, and curvature becomes negative resulting in saddle point.
   - If $\lambda 1 > 0$ and $\lambda 2 < 0$ then $D < 0$ so the concavity is inconsistent, and curvature becomes negative resulting in saddle point.
   - If $\lambda 1 < 0$ and $\lambda 2 < 0$ then $D > 0$ so the concavity is consistent, and curvature becomes positive resulting the surface to concave down.

3. In order to get the curvature using properties mentioned in 2 we need to provide three parameters namely the Hessian matrix H which acts as the curvature tensor containing the information about every direction, a specific direction P in which we are interested and the location $(x_0,y_0)^T$ where we want to find the curvature. The direction is usually provided as uniform column vector. We can compute this using the formula $P^T * H * P$. Suppose if the point (x_0,y_0) is (0,0) and the direction is 45^0 $[\cos(\pi/4), \sin(\pi/4)]$ from that point then using this formula would provide a curvature of -0.5.

4. We know that the Hessian matrix is used to identify the cornerness but calculating eigen values of Hessian matrix for all image pixels is computationally complicated. Harris operator or Harris corner detector identifies corner from hessian matrix in a much simpler way.

   $$\text{Harris} = \det(H) - a * \text{trace}(H)$$

   where a is a constant and trace(H) is the sum of diagonal elements of Hessian matrix and det(H) is the determinant of Hessian matrix. We already know the algorithm for Harris detector.

   Harris Detector Algorithm:
   - Consider the grayscale of the original image
   - Apply a Gaussian filter to smooth out any noise
   - Apply Sobel operator to find the x and y gradient values for every pixel in the grayscale image

- For each pixel p in the grayscale image, consider a 3×3 window around it and compute the corner strength function and call it as Harris value.
- Find all pixels that exceed a certain threshold and are the local maxima within a certain window.
- For each pixel that meets this threshold criteria, compute a feature descriptor.

Now, in order to implement a scale invariant Harris detector, we measure the stability by checking whether the corner is still present after a zoom out. At each scale, we first zoom out the image by a factor of two and compute the Harris corners. This is done in a recursive way, as many times as the number of scales chosen. Then, we compute the corners at the current scale and check that they are present in both scales. The implementation is as follows.

- Compute Harris corners at coarsest scale.
- Zoom out the image by a factor of two.
- Compute Harris corners at the coarse scale (recursive).
- Compute Harris corners at the current scale.
- Select stable corners

The algorithm stops at the coarsest scale. The value of sigma also gets reduced by a factor of two during this process at the coarse scales. This allows to preserve the same area of integration. While selecting stable corners we select the points at the finer scale for which there exists a corner at a distance less than sigma.