Solver: Leroy Low

1)

a) **The new image point will be at P.** The two locations have effectively swapped.
Let original object point be at P1. The original image will be at Q1.

$$\frac{1}{P1} + \frac{1}{Q1} = \frac{1}{f} \quad (1)$$

Now, new object point is at P2, and image is at Q2.

$$\frac{1}{P2} + \frac{1}{Q2} = \frac{1}{f}$$

Since P2 = Q1:

$$\frac{1}{Q1} + \frac{1}{Q2} = \frac{1}{f} \quad (2)$$

Substitute (1) into (2)

$$\frac{1}{P1} + \frac{1}{Q1} = \frac{1}{Q1} + \frac{1}{Q2}$$
$$\frac{1}{P1} = \frac{1}{Q2}$$
$$Q2 = P1$$

b) **3D imaging is the recreation of a 3-dimensional image from 2D cameras.** One 3D imaging technique is using stereo imaging.
In stereo imaging, 2 cameras are used to take images of a scene. One of the cameras is displaced slightly to one side, so that there is parallax for objects of different distances. By measuring the difference in parallax in both images, the distance of the object in the image can be measured, and a 3D image can be reconstructed.

c) Freckles can be removed through a **Median filter.** In a Median filter, for every pixel in the original image, the surrounding region of pixels forming a n x n grid is considered. The new value of the pixel will be set to the median of these pixel values. This has an effect of smoothing away speckle type of noise, and freckles approximates speckle noise.
The parameter that can be tweaked is the size of the n x n grid, and to pick an appropriate size n, different values of n should be tested, and the smallest value of n which can remove the freckles should be used, to ensure minimum loss of image quality.
Note: Gaussian/Box filters COULD work too, but gaussian filters work best against gaussian noise, not speckle noise. Preferred answer is still Median filter as the question specifies freckles.

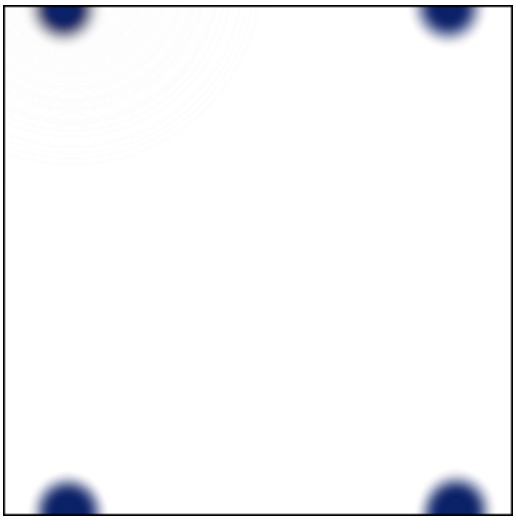
d) Box filtering takes an average value of the entire region.
Let's consider the horizontal axis:
Since the horizontal axis consists of intensities increasing linearly:
1 | 2 | 3 | **4** | 5 | 6 …
For any pixel, the average value of the region around it is the same as its value.
Eg. For pixel with value **4**, region around it is [3, 4, 5], and average is 4.
For the vertical axis, it is constant, thus it does not affect the value.

For the boundary pixels, it depends on how the filter algorithm calculates regions outside the image. If image is zero padded, the image intensity will be affected by the zero values.
This property is useful in special filtering, to filtering noise from large continuous gradients in images, for example, blue skies.

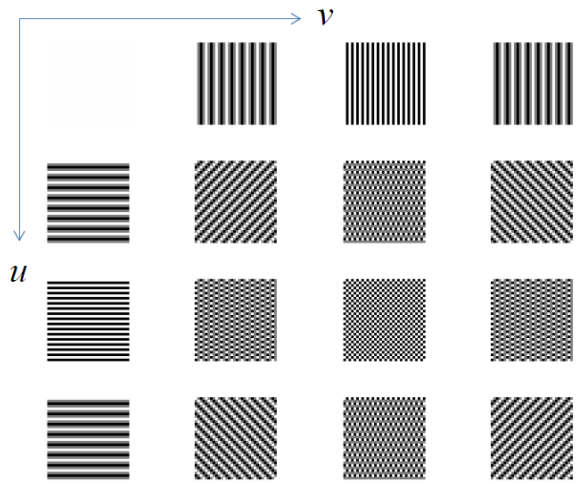
e) **Yes.** Gaussian filtering is similar to box filtering, except that pixels closer to the centre are weighted higher. It still represents the average value of the region of pixels around it, and the filter is symmetrical on the left and right of the original pixel.

