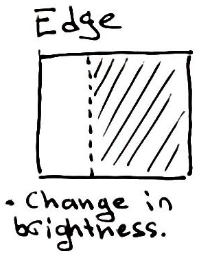
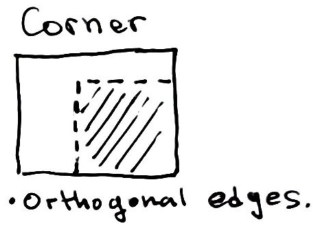
Feature Points (Keypoints + Descriptors)

- A feature is composed of a keypoint and a descriptor.
- A keypoint will contain the 2D point position, and sometimes scale, and orientation.
- A descriptor will contain a visual description of the keypoint, it can be used to compare similarities between feature points.

- Keypoint include corners, edges
  - Keypoint detection methods:
    1. Harris
    2. Shi-Tomasi
    3. Forstner
    4. Difference of Gaussian (DoG)





- Structure matrix (tensor).

$$M = \begin{vmatrix} \sum_w J_x J_x & \sum_w J_x J_y \\ \sum_w J_x J_y & \sum_w J_y J_y \end{vmatrix}$$

- Also called second moment matrix.
- Can be derived by taking a gradient, of a sum of squared differences (SSD) denoted by $f(x,y)$.

$$f(x,y) = \sum_{(x',y') \in W_{xy}} \left[I(x',y') - I(x'+dx', y'+dy') \right]^2$$

- M describes the distribution of a gradient in the window W around point x, y.
- I is an image.
- W is a window around point x, y, that we are shifting by dx, dy.
- J_x is a derivative of I with respect to x. J_y with y.

Actually, let me derive it quick...
Taylor of $I(x'+dx, y'+dx)$ is...

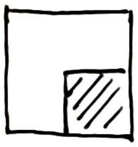
$\sim I(x',y') + I_x(x',y') + I_y(x',y')\ldots$

Cool this is similar to Jacobian $\left[dI/dx' \; dI/dy' \right]$ so, $\underbrace{I(x',y')}_{\text{will cancel out.}} + [J_{x'} \, J_{y'}] \begin{bmatrix} dx' \\ dy' \end{bmatrix}$

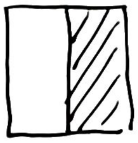
$$f(x,y) \approx \underbrace{\sum_w \left[[J_x \, J_y] \begin{bmatrix} dx' \\ dy' \end{bmatrix} \right]^2}_{\text{Will result in } M\text{, change in } I \text{ in local area.}}$$

- Scharr operator $D_x^H = \frac{1}{32}\begin{vmatrix} 3 & 10 & 3 \\ 0 & 0 & 0 \\ -3 & -10 & -3 \end{vmatrix}$ $D_y^H = \frac{1}{32}\begin{vmatrix} 3 & 0 & -3 \\ 10 & 0 & -10 \\ 3 & 0 & -3 \end{vmatrix}$
- Sobel operator $D_x^S = \frac{1}{8}\begin{vmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{vmatrix}$ $D_y^S = \frac{1}{8}\begin{vmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{vmatrix}$

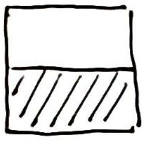
We use convolution to apply scharc and Sobel operators.


$M = \begin{vmatrix} C_x & 0 \\ 0 & C_y \end{vmatrix}$ $C_x > 1$, $O \sim 0$. $C_y > 1$.

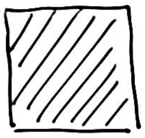
Corner.


$M = \begin{vmatrix} 0 & 0 \\ 0 & C_y \end{vmatrix}$

Edge.


$M = \begin{vmatrix} C_x & 0 \\ 0 & 0 \end{vmatrix}$

Edge.


$M = \begin{vmatrix} 0 & 0 \\ 0 & 0 \end{vmatrix}$

Blank or filled. (Flat).

- Harris corner.

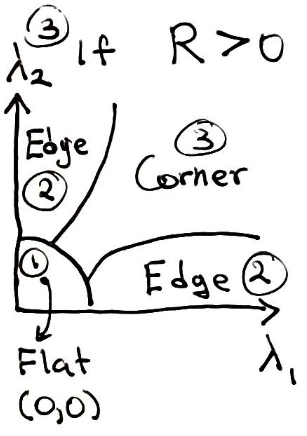
$$R = \det(M) - K(\mathrm{trace}(M))^2 = \lambda_1\lambda_2 - K(\lambda_1 + \lambda_2)^2$$

K is a constant between $K \in [0.04$ and $0.06]$.

① If $|R| \sim 0 \rightarrow \lambda_1 \sim 0$ and $\lambda_2 \sim 0$: Flat.

② If $R < 0 \rightarrow \lambda_1 > \lambda_2$ or $\lambda_2 > \lambda_1$: Edge.

③ If $R > 0 \rightarrow (\lambda_1 \sim \lambda_2) > 0$: Corner.

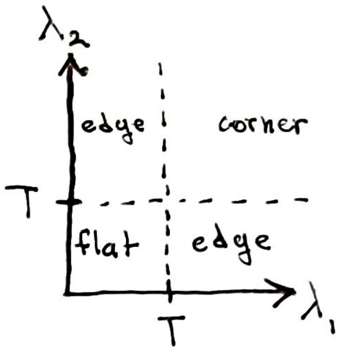


- Shi-Tomasi Corner.

$$\lambda_{min}(M) = \frac{trace(M)}{2} - \frac{1}{2}\sqrt{(trace(M))^2 - 4\det(M)}$$

* Smallest eigenvalue

if $\lambda_{min}(M) \geq T$: corner



- Difference of Gaussian (DoG)

  - The idea is very simple. Blur the same image using a gaussian blur, using 2 different variances σ_1 and σ_2, with $\sigma_2 > \sigma_1$. This way we get blurred image 1 with variance 1 and image 2 with variance 2. Then we subtract the image 1 and 2, from each other. This will increase the visibility of corners, edges and other details. Additionally, this can be extended to scale, and used in similar fashion as pyramids.

* My concern with gaussian is that due to blurring the sharp edges are not preserved. It might be interesting to try a median filter, as its edge preserving.

On the second note, we can use sigma of small values, to reduce the blurring effect. Small sigma is less blurring, and more detail. Larger sigma more blurring, less detail.

Lets say image is $I(x,y)$.
We have gaussian $g_1(x,y) = \frac{1}{\sqrt{2\pi\sigma_1^2}} e^{\left(-\frac{x^2+y^2}{2\sigma_1^2}\right)}$ similar one for g_2.
Taking a gaussian of an image will be $G_1(x,y) = g_1(x,y) * I(x,y)$.
$\uparrow$ Convolution

$$DoG = G_1(x,y) - G_2(x,y) = I * (g_1 - g_2)$$

DoG acts like a band pass filter. Gaussia is a low pass filter, but we are restricting our signal to be between two frequencies, images.

*I would recommend to look into Laplacian of Gaussian (LoG). We can look at the divergence of the gradient of the image. It can be used to find edges but also blobs.