Cairo University
Faculty of Engineering
Computer Engineering Department

Cairo University

Remote Sensing and Satellite Imagery

Dr. Sandra Wahid

Sources of Geometric Distortion

el map msh sura, el map hya coordinates, fa bst5demha 34an a2dr a3ml correction beha.

- Sources of geometric distortion include:
  - The rotation of the earth during image acquisition.
  - Variations in platform altitude and velocity.
  - The wide field of view of some sensors.
  - The curvature of the earth.
  - Sensor non-idealities.

- There are two techniques that can be used to correct the various types of geometric distortion present in digital image data:

Model the nature and magnitude of the **sources of distortion** and use the model to establish correction formulas.

This approach is effective when the types of distortion are well characterized, such as that caused by earth rotation.

Establish **mathematical relationships between the addresses of pixels in an image and the corresponding coordinates of those points on the ground** (via a map)→ **Image-to-Map Registration/Rectification**

This approach is the **most commonly used** and is independent of the analyst's knowledge of the source and type of distortion.

Use of Mapping Functions for Image Correction

- An assumption made in this procedure is the availability of a map of the region covered by the image, which is correct geometrically.

3ndy el image, w 3ndy el map bta3t el mkan, fa 3auz a2dr a3ml map benhom, kol mkan by-map lfen.

- We then define two Cartesian coordinate systems:
  - one describes the location of points in the map ***(x,y)***
  - the other defines the location of pixels in the image ***(u,v) or (x',y')***

3ndk sora, 3arf enha geometrically distorted, fa e7na 3auzen nsl7ha, fa bn3ml keda best5dam el mapping.

- Suppose that the two coordinate systems can be related via a pair of mapping functions:

$$u = f(x, y)$$
$$v = g(x, y)$$

- First, we define a grid over the map to act as the grid of pixel centers for the corrected image, referred to as the ***display grid***

e7na bnmshy 3la el displayed grid, nmshy 3la kol pixel, w nshuf el mfrod fe 3nd kol center, el mfrod yt7t feha anhy pixel.

- We then move over the display grid pixel center by pixel center and use the mapping functions above to find the pixel in the image corresponding to each display grid position.
- Those pixels are then placed on the display grid.
- At the conclusion of the process, we have a geometrically correct image built up on the display grid using the original image as a source of pixels.
- Problems:

1. We do not know the explicit form of the ***mapping functions***.
2. Even if we did, for a given display grid location they may not point exactly to a pixel in the image. In such a case some form of ***interpolation*** will be required.

Mapping Polynomials and the Use of Ground Control Points

- Since explicit forms for the mapping functions are not known
 →they are usually approximated by **polynomials** (first, second or third degree).
 Sometimes orders higher than three are used but care must be taken to avoid the introduction of errors worse than those to be corrected.

- In the case of second degree:

$$u = a_0 + a_1 x + a_2 y + a_3 xy + a_4 x^2 + a_5 y^2$$
$$v = b_0 + b_1 x + b_2 y + b_3 xy + b_4 x^2 + b_5 y^2$$

x/u (column) and *y/v* (row) coordinates

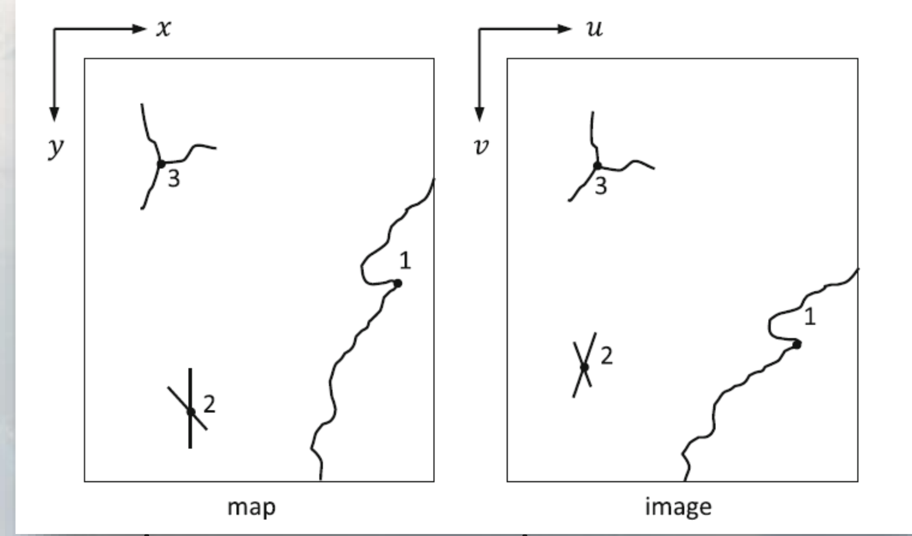
Known as: **Output-to-Input (Inverse) Mapping**
(u,v)→ original ***input*** image to be rectified
(x,y)→ ***output***-rectified image or map