2)

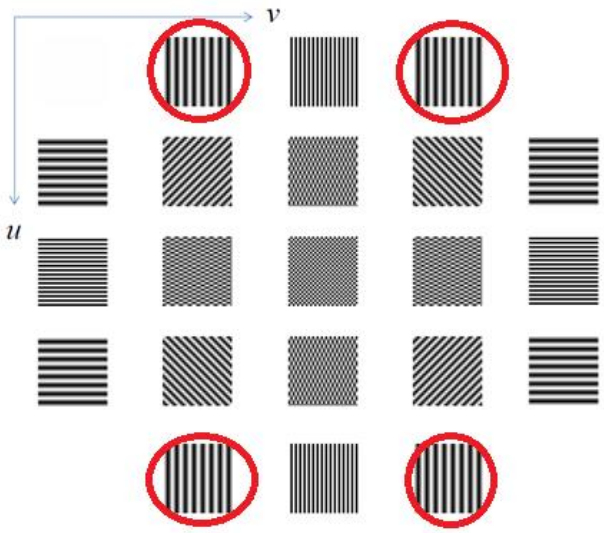
a)



The spectrum will have 4 peaks at the top and bottom of the image, offset slightly from the corners.
The banding in the image is vertical, which will form the majority of the spectrum. As it is vertical banding, the vertical component (u) of the most intense points are going to be 0, while the horizontal component (v) is some value between 0 and the midpoint.
Solvers Opinion/Note: I understand this question is rather technical, but I'll try to explain this below. But first, here are the atomic components of a Fourier transform.



In the context of a Fourier Transform, an image is split into multiple "atomic" images, represented by the squares above. The idea is that any image can be created through summing these "atom" images with different weights. Fourier transform's purpose is to convert any image into its "atoms".
Note that the question paper Figure Q2 has very strong vertical banding, which is very similar to these particular atoms:



These values thus likely have very high weightage, and thus have very high values..
Another thing to note is that the grid is actually symmetrical but the last row and col is omitted in the notes. Although technically incorrect, I have re-added the last row and column to make the symmetry and answer below more obvious.

If you assigned each atom a weightage (0 for low values, 1 for high values), you would get this:

0	1	0	1	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	1	0	1	0

Now this looks very similar to the sketch above!

b) The most intense regions and its surrounding areas of the Fourier power spectrum above should be set to 0. This will remove the vertical banding by removing the components (atoms) that form the banding. When the reverse Fourier transform is subsequently applied, the image will no longer have banding. Unfortunately, any details at those banding frequencies will also be lost.

c) The steps of the Canny Edge Detector are:
1: **Gaussian Edge Filtering** – Image is filtered twice by x and y derivatives of Gaussian to produce edges
2: **Non-Maximal Supression** – Broad filtered edges are reduced to single pixel wide paths
3: **Hysterisis Thresholding** – Pixels above threshold th are set to 1, pixels below threshold tl are set to 0, and pixels between tl and th are set to 1 only if its neighbor is set to 1.
The purpose of the first step is to **extract edges**. The horizontal filter will extract horizontal edges, while the vertical filter will extract vertical edges. By square rooting the squares of each image, the final edge image can be produced.

d) The vertical banding lines may also be detected. This could be because the canny edge detection is also detecting the less intense vertical edges. To solve this, a smaller derivative of Gaussian filter should be used, ie. a **smaller sigma value**. This should be to a value low enough that it can pick up the horizontal line but not the vertical banding.

e) To form a continuous line, a **Hough transform** can be used to link up the broken lines by finding the equation of the line.
First, all edge points are taken, and converted using Hough transform into sinusoidal lines in parameter space. The point where most of the lines intersect will be the parameters for the line equation representing the broken lines, and the line equation can be used to rebuild the line.

