1. Assume m is the boundary (B) size, and n, the hole (H) size. The algorithm complexity is $O(m * n)$:
   - Let M be the original image matrix
   - Let A be the weighting matrix of HxB ($\in R^{m \times n}$) , e.g. $A_{u,v} = w(u, v)$
   - Let v_B be a vector representation of the boundary values, e.g. , $u = (i, j),\ v_{Bu} = m_{i,j}$
   - We can calculate the new values for the hole pixels, $u \in H,\ I(u) = \frac{\sum_{v \in B} w(u, v) \cdot I(v)}{\sum_{v \in B} w(u, v)}$ by simple matrix operation:

     $v_H = A_{u,v} \cdot v_B \oslash A_{u,v} \cdot 1_m$, which is $O(m * n)$

   The boundary size is upper bounded by $O(n)$ - the hole is connected (4/8-way), so the worst case is the case when the intersection between each connected component of each hole pixel is minimized (an example for such case is when each hole pixel is located diagonally in respect to the previous pixel), but in any case, since the size of each connected component is upper bounded by 8 (or even 4 in case of 4-way connection), the complexity of the algorithm is $O(n^2)$.

2. In case we are using the default weighting function (or any other weighting function that depends on the distance between the boundary and the pixel), we can estimate $I(u)$ based on the connected component values only. We will start by estimating the value of the hole pixels with a distance of 1 between them and the boundary, and continue on, in the same manner to the rest of the hole pixeles, by the depth order. The complexity of this algorithm is $O(n)$, because we visit each hole pixel only once, and the estimation per pixel is done on $O(1)$.