- If the coefficients *ai* and *bi* were known then the mapping polynomials could be used to relate any point in the map to its corresponding point in the image.

- Coefficients values can be estimated by identifying sets of features on the map that can also be identified on the image. Those features, referred to as **ground control points (GCPs)** or just control points (CPs).

Mapping Polynomials and the Use of Ground Control Points

- These could be road intersections, street corners, airport runway intersections, sharp bends in rivers, coastline features, … etc.
- Enough are chosen, as pairs on the map and image so that the coefficients can be estimated by **substituting** the coordinates of the control points into the mapping polynomials to yield sets of equations in *ai* and *bi*.



- The minimum number of control points required:
    - for first order polynomial: 3
    - for second order polynomial: 6
    - for third order polynomial: 10
- In practice, significantly **more than those minimums** are chosen and the coefficients are evaluated using **least squares regression/estimation**.

→To avoid excessive influence on the estimated polynomial coefficients if any control points contain significant positional errors either on the map or in the image.

Problem 1 Solved

Ground Control Points

- Most image-to-map rectification still relies heavily on human selection of GCPs.
- A general rule is that there should be a distribution of control points around the **edges** of the image to be corrected, with a **scattering** of points over the body of the image → that is necessary to ensure that the mapping polynomials are well-behaved over the scene.
- After computing equations' coefficients, it is important to determine how well the coefficients account for the geometric distortion in the input image.

The method often used is root-mean-square error (**RMS_{error}**)

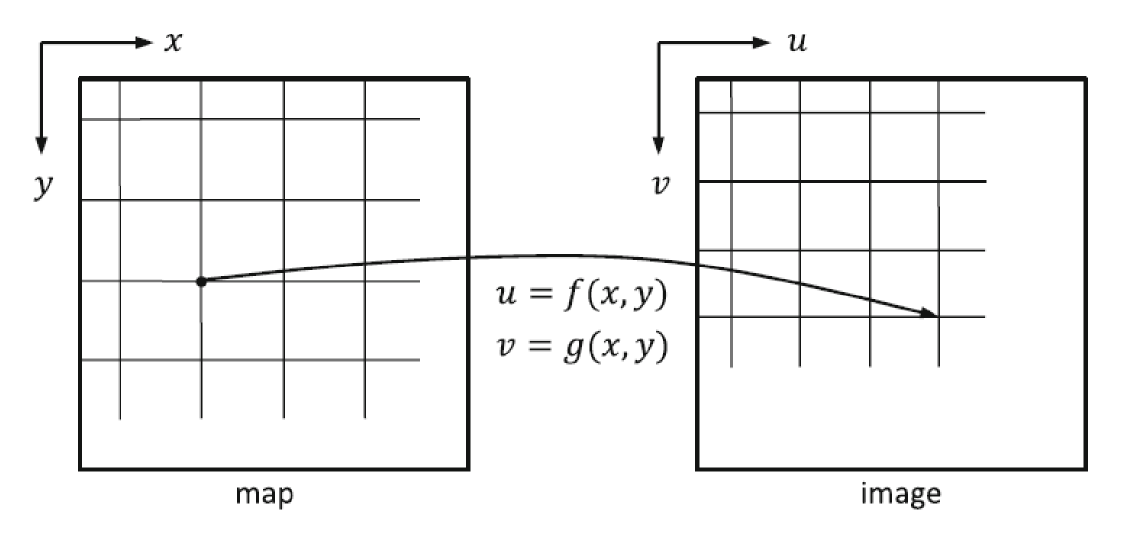
- Let x_{orig} and y_{orig} be the column and row coordinates of GCP in the original input image.
- Let x and y be the column and row coordinates of GCP in the reference map.
- Let x' and y' be the column and row coordinates of GCP from the equation.

