Model fitting:

$y = ax + b \rightarrow d = x\cos\theta + y\sin\theta$

$d(P, \mathcal{g}) = \frac{|\mathcal{g}(P)|}{|\nabla \mathcal{g}(P)|}$ — $|\mathcal{g}(P)|$ → Algebraic distance; $|\nabla \mathcal{g}(P)|$ → Gradient

Hough transform: Detect lines by mapping points in images space to parameter space. A point, a line

Bin sizes → Small bins: ↑ Resolution for detecting, ↑ Computational costs
→ Large bins: ↓ Memory and computational cost, ↓ Precision

Geometric distance: Shortest distance between a point an a curve

Contour: Curve that represents a boundary or edge of an object $\phi(s) = \begin{bmatrix} x(s) \\ y(s) \end{bmatrix}$

- We want to minimize the error functional $E[\phi(s)] = \int [\underbrace{d(s)E_{cont} + B(s)E_{curv}}_{\text{Internal energy}} + \underbrace{\gamma(s)E_{img}}_{\text{External energy}}]ds$

$d(s)$, $B(s)$ and $\gamma(s)$ are contributions of the different energy terms → They are normalized

Robust estimator:

To provide reliable estimates of parameters with outliers

Approaches:
- M-estimators:
  - Modify the loss function, gives less weight to outliers
  - Outliers are not removed them
- RANSAC:
  - Use a subset of data points
  - Check which one fit well
  - Outliers are removed

M-estimators
- Mean squared error: $E(\theta) = \sum d^2(x_i : \theta)$
- Robust estimator: $E(\theta) = \sum f_\sigma(d(x_i : \theta))$ → Geman-McClure estimator

Geman-McClure: Loss function to reduce the influence of outliers $f_\sigma = \frac{x^2}{x + \sigma^2}$
- $x \gg \sigma \rightarrow f_\sigma = 1$
- $x \ll \sigma \rightarrow f_\sigma = \frac{x^2}{\sigma^2}$

Larger σ reduce the penalty for large residuals

MSE is more sensitive than Geman-McClure because higher residuals have less impact

- Small σ → − points
- Large σ → + points

Steps m-estimators:
1. Select a subset
2. Select a σ
3. Fit the model to get θ (based on minimizing loss function)
4. Compute the σ using the median distance of points
5. Continue until the objective is decreasing

RANSAC: Uses small subset in hope at least one will no have outliers

Random Sample Consensus

1. Select a subset 2. Train 3. Choose the best → Explain the majority of outliers → Does not have outliers

Parameters:
- n: Define the subset (point for each evaluation)
- d: min points to estimate (2 for line, 3 for plane)
- k: number of trials
- t: distance, threshold to identify inliers (inliers < threshold)

t → median distance
k based on
- p probability not outliers (0.9)
- w Probability any point is inlier (0.8)

Object recognition

Semantic segmentation: Classify each pixel to category

Instance segmentation: Classify multiple object to a category

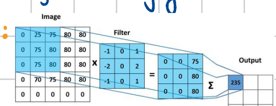
Approaches to object recognition:
- Extract features (SIFT and HOG)
  - SIFT: Detect key points that are invariant to scaling and rotation
  - HOG: Describes object shapes by capturing edge orientation
- Bag to words: Form codebook of visual words
- Eigenfaces: Use PCA to reduce the dimension
- CNN:

Flatten:

1	1	1
4	2	1
0	2	1

→ | 1 | 1 | 1 | 4 | 2 | 1 | 0 | 2 | 1 |

CNN:



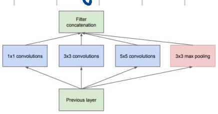
Max Pooling

12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

2 × 2 Max-Pool with stride=2 →

20	30
112	37

- VGG16 → 16 convolution layers
- Inception / GoogleNet: We dont have a pyramid as before, we have multiple receptive
  - Each field returns a result and concatenate the



(a) Inception module, naïve version

RESNET

- Skip connections help with vanishing ingredients

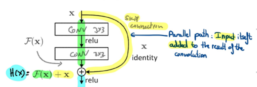


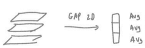
Figure 2. Residual learning: a building block.

