Underwater Image Enhancement Using FPGA-Based Gaussian Filters with Approximation Techniques

Summarized on: October 26, 2023

1. Introduction / Abstract

Core Problem: Images taken underwater are often hazy, blurry, and have distorted colors. This is caused by how light scatters and gets absorbed in water, which makes it difficult to use these images for important tasks like studying marine life or inspecting underwater equipment.

Proposed Solution & Main Finding: This paper presents a new method to clean up this "Gaussian noise" (a type of random static) in underwater images. The researchers implemented a **Gaussian filter** (a common image-smoothing technique) on a special, reconfigurable computer chip called an **FPGA** (**Field-Programmable Gate Array**; a type of integrated circuit that can be programmed by a user after it's manufactured, making it highly flexible for specific tasks). To make the process faster and more energy-efficient, they used two key strategies: a "pipeline" structure that works like an assembly line for data, and "approximate computing," where they intentionally used simplified circuits that make tiny, acceptable errors in exchange for huge gains in speed and power savings. Their method resulted in a speed increase of over 150% and a power reduction of over 34%.

Tradeoffs & Limitations: This performance boost comes at a cost. The new design requires more physical space on the FPGA chip, and the use of approximation causes a slight, controlled decrease in the final image quality. Therefore, this approach is best suited for "error-resilient" applications, like real-time video processing, where top speed and low power are more critical than perfect, pixel-for-pixel accuracy.

2. Methodology

The researchers designed and tested their image enhancement system through the following key steps:

1. **Hardware Setup:** They designed their system for an **FPGA**. To process images efficiently without using a lot of memory, they used a "delay line buffer" structure. This system only stores the small number of pixel rows needed to process the current part of the image, rather than loading the entire image at once.
2. **Applying the Gaussian Filter:** The core of the method is a **Gaussian filter**, which smooths the image by performing a **convolution** (a mathematical operation where a small grid of numbers, called a kernel, is

slid across the image to calculate a new value for each pixel based on its neighbors). This process effectively blurs out the random noise.

3. **Pipelining for Speed:** To accelerate the **convolution** calculation, they designed a four-stage "pipeline." Much like an assembly line, this breaks the calculation into smaller steps. As one set of pixels finishes stage 1, it moves to stage 2, allowing the next set of pixels to enter stage 1. This parallel processing dramatically increases the overall speed.
4. **Approximation for Power Efficiency:** To reduce power consumption, the researchers replaced the standard, precise circuits for addition ("adders") with ten different types of "approximate adders." These simplified circuits are faster and use less energy but can introduce small errors. They strategically applied this approximation only to the least significant bits of the pixel data, minimizing the impact on the final image quality.
5. **Testing and Evaluation:** They tested their design by taking two real underwater images, adding artificial Gaussian noise, and then processing them with their filter. They measured the output image quality using standard metrics (like PSNR and SSIM) and evaluated the hardware performance (power consumption, speed, and area used on the FPGA) using simulation tools.

3. Theory / Mathematics

The research is based on the mathematical principles of the Gaussian function and convolution.

The primary formula is the two-dimensional Gaussian function, which creates the "weights" for the filter:

$$f(x,y) = \frac{1}{2\pi\sigma^2} e^{\frac{-(x^2+y^2)}{2\sigma^2}}$$

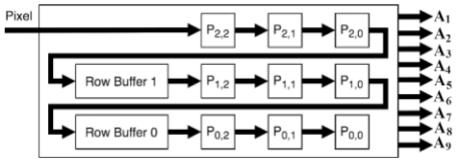
- **What it means:** This equation describes a 2D bell curve. The value `f(x,y)` is highest at the center (0,0) and decreases as you move away. In image filtering, this function is used to create a "kernel" or mask where the center pixel gets the most weight, and surrounding pixels get progressively less weight.
- **Why it's used:** This creates a smooth, natural-looking blur that is very effective at reducing random noise without creating harsh artifacts. The (sigma) value controls the "width" of the bell curve, determining how much the image is blurred.

The filter is applied to the image using a **convolution** operation, which for a 3x3 kernel is calculated as:

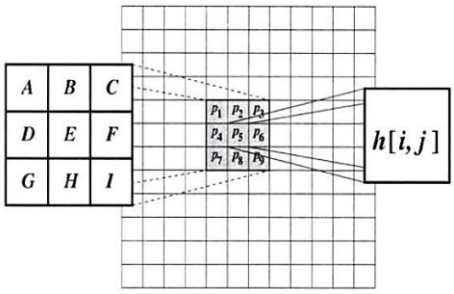
$$h[i,j] = AP_1 + BP_2 + CP_3 + DP_4 + EP_5 + FP_6 + GP_7 + HP_8 + IP_9$$

- **What it means:** The new value for a pixel (`h[i,j]`) is calculated by taking the values of its nine neighbors in a 3x3 grid (`P1` through `P9`) and multiplying each one by a corresponding weight from the Gaussian kernel (`A` through `I`). All these products are then added together.
- **Why it's used:** This is the fundamental operation for applying the filter. It systematically combines the information from a pixel's local neighborhood to compute its new, smoothed value.

