More hints for Poisson matting

1. In the searching step, you are recommended to use built-in function: `KDTreeSearcher()`, `knnsearch()`. These two functions use KD-tree structure to accelerate nearest neighbor search.
   Refer to MATLAB document for more help. You can follow these steps:
   1) Use `meshgrid()` to get x-value and y-value of each pixel. Combine them together, so you have n-by-2 matrix. Where n is number of foreground (or background) pixels, and 2 means its x and y value.
   2) Use `KDTreeSearcher()` to build a KD-tree of that n-by-2 matrix in 1).
   3) Follow 1), get a m-by-2 matrix for all undecided pixels.
   4) Call function `knnsearch()` which take the KD-tree in 2) and the matrix you want to query in 3) as input.
   5) Fifth, according to the resulting index in 4), get the F and B value for undecided pixels.

2. In Poisson equation step. I have updated `poisson_equ()` in skeleton code.(**CSCI3290_Assignment3_2.rar**)
   Now it is much more easier for your to call.
   Notice: you still need to look into the code for possible modification. For example, 'eps' is a variable to avoiding dividing 0 problems. Change that value will change the results.

3. For quick debugging, you may first resize the input images and trimaps to accelerate running time.

4. For different images you may need to tune different parameters.

5. I share one possible parameter setting based on my implementation:

Image:	2.bmp – dragonfly
Size:	resize to 384*256, half of original size.
Gaussian filter:	`fspecial('gaussian',[5,5],0.5);`
'eps' in `poisson_equ()`:	1e-3

   The following is my results after 6 iterations. And F and B for undecided regions.