→ Ideally $x' = x_{orig}$ and $y' = y_{orig}$

- For each GCP, compute: $RMS_{error} = \sqrt{(x' - x_{orig})^2 + (y' - y_{orig})^2}$
- By computing RMS_{error} for all GCPs, it is possible to:
  - 1) see which GCPs exhibit the greatest error 2) sum all the RMS_{error}
- The user specifies a threshold of acceptable total RMS_{error}. If the threshold is exceeded:
  - 1) delete the GCP that has the greatest RMS_{error} 2) recompute the coefficients 3) recompute the RMS_{error} for all points
  - This process continues until the total RMS_{error} is less than the threshold specified or too few points remain to perform a least-squares regression to compute the coefficients.

Building a Geometrically Correct Image

- Referred to as: **rectified image.**
- Having specified the mapping polynomials completely by the use of ground control points →the next step is to find points in the image that correspond to each location in the display grid.
- The spacing of that grid is chosen according to the **pixel size required** in the corrected image and need not be the same as that of the original, geometrically distorted version.
- For the moment suppose that the points located in the image correspond exactly to image pixel centers, even though that rarely happens in practice.
- Then those pixels are **simply transferred** to the appropriate locations on the display grid to build up the rectified image.



Using mapping functions to locate points in the image corresponding to particular display grid (map) positions

Resampling and the Need for Interpolation

- As expected, grid centers from the map-defined display grid will not usually project to exact pixel center locations in the image.
- Some decision now has to be made about what pixel brightness value should be chosen for placement on the new grid.
- Three principal techniques are used for this purpose:

