

1. Results

```
./bmpfilter img_128.bmp output_128.bmp
```

```
BMP file loaded: 128 X 128
Trial 0
Trial 1
Trial 2
Trial 3
Trial 4
Trial 5
Trial 0
Trial 1
Trial 2
Trial 3
Your speedup: 1.628914
```

```
./bmpfilter img_256.bmp output_256.bmp
```

```
BMP file loaded: 256 X 256 Trial 0
Trial 0 Trial 1
Trial 1 Trial 2
Trial 2 Trial 3
Trial 3 Trial 4
Trial 4 Trial 5
Trial 5 Trial 6
Trial 6 Trial 7
Trial 7 Trial 8
Trial 8 Trial 9
Trial 9 Trial 10
Trial 10 Trial 11
Trial 11 Trial 12
Trial 12 Trial 13
Trial 13 Trial 14
Trial 14 Trial 15
Trial 15 Trial 16
Trial 16 Trial 17
Trial 17 Trial 18
Trial 18 Trial 19
Trial 19 Your speedup: 1.464533
```

```
./bmpfilter img_512.bmp output_512.bmp
```

```
BMP file loaded: 512 X 512 Trial 0
Trial 0 Trial 1
Trial 1 Trial 2
Trial 2 Trial 3
Trial 3 Trial 4
Trial 4 Trial 5
Trial 5 Trial 6
Trial 6 Trial 7
Trial 7 Trial 8
Trial 8 Trial 9
Trial 9 Trial 10
Trial 10 Trial 11
Trial 11 Trial 12
Trial 12 Trial 13
Trial 13 Trial 14
Trial 14 Trial 15
Trial 15 Trial 16
Trial 16 Trial 17
Trial 17 Trial 18
Trial 18 Trial 19
Trial 19 Your speedup: 1.523413
```

```
./bmpfilter img_768.bmp output_768.bmp
```

```
BMP file loaded: 768 X 768
Trial 0
Trial 1
Trial 2
Trial 3
Trial 4
Trial 5
Trial 6
Trial 7
Trial 8
Trial 9
Trial 10
Trial 11
Trial 12
Trial 13
Trial 14
Trial 15
Trial 16
Trial 17
Trial 0
Trial 1
Trial 2
Trial 3
Trial 4
Trial 5
Trial 6
Trial 7
Trial 8
Trial 9
Trial 10
Trial 11
Trial 12
Trial 13
Trial 14
Trial 15
Trial 16
Trial 17
Trial 18
Trial 19
Your speedup: 1.504968
```

```
./bmpfilter img_1024.bmp output_1024.bmp
```

```
BMP file loaded: 1024 X 1024
Trial 0
Trial 1
Trial 2
Trial 3
Trial 4
Trial 5
Trial 6
Trial 7
Trial 8
Trial 9
Trial 10
Trial 11
Trial 12
Trial 13
Trial 14
Trial 15
Trial 16
Trial 17
Trial 18
Trial 19
Trial 0
Trial 1
Trial 2
Trial 3
Trial 4
Trial 5
Trial 6
Trial 7
Trial 8
Trial 9
Trial 10
Trial 11
Trial 12
Trial 13
Trial 14
Trial 15
Trial 16
Trial 17
Trial 18
Trial 19
Your speedup: 1.561794
```

2. Strategy

- **Memory Allocation:** Removed dynamic memory allocation (malloc and free) inside the loop. Instead, we directly store the result of the convolution function in the output array.
- **Parameter Passing:** The parameters for the convolution function are passed as const pointers where applicable, which allows the compiler to optimize the code better.
- **Loop Variables:** Used unsigned int for loop variables to match the type of width and height, ensuring consistency and potentially helping the compiler optimize loop conditions.
- **Memory Access Pattern:** Iterating over the image using a nested loop with y and x ensures that memory access is more cache-friendly. This improves spatial locality and helps in better utilization of the CPU cache.
- **Boundary Checking:** Simplified boundary checking to ensure that we only process valid pixels, avoiding unnecessary computations.
- **Pixel Value Clamping:** Ensured pixel values are clamped between 0 and 255 within the convolution function itself.

This implementation should be more efficient and perform better in terms of execution time compared to the original baseline implementation.