DenseNet

* DenseNet implementation:

The main features of the DenseNet architecture include:

1. Dense Connections: Each layer in the network is connected to all the preceding layers, allowing for feature reuse and reducing the number of parameters.
2. Bottleneck Layers: The architecture uses bottleneck layers to reduce the number of featu maps before each dense block, further reducing the number of parameters.
3. Transition Layers: Between each dense block, there is a transition layer that reduces the number of feature maps and spatial dimensions.
4. Global Average Pooling: Instead of using a fully connected layer at the end of the network DenseNet uses global average pooling to reduce the number of parameters and improve regularization.



Forward: From input to output ←
Backforward: Reverse →

Train is harder because the gradients are vanish, when we train the network we push the input forward and then we push the gradients backward. We use the gradients to change the parameters for the next iteration of the training

Normally, at the beginning, the gradients are huge but when we push them, they are becoming smaller and they don't change the parameters. For fixing this, Inception use multiple loss (L1, L2, L3...) → Not training

Localization:

Interception over union (IoU): $\frac{|A \cap B|}{|A \cup B|} = [0, 1]$
- $\cap$ → Interception; $\cup$ → Union
- 0 → mutually exclusive; 1 → The same

Jaccard distance $d_J(A,B) = 1 - IoU = \frac{|A \cup B| - |A \cap B|}{|A \cup B|} = [0, 1]$

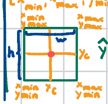
- Precision and recall:

Prediction \ true label	P	N
P	TP	FP
N	FN	TN

$recall = \frac{TP}{total\ positive} = \frac{TP}{TP + FN}$

$Precision = \frac{TP}{total\ P\ Predictions} = \frac{TP}{TP + FP}$

$[x_{min}, x_{max}, y_{min}, y_{max}] \in [0, 1]$



$\hat{y} = [p_c, x_c, y_c, w, h, c_1 \dots c_n]$
- p_c → object probability
- x_c, y_c, w, h → boundary box
- $c_1 \dots c_n$ → class probabilities

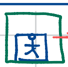
Object detection approaches:
- Find candidates → Sliding window, Grid cells, Region proposals
- Classify candidates
- Possibly refine coordinates

Grid cells: 1. Divide the image into cells 2. Classify between object and BG

Detection:

Regression: To find the box

Classification: Classify the object



→ Conv → Final feature map → L1 Classification, L2 Regression; $L = L1 + L2$

Sliding window: 1. Crop sub window 2. Classify between object and background
→ Identical output of CNN without dense layer its CNN with a single convolutional layer

From world point to image point $\Rightarrow P_i = M \cdot P_w$

- Forward projection: Given world point $\underline{P}_i$ and project matrix M, compute image point P_i
- Calibration: Given world point $\underline{P}_i$ and image point P_i, compute project matrix M
- Reconstruction: Given image point P_i and project matrix M, compute the world point $\underline{P}_i$
- Forward projection is the easiest.
- Reconstruction is the hardest.

$P = K \cdot [R|T] \cdot P$

K = intrinsic camera

R = Rotation camera

$R|T$ = extrinsic camera

P = world point

$K = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}$ $\frac{w}{2}$ $\frac{H}{2}$

$T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$

$M \rightarrow$ projection matrix (3x4)

1. **Rotation about the X-axis by angle θ:**

$$R_x = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

2. **Rotation about the Y-axis by angle θ:**

$$R_y = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

3. **Rotation about the Z-axis by angle θ:**

$$R_z = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

2.2 Given the estimated projection matrix M with rows $\underbrace{(1,2,3,4)}_{m_1}$, $\underbrace{(2,3,4,5)}_{m_2}$, $\underbrace{(3,4,5,6)}_{m_3}$, find the camera parameters u_0 and v_0 (coordinates of the principal point) of the camera.

$u = \frac{m_1 \cdot m_3}{m_3 \cdot m_3}$ $v = \frac{m_2 \cdot m_3}{m_3 \cdot m_3}$

$m_1 = [1, 2, 3, 4]$

$m_2 = [2, 3, 4, 5]$

$m_3 = [3, 4, 5, 6]$

$u = \frac{m_1 \cdot m_3}{m_3 \cdot m_3} = \frac{3+8+15}{9+16+25} = \frac{26}{50}$

$v = \frac{m_2 \cdot m_3}{m_3 \cdot m_3} = \frac{6+8+20}{9+16+25} = \frac{38}{50}$

$$\begin{bmatrix} 3 & 4 & 1 & 0 & 0 & 0 & -3 & -4 & -1 \\ 0 & 0 & 0 & 3 & 4 & 1 & -6 & -8 & -2 \end{bmatrix}$$

$P = (3, 4, 0)$

$p = (1, 2)$

- Sparse stereo matches specific points and dense stereo matches each pixel.
- Sparse can be used for far (between frames) views, while dense should be used for close (between frames) views. Both cannot match uniform patch, invisible in one view, and ambiguity cases.