Nearest Neighbor Resampling

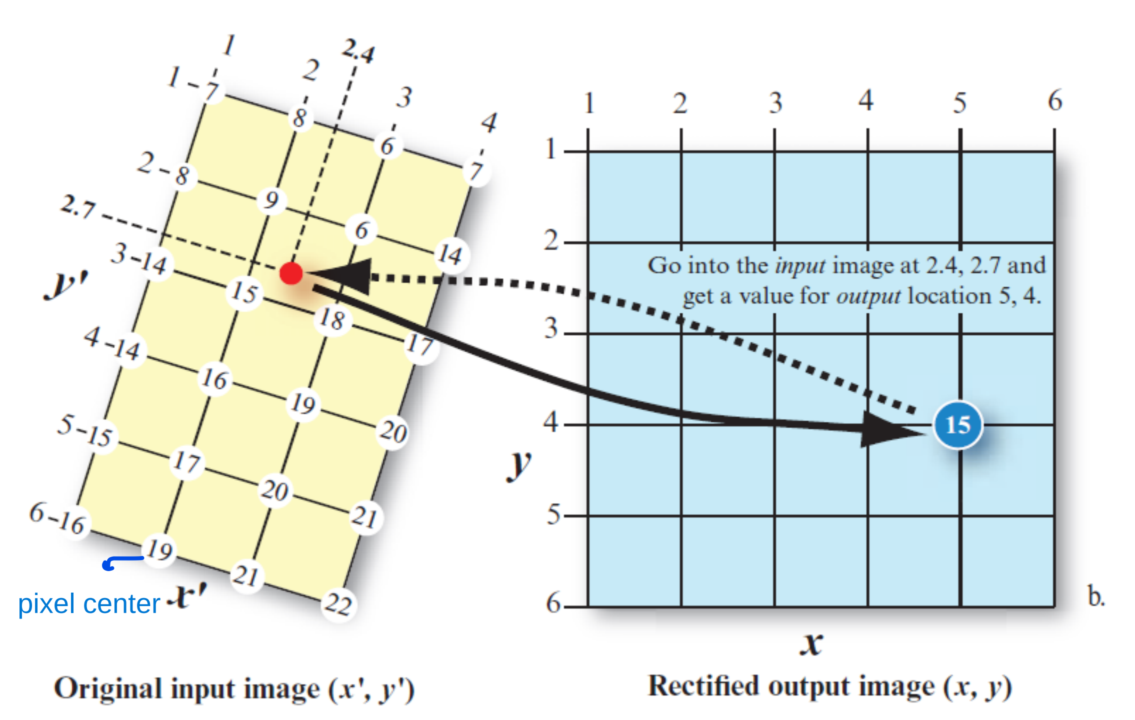
Bilinear Interpolation

Cubic Convolution Interpolation

Problem 2 Solved

Nearest Neighbor Resampling

- Simply chooses the actual pixel that has **its center nearest to the point located in the image**.
- That pixel value is transferred to the corresponding display grid location.
- This is the preferred technique if the new image is to be classified (e.g.: distinguishing one type of vegetation from another) because it consists only of original pixel brightness values, simply rearranged in position to give the correct image geometry.
- The method is only acceptable when the new and old pixel sizes and spacings are not too different.



$$x' = a_0 + a_1 x + a_2 y$$
$$y' = b_0 + b_1 x + b_2 y$$

Sample Point Location (column, row)	Value at Sample Point, Z	Distance from x', y' to the Sample Point, D
2, 2	9	$D = \sqrt{(2.4-2)^2 + (2.7-2)^2} = 0.806$
3, 2	6	$D = \sqrt{(2.4-3)^2 + (2.7-2)^2} = 0.921$
2, 3	15	$D = \sqrt{(2.4-2)^2 + (2.7-3)^2} = 0.500$
3, 3	18	$D = \sqrt{(2.4-3)^2 + (2.7-3)^2} = 0.670$

Bilinear Interpolation

- First-order or bilinear interpolation assigns output pixel values by interpolating brightness values in two orthogonal directions in the input image.
- It basically fits a plane to the **four pixel values** nearest to the desired position *(x′,y′)* in the input image and then computes a new brightness value based on the weighted distances to these points.
- The closer a pixel is to the desired location, the more weight it will have in the final computation of the average.
- The weighted average of the new brightness value ($\boldsymbol{BV_{wt}}$) is computed using the equation:

$$\text{Bilinear}_{BV_{wt}} = \frac{\displaystyle\sum_{k=1}^{4} \frac{Z_k}{D_k^2}}{\displaystyle\sum_{k=1}^{4} \frac{1}{D_k^2}},$$

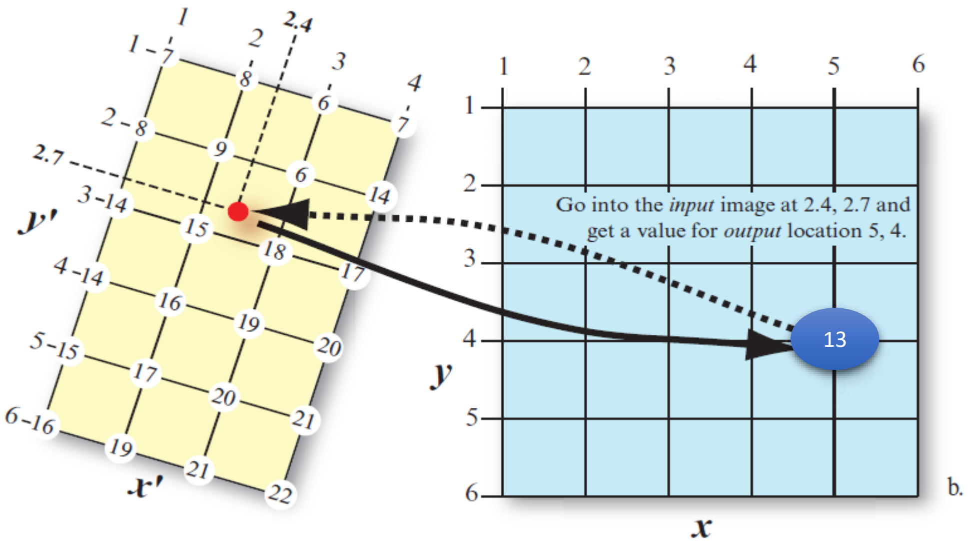
inverse distance

Where:

- Z_k are the surrounding 4 data point values.
- D_k^2 are the distances squared from the point in question (x′,y′) to these data points.

