Lightricks Take-Home Project

Hadar Sharvit

April 30, 2022

See GitHub Here for code and README

1 Question I

let $|H| = n$ and $|B| = m$. Looking at the algorithms definition, for every $u \in H$ we perform two summations, each one over $|B| = m$ elements. In other words, for every $u \in H$ we perform $O(1) \times 2m = O(m)$ operations, where the $O(1)$ indicates the complexity of the multiplication $w(u, v)I(v)$ for some u and some v (in the numerator), or the "multiplication" of $w(u, v)$ with 1 (in the denominator). This results in an $O(nm)$ complexity operation overall.
The above result could also be understood in terms of vector and matrices multiplication, as was implemented in the code. More specifically, our weight function $w : H \times B \to \mathbb{R}^{>0}$ was represented as a matrix W with dimensions $n \times m$, and the sum over all $v \in B$ could be represented as a dot product of W with the vector that represents all points in B (in the numerator) or the dot product with W and a vector of ones with size $|B| = m$ (for the denominator).
Furthermore, as the boundary is generated using at most 8 neighbours of a pixel in the hole, it could contain at most $8n = O(n)$ points, therefore $O(nm) = O(n^2)$. (Do notice though that only if $n = 1$ we have 8 neighbours, and in any other case that considers only one hole, the number of neighbours is usually smaller)

2 Question II

A plausible solution to this would be as followed: divide our Hole H into ℓ disjoint sections $\{H_i\}_{i=1}^{\ell}$ (notice that ℓ is a constant, predefined number which is not a function of n). Next, calculate the center of every H_i (Center of mass, if you will) $\{\mu_i\}_{i=1}^{\ell}$. From here, in an iterative process, set $I(H_j)$ using the same algorithm as before, using μ_j instead of u. In other words, for every $u_j \in H_j$ we set $I(u_j) = \frac{\sum_{v \in B} w(\mu_j, v)I(v)}{\sum_{v \in B} w(center, v)}$. Notice that in this scenario, the weight function could be represented as a different vector for every ℓ (or a matrix $\ell \times m$, rather than a matrix of size $O(n^2)$)
In terms of time complexity, generating $\{H_i\}_{i=1}^{\ell}$ takes $O(n)$ as we go over all points in H and split. From here we loop ℓ times, and in each iteration we calculate the center of mass (which is an $O(n)$ operation), the corresponding weight vector that represents distances between the current center of mass to all points in B (which is an $O(n)$ operation as well), and finally the summation which adds up $|B| = O(n)$ elements as well (this was represented as vector multiplication), All together we are dealing with $\ell \times (O(n) + O(n) + O(n)) = O(n)$.

3 Question III

Couldn't really figure it out, but perhaps there is a way to represent $\sum_{v \in B} w(u, v)I(v)$ (or $W \cdot I_B$) as a convolution, and if that so, one could use the convolution theorem to achieve $nlogn$ runtime. More specifically, using $FT^{-1}[FT[f] \cdot FT[g]]$ instead of $f * g$, if f and g are function such that $f * g$ represents the sum of $w \cdot I$