3)

a) In K-means clustering,
1: k seeds (cluster centres) are initialized
2a: Assign each sample to the nearest cluster centre
2b: Recalculate the cluster centres for each cluster
3: Repeat 2a and 2b until the cluster centres no longer change

b) **Round 1:**
Initial centroids: [1 1], [1 2]
Distance Table (Euclidian Distance):

	Centroids	
Data Points	[1 1]	[1 2]
[1 1]	**0**	1
[1 2]	1	**0**

[3 4]	3.6	**2.8**
[4 5]	5	**4.2**

Group each data point to the closest centroid:

Centroids	[1 1]	[1 2]
Data point	[1 1]	[1 2] [3 4] [4 5]

Round 2:

New centroids: [1 1], [2.67 3.67]

	Centroids	
Data Points	[1 1]	[2.67 3.67]
[1 1]	**0**	3.1
[1 2]	**1**	2.4
[3 4]	3.6	**0.5**
[4 5]	5	**1.9**

Group each data point to the closest centroid:

Centroids	[1 1]	[2.67 3.67]
Data point	[1 1] [1 2]	[3 4] [4 5]

Round 3:

New centroids: [1 1.5], [3.5 4.5]

	Centroids	
Data Points	[1 1.5]	[3.5 4.5]
[1 1]	**0.5**	4.3
[1 2]	**0.5**	3.5
[3 4]	3.2	**0.7**
[4 5]	4.6	**0.7**

Group each data point to the closest centroid:

Centroids	[1 1]	[3.5 4.5]
Data point	[1 1] [1 2]	[3 4] [4 5]

There are no changes in the assignment of data points to centroids from **Round 2**. Thus, K-means ends here.

c) Multiple **filter banks** can be applied to an image. For each pixel in the image, the values from each of the corresponding filter banks can be used for k-means clustering, thus k-means clustering could be used to split an image into different image segments. This works if the image is to be segmented by texture.

d) The **value of k needs to be known** (ie, the number of segments).
k-means can only cluster values that are approximately **spherical**/globular and might not work for every image.
Performance of finding individual clusters depends on the starting seeds, so k-means might have to be **run multiple times**.

4)

a)

$$\begin{bmatrix} kx \\ ky \\ k \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$\begin{bmatrix} k*24 \\ k*36 \\ k \end{bmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 400 \\ 600 \\ 1200 \end{bmatrix}$$

From matrix:
f*400 + 0*600 + 0*1200 = 24*k => 400f = 24*k => 100f = 6k
0*400 + f*600 + 0*1200 = 36*k => 600f = 36*k => 100f = 6k
0*400 + 0*600 + 1200 = k => k = 1200
100f = 6*1200
f = 72mm

b) Using similar triangles:

$$\frac{Xl}{xl} = \frac{Z}{f}$$

$$\frac{-Xr}{-xr} = \frac{Z}{f}$$

$$\frac{Xl}{xl} + \frac{-Xr}{-xr} = \frac{Z}{f} + \frac{Z}{f}$$

$$\frac{Xl - Xr}{xl - xr} = \frac{Z}{f}$$

$$\frac{T}{xl - xr} = \frac{Z}{f}$$

$$Z = \frac{fT}{xl - xr}$$

Disparity can be calculated from the equation, substitute values in:

$$xl - xr = \frac{fT}{Z}$$

$$disparity = \frac{10 * 20}{50} = \mathbf{0.6}$$

c) Refer to lecture notes: let disparity be w:

$$Z = \frac{fT}{w}$$

$$\frac{dZ}{dw} = -f\frac{T}{w^2}$$

$$|\delta Z| = \left|\frac{dZ}{dW}\delta w\right| = \left|f\frac{T}{w^2}\delta w\right|$$

$$= \left|f\frac{T}{\left(\frac{fT}{Z}\right)^2}\delta w\right|$$

$$|\delta Z| = \left|\frac{Z^2}{fT}\delta w\right|$$

When T is smaller (Camera displacement is smaller), δZ is larger, decreasing the accuracy of Z. When Z is higher (distance is further), δZ is larger, thus accuracy of Z is lower.

d) Precision is the **ratio of correct observations (True positives) to total predicted positives (True Positive + False Positive).** A higher precision means a higher proportion of data predicted to be true are actually true.
Recall is the **ratio of correct observations (True Positives) to total actual positives (True positive + False Negative).** A higher recall means a higher proportion of actually true data was determined to be true.
F1-score is a **weighted average of Precision and Recall**, derived by:
2*(Recall*Precision)/(Recall+Precision). A higher F1 score means overall better performance by the classifier.

--End of Answers--