Answers

1. There are m boundary pixels and n pixels inside the hole. The complexity of the algorithm that fills the hole assuming that the hole and boundary already found is $O(n^{1.5})$.
   The number of points in the hole is n, which is the surface of a 2 dimensional shape, and its boundary is of order $\sqrt{n}$ (for example, if the hole is a square then the number $\sqrt{n}$ is exact).
   Each weight calculation is O(1), this is done for $\sqrt{n}$ pixels at the boundary, and we do this for all n pixels of the hole. This algorithm is implemented in the function “HoleFiller.Run”.
2. I will describe algorithm that approximates the result in $O(n)$ to a high degree of accuracy: we choose randomly, say, 30 pixels from the boundary and calculate the weight of each pixel with respect to this subset of boundary pixels and instead of using all the boundary. This algorithm is implemented in the function “HoleFiller. HoleFiller.RunFast”.
3. I will describe algorithm that finds the exact solution in $O(nlogn)$: first we assume that every weight function is invariant to translations, i.e. W(u,v) = W(u-v). this assumption is valid because most weight functions will be a function of the Euclidian distance |u-v|. next we note that the numerator of the expression for the filled hole is a convolution between an image where all pixels are 0 except the boundary pixels, and a kernel matrix which is the weight function evaluated at every point with respect to the center of the matrix. Similarly, the denominator is a convolution between an image where all boundary pixels are set to 1 and all others are 0, with the same kernel matrix. In the implementation we crop a square containing the hole (approximately n pixels), perform the two convolutions, divide the resulting matrices, and plug the result back into the original image. Building the kernel is $O(n)$, convolution between two vectors of size n is $O(n \log n)$, so the total complexity of this implementation is $O(n \log n)$. This algorithm is implemented in the function “HoleFiller.RunConv”.