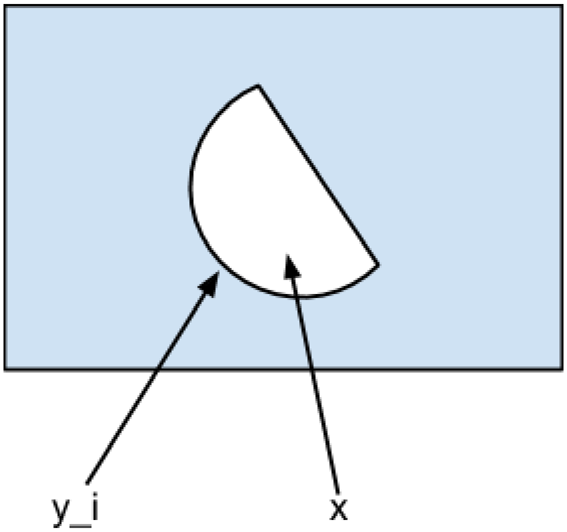
Hole Filling

The goal is to create an algorithm for image hole filling. The algorithm must be implemented in one of the following languages: C, C++, Objective-C or Java.

You're given a grayscale image (2D array), where each pixel value is a float in the range [0, 1], and invalid (missing) values which are marked with the value -1.

You can assume there's only a single hole in the image:



1. Find the boundary of the hole. The boundary pixels can be defined as the 4-connected or 8-connected pixels (input). See http://en.wikipedia.org/wiki/Pixel_connectivity for the definitions.
2. Given an image I, the boundary pixels locations y_i and a missing pixel location x, set the value of the pixel $I(x)$ as follows:

   $$I(x) = \frac{\sum_i w(y_i,x)\cdot I(y_i)}{\sum_i w(y_i,x)}$$

   In your design, you should support any arbitrary weighting function. The default function you'll use is $w_z(y_i,x) = \frac{1}{|x-y_i|^z+\varepsilon}$, where ε is a small float value used to avoid division by zero. The values $z,\ \varepsilon$ should be should be configurable.
3. Show the input image, the image with its boundary marked and the filled image (you can use OpenCV, PIL or any other library that supports image I/O).
4. If there are m boundary pixels and n pixels inside the hole, what's the complexity of the algorithm? (Try to also express the complexity only in terms of n).
5. Write an algorithm that approximates the result from (2) in $O(n)$ to a high degree of

accuracy.

Bonus (hard!):

- Write an algorithm that finds the *exact* solution in $O(nlogn)$.

All parameters given in this exercise should be easily configured via the command line. You cannot use any existing functions for the basic tasks, besides reading the image’s pixel data.