Evaluation Metrics

- **For Regression**: MSE, MAE, RMSE; **For Classification**: Accuracy, Precision, Recall, F1 Score
- Accuracy = $(TP + TN)/(TP + TN + FP + FN)$, Precision = $(TP)/(TP + FP)$, Recall = $(TP)/(TP + FN)$, F1 Score = $(2 \times \text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall})$
- For Semantic Segmentation: IoU, Dice Coefficient, Pixelwise Precision; For Object Detection: mAP
- IoU = Area of Overlap/Area of Union. It is typically used to assess quality of locality. **Distance IoU** $= 1 - IoU + d^2/c^2$ where d is the distance between the centers of the two bounding boxes, and c is the diagonal of the smallest bounding box that encloses both boxes. **mIoU** is average IoU over all classes.
- **Average Precision**: area under the precision-recall curve, $mAP = \frac{1}{n}\sum_{i=1}^{n} AP_i$, where AP_i is the average precision for class i. Precision and recall are calculated at different thresholds of IoU. Calculating mAP over a range of IoU thresholds is the norm.
- **Dice Coefficient**: $(2 \times |X \cap Y|)/(|X| + |Y|)$, where X is the predicted set of pixels, and Y is the ground truth set of pixels.
- IoU loss saturates once the objects have no overlap and, therefore may not allow to learn better features for instances where the predicted and ground-truth boxes do not overlap
- mAP deteriorates more severely for smaller objects than larger objects due to its reliance on IoU.

Image Processing

- **Low Pass Filters**: Gaussian, Median, Averaging; **High Pass Filters**: Original - Low Pass [Detail]
- **Gaussian Kernel**: $G_\sigma = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$. Convolution with self is another gaussian kernel with $\sigma = \sqrt{2} \times$ original
- **Edge Detection**: - Sudden change in intensity. Take discrete derivative $\frac{\partial f}{\partial x}[x, y] \approx F[x + 1, y] - F[x, y]$. Gradient of an image is ∇f . Edge strength is $||\nabla f||$ and gradient direction is θ

$$\nabla f = [\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}] \text{ , } ||\nabla f|| = \sqrt{(\frac{\partial f}{\partial x})^2 + (\frac{\partial f}{\partial y})^2} \text{ , } \theta = \arctan(\frac{\partial f}{\partial y}/\frac{\partial f}{\partial x})$$

Noise in the image might cause too many rapid changes, so we smooth the image before taking the derivative. $\frac{d}{dx}(f * h) = f * \frac{d}{dx}h$. For 1D Gaussian, $G_\sigma(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{x^2}{2\sigma}}, G'_\sigma(x) = -\frac{1}{\sigma}(\frac{x}{\sigma})G_\sigma(x)$

Harris corner detection for feature detection

- Key Idea: Small shifts in the window should cause large changes in the intensity
- Compute image gradients for each pixel over a small region W. Let the image gradients be I_x, I_y.
- Subtract mean from each image gradient to remove the constant intensity and compute the covariance $H = \begin{bmatrix} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{bmatrix}$

This comes from second order taylor expansion of $E(u, v) = \sum_{x,y} w(x, y)[I(x + u, y + v) - I(x, y)]^2$, where x, y are the points in the window, and $w(x, y)$ is the window function.

- Compute eigenvectors and eigenvalues $Hx = \lambda x$
  - $x_{\max}, x_{\min}$ are direction of the (largest,smallest) change in intensity
  - $\lambda_{\max}$, $\lambda_{\min}$ are the amount of increase in direction of $x_{\max}$ and $x_{\min}$ respectively
- Depending on the value of $\lambda_{\max}$ and $\lambda_{\min}$, we can classify the point as a corner, edge, or flat region $\lambda_{\max}$ is large, $\lambda_{\min}$ is small $\rightarrow$ Edge. $\lambda_{\max}$ and $\lambda_{\min}$ are large $\rightarrow$ Corner. $\lambda_{\max}$ and $\lambda_{\min}$ are small $\rightarrow$ Flat region
- Choose points where $\lambda_{\min}$ is local maxima. Normally, we just approximate $\lambda_{\min}$ with Harris operator, which is usually much easier to compute. one common choice is $\frac{\det(H)}{\text{trace}(H)} >$ threshold
- Not invariant to rotation, scale, and illumination changes

