CS-512 – Assignment 1

Image formation and filtering

Due by: October 8, 2020

Review questions

Answer the following questions. Make sure that your answers are concise. In questions requiring explanation, make sure your explanation is brief.

1. Geometric image formation:

   (a) Let f=10 be the focal length of a camera. Let p=(3,2,1) be a world point. Find the coordinate of the point p when projecting it onto the image. Assume that the projection is done in camera coordinates so there is no need for a transformation between coordinate systems.

   (b) Explain the difference between the pinhole camera model where the image plane is behind the center of projection and the pinhole camera model where the image plane is in front of the center of projection. Which model corresponds better to a physical pinhole camera model? How is the other model justified?

   (c) Explain what happens to the projection of an object when the focal length gets bigger and what happens to the projection when the distance to the object gets bigger.

   (d) Given the 2D point (1,1) find its coordinates in homogeneous coordinates (2DH). Find another 2DH point that corresponds to the same 2D point.

   (e) Give the 2DH point (1,1,2), find the 2D point corresponding to it.

   (f) Explain the meaning of the 2DH point (1,1,0).

   (g) Explain what makes it possible to write the non-linear projection equation as a linear equation in homogeneous coordinates.

   (h) Given the projection matrix $M = K[I|0]$ write the dimensions of M and the sub-matrices K,I,0.

   (i) Given a projection matrix M whose rows are [1,2,3,4],[5,6,7,8],[1,2,1,2], and a 3D point P=[1,2,3], find the coordinate of the 2D point p obtained by projecting P using M.

2. Modeling transformations:

   (a) Given the point (1,1) find its coordinates after translating it by (2,3). Perform the computation using a transformation matrix.

   (b) Given the point (1,1) find its coordinates after scaling it by (2,2). Perform the computation using a transformation matrix.

   (c) Given the point (1,1) find its coordinates after rotating it by 45 degrees.

   (d) Given the point (1,1) find its coordinates after rotating it by 45 degrees about the point (2,2).

(e) Given that I want first to rotate an object using a matrix R and then translate it using a matrix T, what should be the combined matrix (expressed in terms of R and T) that needs to be applied to the object.

(f) Let M be a 2D transformation matrix in homogeneous coordinates whose rows are [3,0,0],[0,2,0],[0,0,1]. What is the effect of applying this matrix to transform a point p.

(g) Let M be a 2D transformation matrix in homogeneous coordinates whose rows are [1,0,1],[0,1,2],[0,0,1]. What is the effect of applying this matrix to transform a point p.

(h) Let M be a 2D transformation matrix in homogeneous coordinates whose rows are [3,0,0],[0,2,0],[0,0,1]. What is the transformation matrix will reverse the effects of this transformation?

(i) Let M=R(45)T(1,2) be a transformation matrix in homogeneous coordinates composed of rotation by 45 degrees and a translation by (1,2). Express the inverse of this transformation in terms a rotation and translation matrix.

(j) Find a vector which is perpendicular to the vector (1,3).

(k) find the projection of the vector (1,3) onto the direction defined by the vector (2,5).

3. General camera model:

(a) Explain the need for a general projection matrix that uses different coordinate systems for camera and image.

(b) Given that the camera is rotated by R and translated by T with respect to the world, write the transformation matrix that will convert world to camera coordinates.

(c) Given three unit vectors $\hat{x}, \hat{y}, \hat{z}$, write the rotation matrix describing the rotation of the camera with respect to the world.

(d) Given the transformation matrix between world and camera coordinates $M = \begin{bmatrix} R^* & T^* \\ 0 & 1 \end{bmatrix}$ explain the meaning of R^* and T^*.

(e) Given that there are k_u pixels per mm in the x direction, k_v pixels per mm in the y direction, and that the optical center of the camera is translated by $(u_0, v_0) = (512, 512)$ pixels, write the transformation matrix that will convert camera coordinates to image coordinates.

(f) Let the projection matrix M of a general camera be given by $K^*[R^*|T^*]$. Explain which parts contain the intrinsic and extrinsic parameters of the camera.

(g) Explain the reason for including a 2D skew parameter in the camera model.

(h) Explain what happens to the camera model when taking into account radial lens distortion. What is the complication introduced by the radial lens distortion?

(i) Explain the meaning of a weak-perspective camera and of an affine camera.

4. Color and photometric image formation:

(a) Explain the difference between surface radiance and image irradiance.

(b) Write the radiosity equations relating surface radiance and image irradiance.

(c) Define the albedo of a surface.

(d) Explain what is the reason for using the RGB color model to represent colors.

(e) Given the RGB color cube, what are the colors along the line that connects (0,0,0) with (1,1,1).

(f) Explain the way by which RGB colors are mapped to real-world colors.

(g) Given the CIE RGB color model and its conversion to the XYZ model, explain what is the use for the luminance component Y.

(h) Explain the advantage of the LAB color space.

5. Noise and filtering

   (a) Explain how to estimate the signal to noise ratio (SNR) in an image.

   (b) Explain the difference between Gaussian and impulsive noise. Which filter handles better impulsive noise: an averaging filter or a median filter.

   (c) Given an image having the value of 2 in each cell, write the value of the pixels in this image after applying a 3×3 convolution filter having all 1-s in its entries.

   (d) Given that we need the derivative of an image convolved with a filter explain how the operation can be applied more efficiently.

   (e) Explain the three different ways to handle boundaries during convolution.