3.5 Given R_l, T_l (rotation and translation of the left camera with respect to the world) and R_r, T_r (rotation and translation of the right camera with respect to the world), write the expression for the rotation and translation of the right camera with respect to the left camera.

- $R = R_l^T R_r$
- $T = R_l^T (T_r - T_l)$

3.3 Given an axis-aligned stereo pair with corresponding points $(100, 200)$ and $(103, 200)$ in the left and right images, compute the depth (z-coordinate) of the 3D point that produced this projection. Assume the focal length of both cameras is 10, the baseline is 100, and the system uses camera coordinates.

d = right - left

Focal length of the cameras; distance between cameras

$z = \frac{f \cdot T}{d} = \frac{10 \times 100}{(103-100)} = \frac{1000}{3}$

Disparity: difference between x-coord

4.1 Given an axis-aligned stereo system with a focal length of $10\,\text{mm}$ and a baseline of $20\,\text{mm}$, compute the depth of a point with a disparity of $30\,\text{mm}$.

$z = \frac{f T}{d} = \frac{10 \cdot 20}{30} = \frac{20\cancel{0}}{3\cancel{0}}$

4.4 Given corresponding left and right points $(1, 2)$ and $(2, 3)$ respectively, write the respective row in the matrix that must be formed to solve for the fundamental matrix.

d. The left point is $P_l = (x_l = 1, y_l = 2)$ and right point is $P_r = (x_r = 2, y_r = 3)$

$P_r^T F P_l \Rightarrow [x_r x_l \quad x_r y_l \quad x_r \quad y_r x_l \quad y_r y_l \quad y_r \quad x_l \quad y_l \quad 1] = [2 \quad 4 \quad 2 \quad 3 \quad 6 \quad 3 \quad 1 \quad 2 \quad 1]$

OR

$P_l^T F P_r \Rightarrow [x_l x_r \quad x_l y_r \quad x_l \quad y_l x_r \quad y_l y_r \quad y_l \quad x_r \quad y_r \quad 1] = [2 \quad 3 \quad 1 \quad 4 \quad 6 \quad 2 \quad 2 \quad 3 \quad 1]$

4.3 Let F be a fundamental matrix with rows $(1, 2, 3)$, $(2, 3, 4)$, $(3, 4, 5)$. Let $(1, 2)$ and $(2, 3)$ be corresponding left and right points. Compute the value of $p_r^T F p_l$.

$\begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix} \cdot (2, 3, 1)$

$(7, 12, 16) \cdot (2, 3, 1) = 14 + 36 + 16 = 66$

- Fundamental matrix F = 3x3 matrix that relates two points in two different stereo images $P_r^T F P_l = 0$ $\boxed{F \text{ is rank two}}$
- F also gives the corresponding epipolar line in the other image $l_{r/l} = F P_{l/r}$

Rank of a matrix: Number of rows or columns linearly independent, rank 2 of 3 means one row or column can be expressed as a combination of the other 2

2.1 The optical centers of the camera lenses are distinct, and each center projects onto a distinct point in the other camera's image plane. These points, denoted by e_l and e_r, are called epipoles.

The epipolar line is formed by the intersection of the image planes with the epipolar plane. The epipoles may be inside or outside the image.

1. Stereo

1.1 Sparse stereo matches specific points, while dense stereo matches each pixel. Sparse stereo is suitable for far (between frames) views, while dense stereo should be used for close (between frames) views. Both approaches fail in cases of uniform patches, objects invisible in one view, and ambiguous matches.