SIFT for feature description

- **Scale Invariant Feature Transform**. Key Idea: Find keypoints that are invariant to scale, rotation, and illumination changes
- Take 16x16 square window around detected feature and compute edge orientation (angle of the gradient - 90°) for each pixel
- Throw out weak edges (threshold gradient magnitude) and reate histogram of surviving edge orientations
- Can handle changes in viewpoint (up to about 60 degree out of plane rotation)
- Can handle significant changes in illumination (sometimes even day vs. night)
- Pretty fast — hard to make real-time, but can run in <1s for moderate image sizes

Geometry

$$R_z(\theta) = \begin{bmatrix} \cos(\theta) & -sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, R_y(\theta) = \begin{bmatrix} \cos(\theta) & 0 & \sin(\theta) \\ 0 & 1 & 0 \\ -\sin(\theta) & 0 & \cos(\theta) \end{bmatrix}, R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}, [\mathbf{t}]_\times = \begin{bmatrix} 0 & -t_3 & t_2 \\ t_3 & 0 & -t_1 \\ -t_2 & t_1 & 0 \end{bmatrix}$$

- Euclidian Transformations (Translation, Rotation): $x' = Rx + t$. Preserves distances (areas). Degrees of freedom: 3
- Similarity Transformations (Translation, Rotation, Scaling): $x' = sRx + t$. Preserves angles and ratios of distances. Degrees of freedom: 4
- Affine Transformations (Translation, Rotation, Scaling, Shear): $x' = Ax + t$. Preserves parallel lines, ratio of areas and lengths. Degrees of freedom: 6
- Projective Transformations (Homography): $x' = Hx$. Preserves straight lines. Degrees of freedom: 8

Projective Geometry

- Planes passing through origin and $\perp$ to vector $\mathbf{n}$: $\mathbf{n} \cdot \mathbf{x} = 0$ i.e. $ax_1 + bx_2 + cx_3 = 0$
- Vector $\parallel$ to intersection of 2 planes (a, b, c) and (a', b', c'): $\mathbf{n}'' = \mathbf{n} \times \mathbf{n}'$
- Planes passing through two points $\mathbf{x}$ and $\mathbf{x}'$: $\mathbf{n} = \mathbf{x} \times \mathbf{x}'$
- To each point m of the plane P we can associate a single ray $\mathbf{x} = (x_1, x_2, x_3)$ passing through the origin
- To each line l of the plane P we can associate a single point $\mathbf{l} = (l_1, l_2, l_3)$

We can go in reverse as well

- If we have a line $ax + by + c = 0$, then the point $\mathbf{l} = (a, b, c)$ is the corresponding plane in 3D, and if $x = \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T \in l$, then $\begin{bmatrix} x_1 & x_2 & 1 \end{bmatrix} \begin{bmatrix} a & b & c \end{bmatrix}^\top = 0$
- Point of intersection of 2 lines $x = l_1 \times l_2$
- Line at infinity: $l_\infty = (0, 0, 1)^T$, Point at infinity is always of the form: $\mathbf{x}_\infty = (a, b, 0)^T$
- For a line $l = \begin{bmatrix} a & b & c \end{bmatrix}^T$, the point at infinity is $\mathbf{x}_\infty = (b, -a, 0)^T$

