# Assignment 4 Answers

1 a) Outliers are points those are very different from all other points. The problem associated with outliers is that when the model is fit considering the outliers it results in a wrong solution.

b) Objective function used for robust estimation is:

$$E(\theta) = \sum_{i=1}^{n} \rho_\sigma (d(x_i, \theta))$$

In the standard least squares objective function i.e.

$$E(\theta) = \sum_{i=1}^{n} d(x_i, \theta)^2$$

We take the individual errors and square them. ($\rho_\sigma(x) = x^2$)

Where as in the robust estimation $\rho_\sigma(x) = \dfrac{x^2}{x^2+\sigma^2}$

c) Geman-McClure: $\rho_\sigma(x) = \dfrac{x^2}{x^2+\sigma^2}$ , when $x >> \sigma ; \rho_\sigma = 1$

$x << \sigma ; \rho_\sigma = \dfrac{x^2}{\sigma^2}$

The advantage of this function is that it is not affected by outliers. Using this function the maximum weight the outliers can get is 1, where as in the standard least square function the weight given to outliers is $x^2$

bandwidth parameter $\sigma$ can be adjusted in a iterative manner using below steps.

1. draw a large subset of points uniformly at random.
2. Fit model using robust estimation, give $\theta_n$
3. Compute $\sigma_n = 1.5 * \text{median}(d(x_i, \theta_n))$
4. repeat the process while $(\theta_n - \theta_{n-1}) >$ Threshold.

In the process, we start with a large ~~sign~~ $\sigma$

i.e $\sigma_n = 1.5 * \text{median}(d(x_i, \theta_n))$

and as we fit better and better model, the median of the points decreases and in turn $\sigma$ decreases as we estimate $\sigma$ as $1.5 * \text{median}(d(x_i, \theta))$

d) Principle of the RANSAC algorithm is to use minimum number of points to fit the model and repeat this process several times and choose the best model after many trials. Number of points drawn at each attempt should by small, because there are less chance of getting outliers and atleast in one of my many traids ~~[illegible]~~ will lead to a better model.

e) Parameters of RANSAC algorithm:

$n \rightarrow$ number of points to draw at each evaluation

$d \rightarrow$ minimum number of points needed.

$k \rightarrow$ Number of trials / evaluations.

$t \rightarrow$ distance to identify outliers.

Formula for estimating the number of ~~traid's~~ trials, K:

$$K = \frac{\log(1-p)}{\log(1-w^n)}$$

where p : probability that atleast one of the trials will succeed

w : probability than a point is an inlier.

n : number of points to draw at each trial.

We start with $w = 0.5$ and at each iteration.

we update w as $\frac{\text{Number of inliers}}{\text{Number of points}}$.

f) Objective of image segmentation is to seperate foreground from the background.

In agglomerative (merge) approach, we start with each pixel in a different cluster and merge iteratively based on the distance / similarity of the feature vectors.

We make the clusters denser by merging similar pixels together.

In split approach, we start with having all pixels in a single cluster and iteratively split the cluster by looking at the distance/similarity of (pixels) feature vectors.

We decrease the size of clusters by removing the pixels which do not belong to a particular ~~pixels~~ clusters.

9) K-means algorithm for segmentation

- Select K, the number of clusters to be formed.
- Start with initial guess of K means $\{m_i\}_{i=1}^{K}$. We randomly choose some pixels to be the mean. Here we make sure that the means are seperated enough to span the image.
- repeat until stopping criteria is meet i.e, mean do not change.
  - for each pixel, assign the pixel to the cluster (label) nearest to it

$$l_i = \underset{j \in (1,K)}{\arg\min} \|f_i - m_j\|^2 ;$$

$f_i$: feature vector of ith pixel

$m_j$: mean of jth cluster.

  - Calculate the new mean of the cluster as

$$m_j = \frac{\sum_{i \in S_j} f_i}{\text{number of pixels in } S_j}$$

; $S_j$ is all the pixels labeled $l_j$

Mixture of gaussian algorithm for segmentation.

the process in mixture of gaussian is same as that of K-means.

The difference is in the distance measure used to assign pixels to the cluster centers.

mixture of gaussian ~~use~~ Computes the distance as.

$$d = (f_i - m_j)^T \Sigma_j^{-1} (f_i - m_j)$$

where $\Sigma_j$ is the covariance matrix.

and

$$\Sigma_j = \frac{\sum_{i \in S_j} (f_i - m_j)(f_i - m_j)^T}{\text{number of pixels in } S_j} ; \quad m_j = \frac{\sum_{i \in S_j} f_i}{\# S_j}$$

h) Mean-shift algorithm for segmentation

- Similar to the K-means algorithm in the steps.
- difference is in ~~the~~ calculating the mean of cluster:

$$m_j = \frac{\sum_{i \in S_j} w(f_i - m_j) f_i}{\sum_{i \in S_j} w(f_i - m_j)} \quad ; \quad w(f_i - m_j) = \text{exponent}(-\|f_i - m_j\|)$$

When recomputing the mean of clusters, we give weight for each pixel belonging to the cluster based on its distance to the previous mean of the cluster.

- In K-means and mixture of gaussian, all pixel belonging to the cluster get equal weight, whereas in the mean-shift algorithm, pixels closer to the mean are weighted higher than the ones farther. The idea is that closer the pixel/point is to the mean, the more it should affect the mean, than the ones farther.
- Mean-shift finds cluster center as peaks of histogram.

