

---

Real-world implementation is essential in computer vision. Vision algorithms are often implemented in software using either MATLAB or Intel's Open Computer Vision (OpenCV) library for C/C++. The purpose of this section is to gain an initial familiarity with these two environments. You are asked to implement three simple vision algorithms below, to encourage breadth, at least one of them has to be implemented in OpenCV. Your code must be clearly structured and well-commented.

Gray-scale images each pixel has a single intensity on [0:255].

You should implement one of the problems (2 and 3) in OpenCV.

Post the code and resulting images when the exercise asks you to generate some new image.

- 1. Matlab Warmup** You are given a image [http://cs.gmu.edu/~kosecka/cs682/images/notes\\_color.jpg](http://cs.gmu.edu/~kosecka/cs682/images/notes_color.jpg). Do the following in Matlab, while avoiding to use loops:
  - Load the above image and resize the image by factor of 4 (use help resize);
  - Create a new gray level image (help rgb2gray);
  - Display histogram of gray level image intensities with 20 bins. (help hist or imhist);
  - Create binary image by setting all the pixels which have intensity greater the 125 to 255 and less then 125 to 0. (help find) Use the histogram of the gray level image to choose (by hand) better threshold to make the notes in the binary image readable;
  - Create a negative of the obtained gray level image;
  - Create color image of the same size as original resized binary image, but make the notes appear red. This can be achieve by setting the red channel to its maximal value, while making blue and green channel 0. The while color pixel has values [255, 255, 255].
- 2. Histogram Equalization** Often, we encounter an image whose dynamic range (ie: contrast) is compressed (see Figure 2). For example, in an 8-bit gray-scale image, only a narrow range of the 256 possible intensity values might be used. In that case, an image can contain a significant amount of detail that is not apparent visually. One approach for enhancing detail is called histogram equalization. It is straightforward to perform. First, we generate a histogram  $H$  of the intensities in the image. Specifically, we have one bin for each intensity 0-255. The value in each bin is the number of pixels in the image with that intensity. Second, we normalize the histogram such that the sum of the 256 values in bins 0-255 is 255. Third, we generate a second histogram  $H'$  where  $H'[i] = \sum_{0 \leq j \leq i} H[j]$  for all  $0 \leq i \leq 255$ . Finally,  $\text{destination\_image}[x, y] = H'[\text{source\_image}[x, y]]$ . Implement the transform on the image <http://cs.gmu.edu/~kosecka/cs682/images/landsat.jpg>. Submit your result as a PNG and the source code you used to generate it. An example of histogram equalization (on a different image) is at the end of this document. FAQ: It is helpful to keep floating-point precision until you write the new image in the last step. Otherwise, integer rounding will throw you off.
- 3. Contrast Stretching** Another approach to detail enhancement in the face of dynamic range compression is contrast stretching. Contrast stretching is even easier than histogram equalization:  $\text{destination\_image}[x, y] = (\text{source\_image}[x, y] - \text{image\_min}) * 255 / (\text{image\_max} - \text{image\_min})$  where image min and image max are the minimum and maximum intensity values present in the image. Implement the transform on the image <http://cs.gmu.edu/~kosecka/cs682/images/landsat.jpg>. Submit your result as a PNG and the source code you used to generate it.
- 4. Perspective Projection** It is often useful while testing some of the algorithms to simulate the perspective projection process. Write a MATLAB function, which implements the image formation process.

Write a function  $x = \text{project}(X, R, T, K)$  which takes as an input image coordinates of 3D points in the world coordinate frame and generates pixel coordinates of the projected points in the image, assuming that  $(R, T)$  is the displacement of the camera coordinate frame with respect to the world frame,  $K$  is the matrix of intrinsic image parameters, and  $X$  is a  $3 \times n$  vector of the coordinates of 3D points. To test the function consider a unit cube placed in the origin of the world coordinate system (specified by 8 vertices  $[0, 0, 0]^T, [1, 0, 0]^T, \text{etc}$ ), assume that the camera is translated along z-axis by some amount and rotated around x-axis by angle  $20^\circ$ . You can assume that matrix  $K$  is  $[800, 0, 250; 0, 800, 250; 0, 0, 1]$ . Generate the image of the cube. Its enough when you plot the vertices of the cube and optionally connect them by line to visualize it better. Submit the MATLAB code and generated MATLAB figure. It is commonly assumed that in the coordinate system of the camera the z-axis is pointing towards the scene and y down and x to the right. The function for generating rotation matrix (`rot_matrix.m`) from axis and angle parametrization and its inverse (`angaxis.m`) can be found in directory <http://cs.gmu.edu/~kosecka/cs682/code/>.