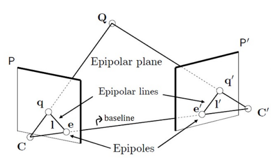
1.2 Answer:

- **NCC (Normalized Cross-Correlation):**

$$\phi(w_1, w_2) = \sum_i \left(\frac{w_1(x_i, y_i) - u_1}{\sigma_1} \right) * \left(\frac{w_2(x_i, y_i) - u_2}{\sigma_2} \right)$$

- **SSD (Sum of Squared Differences):**

$$\phi(w_1, w_2) = \sum_i (w_1(x_i, y_i) - w_2(x_i, y_i))^2$$



Q : 3D world point
P : First Image Plane
P' : Second Image Plane
C : Camera Center of First Camera
C' : Camera Center of Second Camera
q : 2D image point of Q on the first image plane
q' : 2D image point of Q on the second image plane
e : The intersection point of the first image plane with baseline **(epipole)**
e' : The intersection point of the second image plane with baseline **(epipole)**
l : The line on the first image plane connecting the epipole with 2D projected image point q **(epipolar line)**
l' : The line on the second image plane connecting the epipole with 2D projected image point q **(epipolar line)**
baseline : The line joining two camera center points

2.2 Write the expression of the essential matrix E. Given a set of corresponding points p_l, p_r (in camera coordinates), write the epipolar constraint equation using the essential matrix E.

$E = R^T [T]_x$

$P_r^T E P_l = 0$

2.3 Write the expression of the fundamental matrix F. Given a set of corresponding points p_l, p_r (in image coordinates), write the epipolar constraint equation using the fundamental matrix F.

$F = K_r^{-T} E K_l^{-1}$

$P_r F P_l = 0$

Essential matrix E: Describe the geometric relationship between two cameras. Determine the relative pose between two cameras

2.8 Given corresponding points $(100, 200)$ and $(50, 100)$ in the left and right images respectively, write the first line in the matrix that has to be formed to solve for the unknown fundamental matrix using the 8-point algorithm. Do not normalize the points.

Each pair of points contributes one row to the matrix A. The general formula for a row is:

$[x' \cdot x,\ x' \cdot y,\ x',\ y' \cdot x,\ y' \cdot y,\ y',\ x,\ y,\ 1]$

Substitute the given points:

- $x = 100,\ y = 200,\ x' = 50,\ y' = 100$

The row becomes:

$[50 \cdot 100,\ 50 \cdot 200,\ 50,\ 100 \cdot 100,\ 100 \cdot 200,\ 100,\ 100,\ 200,\ 1]$

Step 2: Simplify the values:

$[5000,\ 10000,\ 50,\ 10000,\ 20000,\ 100,\ 100,\ 200,\ 1]$

2.9 To normalize the points:

$$q_i = \frac{P_i - u_p}{\sigma_p}, \quad q_i' = \frac{P_i' - u_p'}{\sigma_p'}$$

Normalization improves numerical stability and accuracy. The fundamental matrix of the original points is recovered as:

$$F = M^T F' M$$

3.3 Let the right image be rotated by R and translated by T with respect to the left image. Let p_l and p_r be corresponding points in the left and right images. Write the matrix that must be formed to solve for the coefficients (a, b, c) of the corresponding triangulated 3D point.

3.3 The matrix A to solve for coefficients $ (a, c, b) $ is:

$$A = \begin{bmatrix} P_l \\ P_l \times RP_r \\ -RP_r \end{bmatrix}$$

Solve using:

$$[a, b, c]^T = A^{-1}T$$

1.4 Write the fundamental motion projection equation relating 3D motion vectors V to 2D projected motion vectors v, the focal length f, and the position of the object point P. Assume the z-coordinate of the object point is z, and that the z-component of the 3D motion vector is V_z. What is the z-component of the projected motion vector v, according to this equation?

1.4 The fundamental motion projection equation is:

$$v = \frac{f}{z}(zV - V_z P)$$

The z-component of v is zero according to this equation.

1.7 The instantaneous epipole coordinates are:

$$x_0, y_0 = \left(\frac{f\tau_x}{\tau_z}, \frac{f\tau_y}{\tau_z}\right)$$

1.8 Motion parallax occurs when objects at different depths move relative to one another due to the observer's motion. The relative motion field equations are:

$$\Delta v_x = (x - x_0)\tau_z\left(\frac{1}{Z} - \frac{1}{Z'}\right)$$
$$\Delta v_y = (y - y_0)\tau_z\left(\frac{1}{Z} - \frac{1}{Z'}\right)$$

1.1 Explain the difference between 3D motion vectors, 2D projected motion vectors, and observed 2D motion vectors (optical flow). Is it possible that motion in 3D will not produce optical flow vectors?

1.1 3D motion vectors represent motion in the real world. 2D projected motion vectors are the projection of 3D motion vectors onto the image plane. Optical flow vectors are the observed 2D motion vectors derived from image sequences.

1.2 What will be the projected motion field in a video taken by a car driving on a straight road and looking to the side (assume that objects in the scene do not move)? Where will projected motion vectors be larger?

