answers

1. Assuming the hole and the boundary are already found, there will be required $n \cdot m$ computitions (for each hole pixel, calculating its color according to m boundary pixels). The number m is bounded above by $4n + 4$ (WLOG say 8-connectivity), when hole is a digonal. In such case $m = 4n + 4$, then:

$$n \cdot m \leq n \cdot (4n + 4) = 4n^2 + 4 \in O\left(n^2\right)$$

2. Algorithm:
   - Diving neighbor boundary pixels into a fixed number of sets (say k) (where in each set all pixels are neighbors.
   - Assigning to a set S values v and (x, y), where v is the mean coulor of all the pixels in S and (x, y) is the location of the middle pixel in the set.
   Now apply the algorithm on the "compressed" bounday.
   Time analysis:
   - Dividing m pixels into k sets and set values: $m \in O\left(n\right)$
   - Filling the hole: $k \cdot n \in O\left(n\right)$

$$\Downarrow$$

$$m + kn \in O\left(n\right)$$

   A possible implementation is holding the boundary points in an ordered data structure (such as ArrayList) and add the boundary pixel neighbor by neighbor. Then easily divide the arraylist into k sublists.