- I've tried to implement the SOR, but stuff doesn't go well, and I can only submit the CU file that can properly run. If you'd like to review the code, they are blocked by a return statement in the iteration kernel.

  The basic idea is simple,

  $$C' = \omega C + (1 - \omega) C_0$$

  where C_0 is the background pixel, while ω is limited by the iteration linearly

  $$\omega = 0.99 + 0.02(1 - \frac{\text{iteration}}{\text{iteration}_{MAX}})$$

  by linearly, means that ω decreases by the iteration counts, and it will swing pass 1, which is the original formula, and continue to the constraint regime. However, it seems that it can still over compensate. I believe ω can be fine tuned by the RMS value, but I have run out of time.

- I've used Thrust to implement a RMS error calculator. The result shows that 20k cycles stops at RMS differences $5 \times 10{-}5$, hence I've limited the cycle by dual constraint: maximum cycles or the RMS differences fall below a certain value – which can minutely limit the iterations required.