1.2 The projected motion field in a video taken by a car driving on a straight road while looking to the side is a parallel motion field with vectors parallel to the car's motion direction. Motion vectors will be larger for world points closer to the car (i.e., smaller z-values).

1.5 Assuming 3D motion with translational velocity τ and rotational velocity ω, write the equation for the projected translational and rotational motion.

1.5 For 3D motion with translational velocity τ and rotational velocity ω, the projected translational and rotational motions are:

- Motion components:

$$v_x = v_x^{(\tau)} + v_x^{(\omega)}$$
$$v_y = v_y^{(\tau)} + v_y^{(\omega)}$$
$$v_z = 0$$

- Translational motion:

$$v_x^{(\tau)} = \frac{\tau_z x - \tau_x f}{Z}$$
$$v_y^{(\tau)} = \frac{\tau_z y - \tau_y f}{Z}$$

- Rotational motion:

$$v_x^{(\omega)} = -\omega_y f + \omega_z y + \frac{\omega_x xy}{f} - \frac{\omega_y x^2}{f}$$
$$v_y^{(\omega)} = \omega_x f - \omega_z x - \frac{\omega_y xy}{f} + \frac{\omega_x y^2}{f}$$

2.1 The optical flow constraint equation (OFCE) is:

$$\nabla I \cdot v = -I_t$$

The basic assumption is that image brightness remains constant at each location of the object.

2.4 Write the objective function of block-based optical flow estimation in a patch. Write the system of equations that must be solved in order to find the optical flow in the patch. What is the purpose of weighted block methods? How do the weights modify the solution?

2.4 The objective function for block-based optical flow estimation is:

$$E(v) = \sum_w (\nabla I \cdot v + I_t)^2$$

The system of equations to solve for optical flow v is:

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i)I_y(q_i) \\ \sum_i I_y(q_i)I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i)I_t(q_i) \\ -\sum_i I_y(q_i)I_t(q_i) \end{bmatrix}$$

In weighted block methods, weights are assigned such that locations closer to the center of the window get higher weights. These weights modify the summations in the equations to emphasize central pixels.

2.2 Explain the aperture problem. What part of the motion can we hope to recover based on a single point?

2.2 The aperture problem arises because the OFCE provides only the motion component projected onto the image gradient direction. From a single point, we can recover motion only in the direction of the gradient.

2.5 Explain the advantage of an affine motion model. Write the objective function for the affine model, then write the solution. How are the motion vectors in a patch recovered once the affine motion parameters are recovered?

2.5 Affine motion estimation does not assume constant optical flow in each window.

Objective:

$$E(\mathbf{a}) = \sum_{(x,y)\in\text{patch}} (I_x u + I_y v + I_t)^2$$

When substituting the affine model:

$$E(\mathbf{a}) = \sum_{(x,y)\in\text{patch}} (I_x(a_1 + a_2 x + a_3 y) + I_y(a_4 + a_5 x + a_6 y) + I_t)^2$$

Solution: Affine flow vectors are computed using the estimated affine parameters a. To compute a, solve:

$$\begin{bmatrix} \sum I_x^2 & \sum I_x^2 x & \sum I_x^2 y & \sum I_x I_y & \sum I_x I_y x & \sum I_x I_y y \\ \sum I_x^2 x & \sum I_x^2 x^2 & \sum I_x^2 xy & \sum I_x I_y x & \sum I_x I_y x^2 & \sum I_x I_y xy \\ \sum I_x^2 y & \sum I_x^2 xy & \sum I_x^2 y^2 & \sum I_x I_y y & \sum I_x I_y xy & \sum I_x I_y y^2 \\ \sum I_x I_y & \sum I_x I_y x & \sum I_x I_y y & \sum I_y^2 & \sum I_y^2 x & \sum I_y^2 y \\ \sum I_x I_y x & \sum I_x I_y x^2 & \sum I_x I_y xy & \sum I_y^2 x & \sum I_y^2 x^2 & \sum I_y^2 xy \\ \sum I_x I_y y & \sum I_x I_y xy & \sum I_x I_y y^2 & \sum I_y^2 y & \sum I_y^2 xy & \sum I_y^2 y^2 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \end{bmatrix} = \begin{bmatrix} -\sum I_t I_x \\ -\sum I_t I_x x \\ -\sum I_t I_x y \\ -\sum I_t I_y \\ -\sum I_t I_y x \\ -\sum I_t I_y y \end{bmatrix}$$

The affine flow vecctors are then computed using:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} a_1 \\ a_4 \end{bmatrix} + \begin{bmatrix} a_2 & a_3 \\ a_5 & a_6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

F [A] The projected motion field v is always a radial motion field.
not

F [B] The Horn-Schunck algorithm for optical flow estimation considers the estimation of each optical flow vector at each location independent of all other locations.
It enforces Smoothness by coupling the flow vectors of neighboring pixels

T [C] The projected motion vectors v of a pure translation in which there is a translation in Z (i.e. $T_z \neq 0$) form a radial motion field.

