Assignment 2
Resampling and Interpolation

CS473/CS673: Medical Image Processing

Due 4:00pm Thursday February 4, 2015

Goal: to get first-hand experience with interpolation and resampling of images and volumes.

Questions

1. [5 marks] Implement an interpolation function capable of nearest-neighbour and linear interpolation in both 2D and 3D. The function should have the calling prototype,

   ```
   y = MyInterp(f, x, method)
   ```

   where `f` is the image/volume to be interpolated, `x` is a vector representing the location to be interpolated (expressed in Matlab index coordinates), and `method` is either `'nearest'` or `'linear'`. Your function should automatically detect whether to do 2D or 3D interpolation. If `f` is a 2D array or has only 1 slice, then `x` should be a 2-vector and your function should perform 2D interpolation. If `f` has more than 1 slice, then `x` should be a 3-vector, and your function should perform 3D interpolation. The output value, `y`, is a scalar interpolation value. If `x` falls outside of `f`, the return value should be 0. You may **not** use Matlab's `interp` functions for this; you must implement the interpolation operation yourself.

2. [4 marks] Implement a function that applies an affine spatial transformation to a 2D or 3D dataset. Your method should have the calling prototype

   ```
   g = MyAffine(f, A, method)
   ```

   where `f` is the image/volume to be transformed, `A` is a 4×4 homogeneous-coordinate transformation matrix, `method` is either `'nearest'` or `'linear'`, and `g` is the spatially-transformed (resampled) version of `f`. Note that `g` should be the same size as `f`. Your function should call your `MyInterp` function from question 1. You may **not** use Matlab's `tformarray` function; you must implement the transformation yourself.

3. [4 marks] Write a Matlab function that blurs an image (2D or 3D) by convolving it with a Gaussian kernel. The function should have the calling prototype

   ```
   g = MyGaussianBlur(f, sigma)
   ```

   where `sigma` is the standard deviation that defines the width of the Gaussian kernel. Your function should automatically detect whether `f` is 2D or 3D using the same method as question 1, and deal with it appropriately. Implement the convolution in the frequency domain. The output `g` should be the same size as `f`. The 2D Gaussian function is given by the formula

   $$g(x,y) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^2} \exp\left(\frac{-\left(x^2+y^2\right)}{2\sigma^2}\right) .$$

   The corresponding 3D version is

   $$g(x,y,z) = \frac{1}{\left(\sqrt{2\pi}\sigma\right)^3} \exp\left(\frac{-\left(x^2+y^2+z^2\right)}{2\sigma^2}\right) .$$

   The supplied function `Gaussian` computes this for you. See its help file ("`help Gaussian`") for usage instructions.

Graduate Question

If you are registered in CS 673, then you must also do this problem.

4. [4 marks] **Non-affine deformation:** Write a Matlab function that performs deformations similar to the ones shown below. In particular, the pixels near the centre of the image are either pinched closer to the centre, or drawn out as if looking through a fish-eye lens. Your function must be called `MyLens` and have the calling signature

   ```
   g = MyLens(f, p, method)
   ```

   See the supplied starter code for more specifics. Your function should output an image the same size as the input image. The parameter `p` is a scalar, and `method` is either `'nearest'` or `'linear'`, as used in your `MyInterp` function. This function only needs to work in 2D (not 3D). You must call your `MyInterp` function to perform any resampling.
   Hint: You might find it useful to use Gaussian functions.

Original image | Deformed with $p = -1$ | Deformed with $p = 0.5$

What do I hand in?

1. `MyInterp.m`
2. `MyAffine.m`
3. `MyGaussianBlur.m`
4. `MyLens.m` (if you are registered for CS 673)

Zip up your code and submit the zip file through the "Drop Box" on Desire2Learn. Your Matlab code will be tested by running it. However, we will also be looking at the listings of your scripts and functions. It is essential that you document your code. A well-designed and properly implemented script or function will NOT earn full marks without comments in the code.