   (f) Write a basic 3×3 smoothing filter. What is the sum of all entries in this filter? Explain the reason for the sum to be selected as it is.

   (g) Explain how to implement a 2D convolution with a Gaussian using two 1D convolution filters. Which option is more efficient? Is it possible to implement any 2D filter in this way?

   (h) Given a 1D Gaussian filter with $\sigma = 2$, what should be the size of this filter?

   (i) Explain how a Gaussian image pyramid is produced. What is the reason for producing such pyramids? What is the amount of additional processing done in a pyramid compared with a single image?

   (j) Explain how the Laplacian pyramid is produced and its use.

6. Edge detection

   (a) Why is edge detection useful? What are the desired properties of edge detection?

   (b) Explain the basic steps of edge detection and the need for them: smoothing, enhancement, localization.

   (c) Describe two filters for computing the image gradient. What is the meaning of the image gradient? What is it used for?

   (d) Explain how the Sobel filter can be produced from a smoothing and derivative filters.

   (e) Explain how to generate a more accurate derivative filter with an arbitrary σ. Write the elements of a filter for more accurate derivative computation with $\sigma = 2$.

   (f) Explain how an edge can be localized using the first or second order derivative of the image.

   (g) Let $\sigma = 1$. Write the Laplacian of Gaussian (LOG) filter using this σ. Explain how to use LOG to detect edges.

   (h) Explain the main difference between the Canny edge detection algorithm and a standard edge detection that does not use directional derivatives. What is the condition for detecting an edge candidate in Canny?

   (i) Explain the non-maximum suppression and hysteresis thresholding parts of the Canny algorithm.

Programming questions

In this part you need to write a program to perform simple image manipulation using openCV. The program should load an image by either reading it from a file or capturing it directly from a camera. When the user presses a key perform the operation corresponding to the key on the original image (not the result of the last processing step). The program should satisfy the following specifications:

1. The image to be processed by the program should be either read from a file or captured directly from a camera. If a file name is specified, the image should be read from it. Otherwise the program should attempt to capture an image from a camera. When capturing an image from the camera, continue to capture and process images continuously.

2. The read image should be read as a 3 channel color image.

3. The program should work for any size image. Make sure to test it on different size images.

4. When implementing your own convolution use Cython to speed up execution. If when you use Cython you do not get faster execution check what Cython is doing. Execute in a terminal: `cython myfile.pyx -a` then open the html file that is generated and read it.

5. Special keys on the keyboard should be used to modify the displayed image as follows:

   (a) 'i' - reload the original image (i.e. cancel any previous processing)

   (b) 'w' - save the current (possibly processed) image into the file 'out.jpg'

   (c) 'g' - convert the image to grayscale using the openCV conversion function.

   (d) 'G' - convert the image to grayscale using your implementation of conversion function.

   (e) 'c' - cycle through the color channels of the image showing a different channel every time the key is pressed.

   (f) 's' - convert the image to grayscale and smooth it using the openCV function. Use a track bar to control the amount of smoothing.

   (g) 'S' - convert the image to grayscale and smooth it using your function which should perform convolution with a suitable filter. In this question you must implement the convolution operation yourself instead of using a ready made one. Use a track bar to control the amount of smoothing.

   (h) 'd' - downsample the image by a factor of 2 without smoothing.

   (i) 'D' - downsample the image by a factor of 2 with smoothing.

   (j) 'x' - convert the image to grayscale and perform convolution with an x derivative filter. Normalize the obtained values to the range [0,255].

   (k) 'y' - convert the image to grayscale and perform convolution with a y derivative filter. Normalize the obtained values to the range [0,255].

   (l) 'm' - show the magnitude of the gradient normalized to the range [0,255]. The gradient is computed based on the x and y derivatives of the image.

   (m) 'p' - convert the image to grayscale and plot the gradient vectors of the image every N pixels and let the plotted gradient vectors have a length of K. Use a track bar to control N. Plot the vectors as short line segments of length K.

   (n) 'r' - convert the image to grayscale and rotate it using an angle of θ degrees. Use a track bar to control the rotation angle. The rotation of the image should be performed using an inverse map so there are no holes in it. Use the cv2.getRotationMatrix2D and cv2.warpAffine functions.

(o) 'h' - Display a short description of the program, its command line arguments, and the keys it supports.

6. In the report you prepare you must summarize the algorithms you used, and evaluate the performance obtained. See provided sample report.

Submission Instructions:

1. Create a folder AS1 in your bitbucket repository and create inside it the following sub-folders: `src`, `doc`, and `data`. Organize the submission materials inside the sub-folders as follows:

   (a) `doc`: Report prepared as a PDF file. The report should contain answers to questions, a summary of program design issues, description of specific problems you faced and the way in which you solved them, and sample input/output results (text/graphic). The report needs to be sufficiently detailed. It is very important that you evaluate the algorithms you implemented for correctness and for performance. Try using different parameters and observe (and report) the effect of the parameters.

   (b) `src`: All program files.

   (c) `data`: All data files (e.g. test images).

2. Note that we must be able to view your report and execute your program in order to grade it.

3. For programming questions use OpenCV and Python (support code will be provided).

4. On or before the due date upload your submission to your bitbucket repository. If you are late, upload the submission when you are ready. To compute "late days will" we will use the last update date of your repository. Do not make any changes to the folder after submitting it so that it does not cause a change to your submission date.

5. Do not submit a paper copy of your report. You will be contacted by email if some material is missing or if you will need to meet with the TA.