F [D] The relative motion field of instantaneously coincident points is always a radial motion field.
not
(It depends on the specific motion and scene structure, not necessary radial

F [E] When decomposing the projected motion field according to the translational and rotational components of the 3D motion V, the translational component does not depend on the depth Z.
Translation motion depends on the depth: closer objects exhibit larger motions

T [F] Assuming that the motion is constant in a neighborhood allows finding the optical flow vector in the center of the neighborhood regardless of the content of the neighborhood.

T [G] The Horn-Schunck algorithm for optical flow estimation assumes that the image brightness at each object point is constant.

F [H] The Optical Flow Constraint Equation (OFCE) at a given point allows computing the optical flow vector at that location without any ambiguity.
→ Provides one equation with two unknowns (v_x and v_y) leading to ambiguity

T [I] The Optical Flow Constraint Equation (OFCE) assumes that the image brightness at each object point is constant.

RANSAC handle dataset with a significant proportion of outliers by iteratively fitting a model to random subsets of data and selecting the one with the most outliers

T [A] Using the RANSAC algorithm is a good approach for trying to deal with outliers.

Threshold is typically predefine or adjusted manually
F [B] The RANSAC algorithm continuously updates the distance threshold used to detect inliers.

minimizes
F [C] The best model l is the one that maximizes $E(l)$.

Based on the estimated inliers and the level of confidence
T [D] The RANSAC algorithm continuously updates the number of trials that has to be conducted.

T [E] The gradient of $E(l)$ is commonly used in trying to find the optimal l.

Outliers lead to inaccurate results
F [F] Outliers are important for an accurate model and should be included when computing the model parameters.

RANSAC deals with outliers not with gaussian noise
F [G] Using the RANSAC algorithm is a good approach for trying to deal with Gaussian noise.

RANSAC uses a fixed sample size
F [H] The RANSAC algorithm continuously updates the number of points that has to be drawn at each evaluation.

Geman reduces their importance, for this reason gives less weight
F [I] The Geman-McClure function used in M-estimators gives higher weight to large errors.

They can be outside, depending on relative positions and orientations
F [A] The epipoles must always be inside the images.

T [B] Given the point p_l in the left image the right epipolar line in the right image corresponding to p_l is given by Fp_l.

$E = K_r^{*T} F K_l^*$
T [C] The matrices K_l^*, K_r^*, R, and T, are all necessary for computing the essential matrix E.

It always rank 2. Full rank is 3
F [D] The matrix F must always be a full rank matrix.

T [E] Normalizing the set of corresponding points helps in obtaining a more stable estimate of the fundamental matrix.

T [F] The fundamental matrix may be estimated based on a set of 6 corresponding points in the left and right images.

F [G] Given the point p_l in the left image the right epipolar line in the right image corresponding to p_l is given by $p_r^T F$.
is computed as Fp_l not $p_r^T F$

These are used for E, not for F
F [H] The matrices K_l^*, K_r^*, R, and T, are all necessary for computing the fundamental matrix F.

T [I] The epipoles may be computed by applying SVD to the fundamental matrix F.

A 1D gaussian filter can be applied twice (along the column and along the rows) to simulate the behaviour of a 2D filter
F [A] It is not possible to correctly use G_1 for producing a Gaussian image pyramid because it is a 1D filter.

Laplacian filter creates Laplacian pyramids
F [B] It is possible to produce the Gaussian pyramid P by convolving the image with a Laplacian filter instead of a Gaussian filter.

Gaussian filter smooth images and reduce noise, but do not enhance edges
Laplacian or Sobel do it
F [C] The Gaussian filter G_2 enhances edges in the image and so is a good way to perform edge detection.

T [D] The Gaussian pyramid P enables analyzing the image at multiple scales.