Normalized 8-point algorithm for estimating the fundamental matrix + DLT

$$\begin{bmatrix} x_1x_1' & x_1y_1' & x_1 & y_1x_1' & y_1y_1' & y_1 & x_1' & y_1' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_mx_m' & x_my_m' & x_m & y_mx_m' & y_my_m' & y_m & x_m' & y_m' & 1 \end{bmatrix} \begin{bmatrix} f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 & f_8 & f_9 \end{bmatrix}^\top = 0 \rightarrow \mathbf{AX} = \mathbf{0}$$

- Normalize the points (since we are working in image coordinates) [Let normalizing matrix in both cameras be $\mathbf{T}$ and $\mathbf{T}'$]
- Construct the matrix $\mathbf{A}$. Find SVD of $\mathbf{A}$, and using that, find approximate value $\hat{\mathbf{F}}$, and enforce rank-2 constraint to get $\mathbf{F}$

$$\min_{\mathbf{F}} ||\mathbf{F} - \hat{\mathbf{F}}||_F, \text{subject to } \det \mathbf{F} = 0 \qquad \rightarrow \qquad \mathbf{F} = U \begin{bmatrix} \Sigma_1 & 0 & 0 \\ 0 & \Sigma_2 & 0 \\ 0 & 0 & 0 \end{bmatrix} V^\top \text{ where } \hat{\mathbf{F}} = U\Sigma V^\top$$

- Denormalize the fundamental matrix to get the final fundamental matrix $\mathbf{F}_{\text{final}} = (\mathbf{T}')^\top \mathbf{F} \mathbf{T}$
- Fundamental Matrix Equation: $\mathbf{x}'^\top \mathbf{F} \mathbf{x} = 0$, Homography Equation: $\mathbf{x}' \times \mathbf{H}\mathbf{x} = \mathbf{0}$
- For DLT, every point generates 2 pairs $\begin{bmatrix} x_1 & y_1 & 1 & 0 & 0 & 0 & -x_1x_1' & -x_1'y_1' & -x_1' \end{bmatrix}, \begin{bmatrix} 0 & 0 & 0 & x_1 & y_1 & 1 & -y_1x_1' & -y_1'y_1' & -y_1' \end{bmatrix}$ in $\mathbf{A}$, so you only need 4 corresp. points. Baki same algo only for homography estimation (except for rank 2 constraint and normalization)

RANSAC

- Randomly sample a subset of points, fit a model to that subset (Take 4 points, compute homography using DLT)
- Find the number of inliers for that model [$d(p'i, Hpi) < \epsilon$]
- Repeat for a large number of iterations, and choose the model with the most inliers (Re-compute least-squares H estimate using all of the inliers)
- Robustness : If there are G proportion of inliers, and our model need P pairs, Probability of picking P inliners = G^P, Probability of not picking set of inliners after N iterations = $(1 - G^P)^N$
- If there are multiple structures, RANSAC won't out of the box due to unknown number of structures, inlier set for each structure, and noise affecting the structures.

Essential Matrix and Fundamental Matrix Properties

- Essential matrix for calibrated cameras, fundamental matrix for uncalibrated cameras. Replace $\mathbf{E}$ with $\mathbf{F}$ and all conditions still hold unless stated differently. $\mathbf{F} = (\mathbf{K}')^{-\top} \mathbf{E} \mathbf{K}^{-\mathbf{1}}$
- $\mathbf{E}$ is singular, i.e., $\det(\mathbf{E}) = 0$, and has 5 degrees of freedom, and rank 2. $\mathbf{F}$ has 7 degrees of freedom.
- $(x')^\top l' = 0$, $x^\top l = 0$; $\quad l' = \mathbf{E}x, l = \mathbf{E}x'$; $\quad (e')^\top \mathbf{E} = 0$, $\mathbf{E}e = 0$
- $(x')^\top \mathbf{E} x = 0$ for corresponding points x and x'. (the epipolar constraint)
- $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$, where $\mathbf{R}$ is the rotation matrix and $[\mathbf{t}]_\times$ is the skew-symmetric matrix of the translation vector $\mathbf{t}$.