Bilinear Interpolation

- Example:



Original input image (x', y')

Rectified output image (x, y)

Sample Point Location (column, row)	Value at Sample Point, Z	Distance from x', y' to the Sample Point, D	D_k^2	$\frac{Z}{D_k^2}$	$\frac{1}{D_k^2}$
2, 2	9	$D = \sqrt{(2.4-2)^2 + (2.7-2)^2} = 0.806$	0.65	13.85	1.539
3, 2	6	$D = \sqrt{(2.4-3)^2 + (2.7-2)^2} = 0.921$	0.85	7.06	1.176
2, 3	15	$D = \sqrt{(2.4-2)^2 + (2.7-3)^2} = 0.500$	0.25	60.00	4.000
3, 3	18	$D = \sqrt{(2.4-3)^2 + (2.7-3)^2} = 0.670$	0.45	40.00	2.222
				Σ 120.91	Σ 8.937
				BV_{wt} = 120.91/8.937 =13.53	

Truncate to 13

Cubic Convolution Interpolation

- Assigns values to output pixels in the same manner as bilinear interpolation, except that **sixteen input pixels** surrounding the location of the desired pixel are used.
- The weighted average of the new brightness value (***BV**$_{wt}$*) is computed using the equation:

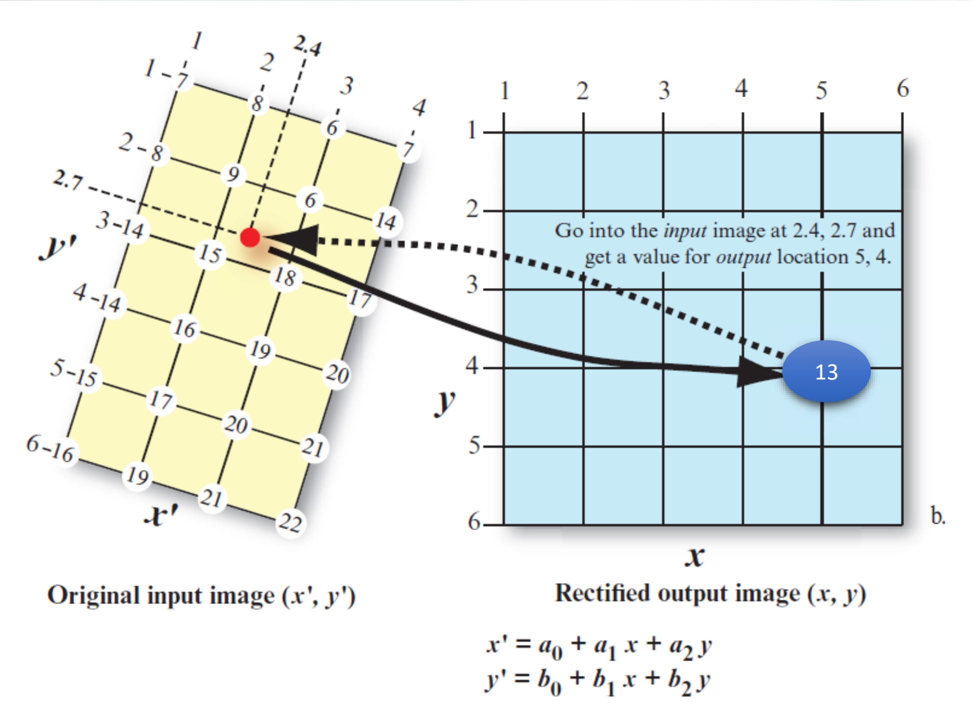
$$\text{Cubic Convolution}_{BV_{wt}} = \frac{\sum_{k=1}^{16} \frac{Z_k}{D_k^2}}{\sum_{k=1}^{16} \frac{1}{D_k^2}},$$

Where:

- Z_k are the surrounding 16 data point values.
- D_k^2 are the distances squared from the point in question (x',y') to these data points.