F [E] The size of the array that is used to store G_2 is not sufficient for accurately representing G_2.

T [F] The size of the array that is used to store G_1 is sufficient for accurately representing G_1.

Depending on the implementation
F [G] The Gaussian pyramid P consumes less memory for storing an image compared with storing the image in a single array.

Is computationally less effective than aply an 2D filter ones
T [H] Applying G_1 to the image twice can be used instead of applying G_2 once, but this is less efficient.

It is less effective at removing salt-pepper noise
F [I] The Gaussian filter G_2 is an effective filter for removing impulsive (salt & pepper) noise from the image.

Question 4 ♣ Let R be a 4×4 3D rotation matrix representing rotation by 90 degrees about the Z-axis in homogeneous coordinates. Compute a as the sum of elements in R. Let T be a 4×4 3D translation matrix representing translation by $[1, 2, 3]$ in homogeneous coordinates. Compute b as the sum of elements in T. Mark **all** the statements that are **correct**.

$$R = \begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad T = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \boxed{a=2,\ b=10}$$

Question 8 ♣ Let $S = \{(1,1), (2,2), (3,3)\}$ be a set of points. Let $l = (-1, \ 1, \ 0)$ be the parameters of a line that are computed based on S using a line fitting algorithm. Compute the error produced when using l to model the line containing the points S and set a as this error. Compute the normal N of this line, then compute b as the sum of the elements of the normalized vector N. Mark **all** the statements that are **correct**.

$S = \begin{bmatrix} 1 & 1 \\ 2 & 2 \\ 3 & 3 \end{bmatrix}$ $\quad ax+by+c=0 \quad [a,b,c] = [-1,1,0]$

Normal vector $= (-1, 1)$ $\quad |d| = \sqrt{(-1)^2+(1)^2} = \sqrt{2}$

Normal vector normalized $\Rightarrow \left(\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$ $\quad \boxed{b=0}$

For (1,1): $\text{Error} = \frac{|ax+by+c|}{\sqrt{a^2+b^2}}$

$\text{Error} = \frac{-1\cdot 1 + 1\cdot 1 + 0}{\sqrt{2}} = \frac{0}{\sqrt{2}} = 0$

For (2,2): $\text{Error} = \frac{-1\cdot 2 + 1\cdot 2 + 0}{\sqrt{2}} = 0$

For (3,3): $\text{Error} = 0$ $\quad \boxed{a=0}$

Question 3 ♣ Let $P = (1,2,3)$ be a world point and $p = (1,1)$ be a corresponding image point. Let M be the unknown perspective projection matrix. Let m_1^T, m_2^T, m_3^T be the rows of the matrix M. Let x be a vector composed of the three rows of M. Let $Ax = \mathbf{0}$ be the system of 2 equations that is formed using the points P and p to solve for x. Write the elements of the 12×2 matrix A using the corresponding points P and p. Compute a as the sum of the elements in this matrix. Let the the first, second, and third rows of M be $[2, \ 0, \ ,1 \ ,0]$, $[1, \ 2, \ ,0 \ ,1]$, $[0, \ 1, \ ,0 \ ,2]$, respectively. Let ρ be the unknown scale of the estimated matrix M. Compute b as the absolute value of ρ. Mark **all** the statements that are **correct**.

$$A = \begin{bmatrix} X & Y & Z & 1 & 0 & 0 & 0 & 0 & -xX & -xY & -xZ & -x \\ 0 & 0 & 0 & 0 & X & Y & Z & 1 & -yX & -yY & -yZ & -y \end{bmatrix} \quad P=(X,Y,Z) \quad p=(x,y)$$

$$\Rightarrow A = \begin{bmatrix} 1 & 2 & 3 & 1 & 0 & 0 & 0 & 0 & -1 & -2 & -3 & -1 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 1 & -1 & -2 & -3 & -1 \end{bmatrix} \quad \boxed{a=0}$$

ρ is the determinant 3x3 formed by the three rows of $M \Rightarrow \rho = \begin{vmatrix} 2 & 0 & -1 \\ 1 & 2 & 0 \\ 0 & 1 & 0 \end{vmatrix} = 0+0-1-0-0-0 = -1 \rightarrow$ Remember to do the absolute value $\boxed{b=1}$