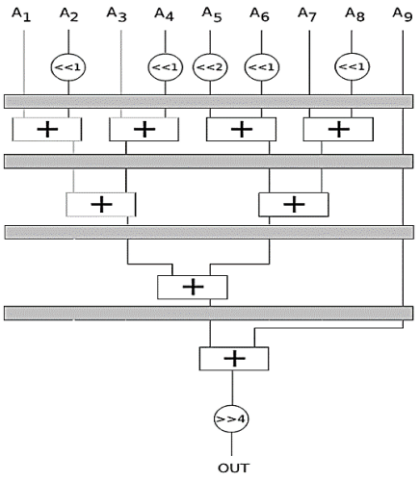
4. Key Diagrams or Visual Elements



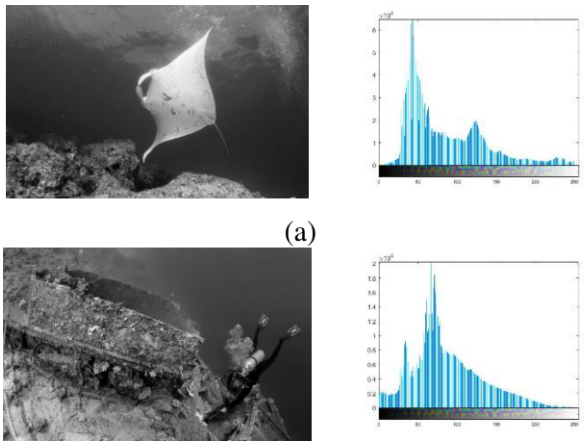
- **Figure 1: Delay line buffer structure:** This diagram shows the memory architecture used on the FPGA. It illustrates how two "row buffers" and a 3x3 "window buffer" are used to efficiently feed pixels to the filter, minimizing the need to access larger, slower memory.



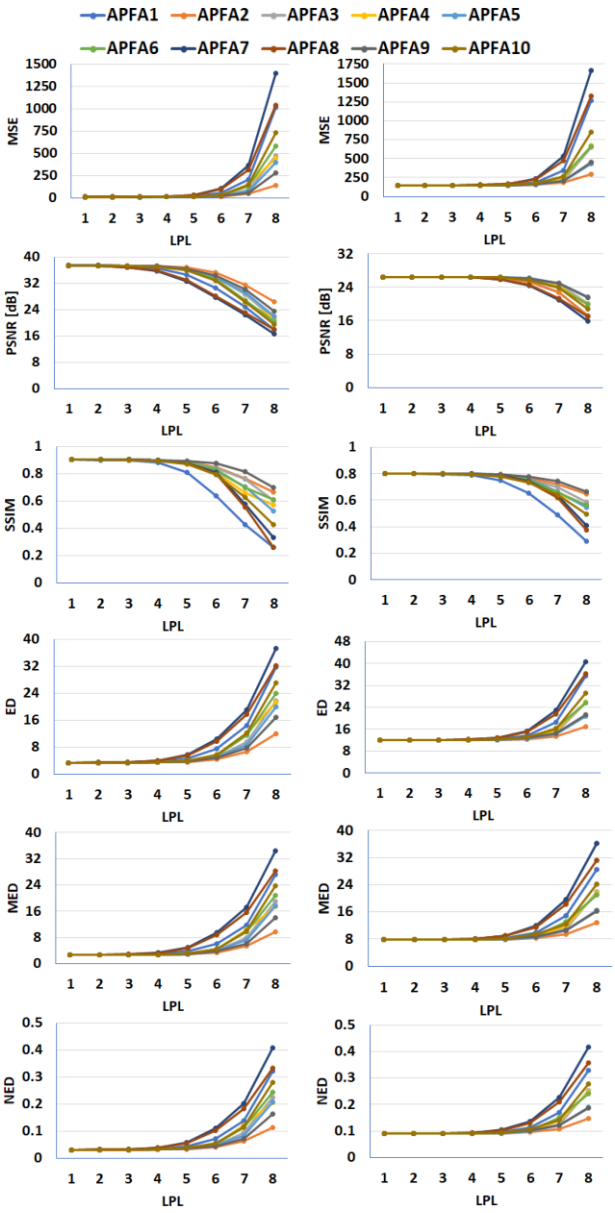
- **Figure 2: Convolution Operation on an image with a 3x3 kernel:** This visual perfectly illustrates the concept of convolution. It shows the 3x3 grid of weights (the kernel) being overlaid on a section of the image's pixel grid. This helps visualize how a new pixel's value is calculated from its neighbors.



- **Figure 5: Pipeline Gaussian filter:** This block diagram is crucial as it shows the paper's main innovation for speed. It depicts the filter's calculation being broken down into four distinct stages, demonstrating how data flows through them sequentially like an assembly line to enable parallel processing.



- **Figure 7: Two raw underwater images:** These images of a "flatfish" and a "diver" are the real-world test cases. They show the kind of hazy, low-contrast images the algorithm is designed to improve.



- **Figure 8: Evaluation Metrics for Image Enhancement:** These graphs show the results of the experiment. They plot image quality metrics (like PSNR) against the level of approximation used. The key finding shown here is that image quality remains high for up to 5-6 bits of approximation but then drops off sharply, defining the useful limit of the technique.

- **Table 2 & 3: Comparison on physical properties:** These tables present the hardware performance results. Table 2 shows that the

pipelined filter is significantly faster and more power-efficient than the standard one. Table 3 provides a detailed breakdown of the power, speed, and area for each of the ten approximate adders tested, highlighting which ones offer the best trade-offs.

5. Conclusion

The researchers successfully designed and demonstrated a novel architecture for enhancing underwater images on an FPGA. By combining a pipelined structure with various approximate adders, they created a **Gaussian filter** that is over 150% faster and uses over 34% less power than a standard implementation. While this comes at the cost of increased chip area and a slight, controlled reduction in image quality, the performance gains are substantial. The study confirms that there is a critical balance between the level of approximation and the resulting image quality, with the best results achieved when approximating 5 to 6 of the least significant bits.

Why It Matters: This research provides a practical blueprint for building high-speed, low-power vision systems for use in challenging environments. This technology could be directly applied to autonomous underwater vehicles (AUVs), enabling them to process video and "see" more clearly in real-time without quickly draining their batteries, thus improving their ability to navigate, explore, and conduct scientific research in the ocean's depths.