Cubic Convolution Interpolation

- Example:



Sample Point Location (column, row)	Value at Sample Point, Z	Distance from x', y' to the Sample Point, D	D_k^2	$\frac{Z}{D_k^2}$	$\frac{1}{D_k^2}$
1, 1	7	$D = \sqrt{(2.4-1)^2+(2.7-1)^2} = 2.202$	4.85	1.443	0.206
2, 1	8	$D = \sqrt{(2.4-2)^2+(2.7-1)^2} = 1.746$	3.05	2.623	0.328
3, 1	6	$D = \sqrt{(2.4-3)^2+(2.7-1)^2} = 1.80$	3.24	1.852	0.309
4, 1	7	$D = \sqrt{(2.4-4)^2+(2.7-1)^2} = 2.335$	5.45	1.284	0.183
1, 2	8	$D = \sqrt{(2.4-1)^2+(2.7-2)^2} = 1.565$	2.45	3.265	0.408
2, 2	9	$D = \sqrt{(2.4-2)^2+(2.7-2)^2} = 0.806$	0.65	13.85	1.539
3, 2	6	$D = \sqrt{(2.4-3)^2+(2.7-2)^2} = 0.921$	0.85	7.06	1.176
4, 2	14	$D = \sqrt{(2.4-4)^2+(2.7-2)^2} = 1.746$	3.05	4.59	0.328
1, 3	14	$D = \sqrt{(2.4-1)^2+(2.7-3)^2} = 1.432$	2.05	6.829	0.488
2, 3	15	$D = \sqrt{(2.4-2)^2+(2.7-3)^2} = 0.500$	0.25	60.00	4.000
3, 3	18	$D = \sqrt{(2.4-3)^2+(2.7-3)^2} = 0.670$	0.45	40.00	2.222
4, 3	17	$D = \sqrt{(2.4-4)^2+(2.7-3)^2} = 1.63$	2.65	6.415	0.377
1, 4	14	$D = \sqrt{(2.4-1)^2+(2.7-4)^2} = 1.911$	3.65	3.836	0.274
2, 4	16	$D = \sqrt{(2.4-2)^2+(2.7-4)^2} = 1.360$	1.85	8.649	0.541
3, 4	19	$D = \sqrt{(2.4-3)^2+(2.7-4)^2} = 1.432$	2.05	9.268	0.488
4, 4	20	$D = \sqrt{(2.4-4)^2+(2.7-4)^2} = 2.062$	4.25	4.706	0.235
				Σ 175.67	Σ 13.102

$BV_{wt} = 175.67 / 13.102$

$BV_{wt} = \mathbf{13.41}$

Truncate to 13

Image-to-Image Registration

- Many applications in remote sensing (e.g. change detection) require two or more scenes of the same geographical region, acquired at different times, to be processed together.

- Two images can be registered to each other by registering each to a map coordinate base separately in the manner previously demonstrated.

- Alternatively, and particularly if georeferencing is not important, one image can be chosen as a **master, or reference**, to which the other, known as the **slave**, is registered.

- Same techniques as image-to-map registration are used.
- However, the coordinates are now the pixel coordinates in the master image rather than the map coordinates.

- A benefit in image-to-image registration is that only one registration step is required, by comparison to two if both are taken back to a map base.

Other Image Geometry Operations

- Intentional changes to image geometry can be performed including:
  - Image Rotation
  - Scale Changing and Zooming

Image Rotation:

To rotate an image in an anticlockwise sense by any specified angle ζ:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos\zeta & \sin\zeta \\ -\sin\zeta & \cos\zeta \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

Negative the angle

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} \cos\zeta & -\sin\zeta \\ \sin\zeta & \cos\zeta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

Other Image Geometry Operations

Image Scale Changing and Zooming:

- The scales of an image in both the vertical and horizontal directions can be altered by the transformation:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix}$$

a and *b* are the desired scaling factors

- To resample the scaled image onto the display grid we use the inverse operation to locate pixel positions in the original image corresponding to each display grid position:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} 1/a & 0 \\ 0 & 1/b \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

- Interpolation is used to establish the actual pixel brightness values to use, since *u* and *v* will not normally fall on exact pixel locations.
- Frequently *a=b* so that the image is simply magnified called **zooming**.
- If the *nearest neighbor interpolation* procedure is used in the resampling process the zoom implemented is said to occur by ***pixel replication*** and the image will look progressively *blocky* for larger zoom factors.
- If *cubic convolution interpolation* is used there will be a *change in magnification* but the image will *not take on the blocky* appearance. Often this process is called ***interpolative zoom***.


Thank You