5. **Rigid Body Transformations** Consider rigid body transformations in the plane. Draw a right triangle defined by three points  $A = (2, 1), B = (4, 1), C = (4, 6)$ .

- Consider a rotation matrix

$$T_1 = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

- What is the determinant of the matrix ?
- Apply the rotation matrix to the triangle and show the result.

- Consider transformation matrix

$$T_2 = \begin{bmatrix} \sin \theta & \cos \theta \\ \cos \theta & -\sin \theta \end{bmatrix}$$

- What is the determinant of the matrix ?
- Apply the transformation matrix to the triangle and show the result. Is  $T_2$  rigid body transformation (i.e. can you move the triangle from initial position to the final without leaving the plane ? What is the difference between  $T_1$  and  $T_2$ , how are the results different?

6. **Rigid body transformations composition** Suppose that you are given the relative displacement between the coordinate frame  $\{1\}$  and  $\{2\}$ ,  $g_{21} = (R_{21}, T_{21})$  expressed in the frame  $\{1\}$  and relative displacement between the frame  $\{3\}$  and the frame  $\{2\}$  expressed in the frame  $\{3\}$ , and denoted by  $g_{23} = (R_{23}, T_{23})$ . What is the relative displacement  $g_{31}$ , between the frame  $\{1\}$  and frame  $\{3\}$  expressed in the frame  $\{1\}$  ?. a) Draw a figure; and write  $g_{31}$  in terms of given transformations/displacements. b) Write down explicitly what is the rotational and translational part of  $g_{31}$ , in terms of given rotations  $R_{ij}$  and  $T'_{ij}$ s.

7. **Vanishing Point** Straight line in 3D world is projected in to a straight line in the image. The projections of two parallel line intersect in the image at so called *vanishing point*.

- Show (mathematically) that projections of parallel lines in the image intersect in a point.
- Compute for a certain family of parallel lines where in the image will the vanishing point be.
- When does the vanishing point of the lines in the image lie at infinity (i.e. they do not intersect)?

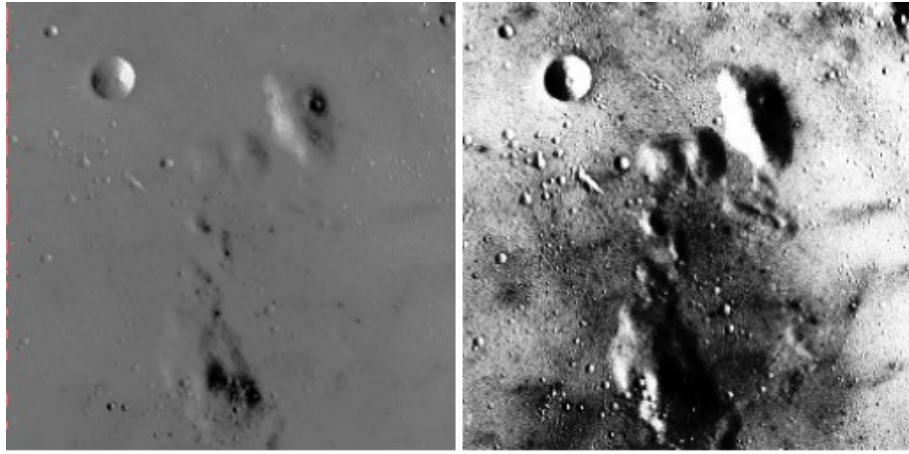


Figure 1: Histogram equalization: before and after