





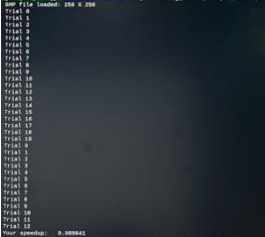
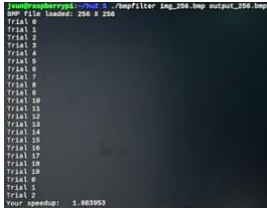




2024 system programming hw2 report

202211153 이지은

First section: Implementation results

	Code before modification	Code after modification
img_128.bmp	0.941860	1.867849
img_256.bmp	0.988641	1..883953
img_512.bmp	0.989310	1.896843
img_768.bmp	1.055106	2.035159
img_1024.bmp	0.983012	2.251391

	Code before modification	Code after modification
img_128.bmp		
img_256.bmp		
img_512.bmp		
img_768.bmp		
img_1024.bmp		

Second section: Optimization approach

Optimization Strategy

1. Use of inline functions

By using inline functions, we reduced the overhead of function calls. By defining the convolution function and filter_optimized function as inline functions, we prevented performance degradation due to function calls and improved the functions' performance.

2. Loop Optimization

We removed all loops by directly constructing all calculations within the convolution function in code. For calculations within the filter_optimized function, we moved all code that could be calculated outside the loop and that was executed frequently to outside the loop, thereby improving performance.

3. Elimination of Unnecessary Memory Allocations

We modified the approach to avoid performance degradation due to repeated dynamic memory allocation and deallocation. Unlike the pre-optimization approach, which dynamically allocated and deallocated the Pixel structure each time, the optimized approach stores values directly in the output array, saving time spent on memory management.

Discuss with evaluated results

	Code before modification	Code after modification	Percentage of code performance improvement
img_128.bmp	0.941860	1.867849	198.3 %
img_256.bmp	0.988641	1..883953	190.6 %
img_512.bmp	0.989310	1.896843	191.7 %
img_768.bmp	1.055106	2.035159	192.9 %
img_1024.bmp	0.983012	2.251391	229.0 %

Average percentage of code performance improvement: 200.46 %

I have optimized the code using the aforementioned optimization strategies, resulting in approximately a 2x improvement in performance.