Pokud bychom chtěli implementovat mediánový filtr tak, že bychom přímočaře pro každý pixel zkopírovali veškeré hodnoty ze čtverce filtru do pole a spustili QuickSelect, dostali bychom časovou složitost $\Theta(N^2K^2)$. Avšak takhle budeme trávit zbytečně moc času opakovaným prohledáváním týchž pixelů.

To však lze vylepšit použitím vyhledávacího stromu umožňující duplicitní klíče (v C++ by to byl `std::multiset`). Při jeho používání si budeme udržovat ukazatel na prvek na pozici $\left\lceil \frac{n}{2} \right\rceil - 1$ (pro strom o lichém počtu prvků medián, jinak levý prvek ze dvou "prostředních"), kde n je aktuální počet prvků tohoto stromu, což zvládneme při každé operaci v konstantním čase. Díky tomuto ukazateli budeme schopni získat medián prvků v tomto stromě v konstantním čase.

Pak pro každý řádek obrázku vytvoříme dočasně nový vyhledávací strom, pro první pixel řádku přidáme všechny pixely ve čtverci filtru, zjistíme medián a pak pro každý další pixel v řádku přidáváme nebo odebíráme prvky po sloupcích tak, abychom dostali strom s pixely v novém čtverci filtru. Každé přidávání a odebírání pixelu ze stromu trvá $\mathcal{O}(\log K^2) = \mathcal{O}(2 \log K) = \mathcal{O}(\log K)$ a protože s každým pixelem pracujeme nejvýše dvakrát, průchod filtru řádkou má složitost $\mathcal{O}(NK \log K)$. Pro celý obrázek je tedy časová složitost tohoto algoritmu $\mathcal{O}(N^2K \log K)$ a prostorová složitost je $\mathcal{O}(N^2 + K^2) = \mathcal{O}(N^2)$.