Question 7 ♣ Assume an axis-aligned stereo system where the distance between the optical centers is $T = 4$, the focal length of both cameras is $f = 2$. Let $p_l = (1,2)$ and $p_r = (2,2)$ be a set of corresponding points in the left and right images respectively (in camera coordinates). Compute a as the z-coordinate of the 3D point which produced p_l and p_r (in camera coordinates). Assume that the intrinsic camera parameters matrix is given by K^* where the first, second, and third rows of this matrix are given by $[2, \ ,0 \ ,0 \ ]$, $[0, \ ,2 \ ,0 \ ]$, $[0, \ ,0 \ ,1 \ ]$, respectively. Let $\bar{p}_l$ be the image coordinates of the point p_l. Use the matrix K^* to compute $\bar{p}_l$. Compute b as the sum of the x- and y-coordinates of $\bar{p}_l$. Mark **all** the statements that are **correct**.

$z = \frac{fT}{d} \Rightarrow \frac{2\cdot 4}{1} = \frac{8}{1} = -8$ $\quad \boxed{a=8}$

$\bar{p}_l = K p_l = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 1 \end{bmatrix}$ $\quad \boxed{b=2+4=6}$

Question 10 ♣ Let M be a 3×4 projection matrix of a camera with a focal length of $f = 2$ with no scale, skew, translation, or rotation. Assuming that the $(3,3)$ element of M is 1, compute a as the sum of elements in M. Let $P = (1,2,4)$ be a point in 3D. Compute the coordinates of the 2D point $p = (x,y)$ that is produced by projecting P using M, then compute b as the sum of elements in p (i.e. $x+y$). Mark **all** the statements that are **correct**.

$$M = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \Rightarrow \boxed{a=5} \quad \begin{bmatrix} 2 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 2 \\ 4 \\ 1 \end{bmatrix} = \begin{bmatrix} 2 \\ 4 \\ 4 \end{bmatrix} = \begin{bmatrix} 2/4 \\ 1 \end{bmatrix} \Rightarrow \boxed{b=1.5}$$

Question 1 ♣ Let W be a 3×3 window centered about the pixel p. Let the gradient vectors in this window (top to bottom, left to right) be $\{(0,0), (0,1), (0,1), (0,0), (-1,0), (0,0), (0,0), (-1,0), (0,0)\}$. Compute the correlation matrix C in this window without normalizing it then compute a as the sum of of entries in this matrix. Let Q be the matrix $\sum_i \nabla I(p_i) \nabla I(p_i)^T x_i$ where $\nabla I(p_i)$ are gradients at locations in W and x_i are the x-coordinates of corresponding locations p_i in W. Assume that the first row of Q is given by $[1 \ \ 2]$ and that the second row of Q is given by $[2 \ \ 3]$. Using the matrix Q and the matrix C you computed before, compute the location t of the corner in this window. Compute b as the sum of the x- and y-coordinates of the location t. Mark **all** the statements that are **correct**.

$C = \sum_i \nabla I(p_i) \cdot \nabla I(p_i)^T$

Gradient (0,0) $\Rightarrow \begin{bmatrix} 0 \\ 0 \end{bmatrix} \cdot [0,0] = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \cdot 5 = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

Gradient (0,1) $\Rightarrow \begin{bmatrix} 0 \\ 1 \end{bmatrix} \cdot [0,1] = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix} \cdot 2 = \begin{bmatrix} 0 & 0 \\ 0 & 2 \end{bmatrix}$

Gradient (-1,0) $\Rightarrow \begin{bmatrix} -1 \\ 0 \end{bmatrix} \cdot [-1,0] = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \cdot 2 = \begin{bmatrix} 2 & 0 \\ 0 & 0 \end{bmatrix}$

$\Rightarrow \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix} \Rightarrow \boxed{a=4}$

$t = Q^{-1} \cdot$ sum of rows of $C \rightarrow \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

$Q^{-1} = \frac{1}{\det Q} \cdot \begin{bmatrix} q_{22} & -q_{12} \\ -q_{21} & q_{11} \end{bmatrix} = -1 \cdot \begin{bmatrix} 3 & -2 \\ -2 & 1 \end{bmatrix} = \begin{bmatrix} -3 & 2 \\ 2 & -1 \end{bmatrix}$

$\begin{vmatrix} 1 & 2 \\ 2 & 3 \end{vmatrix} = 3-4 = -1$

$\begin{bmatrix} -3 & 2 \\ 2 & -1 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 2 \end{bmatrix} = \begin{bmatrix} -2 \\ 2 \end{bmatrix} \Rightarrow \boxed{b=0}$