2) a) Projection equation $p = MP$

Forward projection: is to find the image coordinate of the 3D object given the coordinates of the object in world (3D) and the projection matrix $M$.

Camera Calibration: given the image coordinate and the world coordinates of the object, find the camera parameters (internal & external) used in the projection.

Reconstruction: given the image coordinates of object, $p$, and the projection matrix, $M$, find the world coordinates (3D) of the object.

Forward projection is the easiest as there is no ambigious decision to make i.e, each point in 3D corresponds to a single point in 2D.

Reconstruction is the most difficult, because we need to add the information we already lost from going to 2D from 3D and each point in 2D can represent a ~~plane~~ line in 3D, which make it ambiguous.

b) $\{p_i\}_{i=1}^{n} \leftrightarrow \{P_i\}_{i=1}^{n}$ ← necessary input for camera calibration

$\{x_i, y_i\}_{i=1}^{n} \leftrightarrow \{X_i, Y_i, Z_i\}_{i=1}^{n}$

for camera calibration, we need the corresponding points in both 2D and 3D.

c) Steps in the non-coplanar calibration algorithm.

1. given image points(p) & world points(P), Estimate the (3x4) Projection matrix M, using $p = MP$
2. Find the camera parameters, internal ($K^*$) and external ($R^*$ and $T^*$) using the estimated projection matrix in step 1, as we know that $M = K^*[R^*|T^*]$

d) $M = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 0 & 3 & 4 \\ 1 & 1 & 1 & 1 \end{bmatrix}$ $P_i = \underset{3D}{\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}} \Rightarrow P_i = \underset{3DH}{\begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix}}$

$p_i = M P_i \quad := \begin{bmatrix} 1 & 2 & 3 & 4 \\ 1 & 0 & 3 & 4 \\ 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 1+4+9+4 \\ 1+0+9+4 \\ 1+2+3+1 \end{bmatrix} = \underset{2DH}{\begin{bmatrix} 18 \\ 14 \\ 7 \end{bmatrix}} = \underset{2D}{\begin{bmatrix} 18/7 \\ 2 \end{bmatrix}}$

e) for point 1 $\underset{2n\times 12}{\begin{bmatrix} P_1^T & 0^T & -x_1 P_1^T \\ 0^T & P_1^T & -y_1 P_1^T \\ & & \end{bmatrix}} \underset{12\times 1}{\begin{bmatrix} [m_1] \\ [m_2] \\ [m_3] \end{bmatrix}} = \underset{2n\times 1}{\begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}}$

Given $(1,2,3) \leftrightarrow (100, 200)$

$\Rightarrow \begin{bmatrix} 1 & 2 & 3 & 1 & 0 & 0 & 0 & 0 & -100 & -200 & -300 & -100 \\ 0 & 0 & 0 & 0 & 1 & 2 & 3 & 1 & -200 & -400 & -600 & -200 \\ & & & & & & & & & & & \end{bmatrix}$

f) Minimal number of points that is necessary to be able to find a unique solution for M is 6

Solution is obtained by performing singular value decom- -position on the $2n \times 12$ matrix and taking the last column of the matrix $V$

where $A = UDV^T$ ; A is our $2n \times 12$ matrix

ie $12 \times 12$ matrix formed by the equations.

g) The principal that is used to extract the unknown parameters from the projection matrix M is :

$$M = R^*[K^* | T^*]$$

We use the fact that the rotation matrix has the orthogonal vectors along the rows.

We exploit this fact by taking the dot product and cross product of the rows in the M, thereby cancelling out some unknowns.

h) To compute the quality of the projection matrix M that we estimated, we use the $\{p_i\}_{i=1}^n \leftrightarrow \{P_i\}_{i=1}^n$ correspondence of the image & world points given as input. We use the estimated projection Matrix M and $p_i = M P_i$ to find the image points corresponding to the world points and compare them with the known/correct point.

We calculate the quality or error as:

$$E(K^*, R^*, T^*) = \sum_{i=1}^{n} \left(x_i - \frac{m_1^T P_i}{m_3^T P_i}\right)^2 + \left(y_i - \frac{m_2^T P_i}{m_3^T P_i}\right)^2$$

We want this error to be low...

Scanned by CamScanner

i) Principal of planar calibration :

We show the calibration target to the Camera. We have to show it to the camera atleast 3 times, as one view is not enough to ~~find the points~~. do the calibration.

Approach :

1. estimate 2D homography (2D projective map) between calibration target and image.

2. Estimate the intrinsic camera parameters from several views.

3. Compute extrinsic parameter for any of the view.

In non-Coplanar Calibration one view of the Calibration target is enough to Calibrate the Camera parameters., whereas for planar Calibration, we need atleast 3 different view of the calibration target.

j) 2D homography (2D projective map) H transforms the 2DH point to 2DH itself and H is a 3X3 matrix

$$H = K^* [r_1 \; r_2 \; T^*]$$

Whereas projection matrix M transforms the 2DH point to 3DH. and M is a 3X4 matrix.

$$M = K^* [r_1 \; r_2 \; r_3 \; T^*]$$

The assumption that is used to make sure that we deal with homography matrices is that the z coordinates of the points are 0.

ie $P_i = \begin{bmatrix} x_i \\ y_i \\ 0 \\ 1 \end{bmatrix}$

3DH