INTERNATIONAL JOURNAL OF
COASTAL, OFFSHORE & ENVIRONMENTAL ENGINEERING

Available online at: www. Ijcoe.org

Hardware-Accelerated Image Moment Computation for UUV Navigation

Mehrnaz Monajati[1]

[1] *Assistant professor, Department of Electrical and Computer Engineering, Graduate University of Advanced Technology, Kerman, Iran; m.monajati@kgut.ac.ir*

ARTICLE INFO

Article History:
Received: 01 May. 2023
Accepted: 05 Sep. 2023
Available online: 05 Jan. 2024

Keywords:
Unmanned Underwater Vehicles (UUVs)
Image moment
Pipeline
Systolic array
Real time

ABSTRACT

Image moments are pivotal in the control of Unmanned Underwater Vehicles (UUVs), empowering them to extract valuable insights from onboard camera images, identify objects, navigate autonomously, and adapt to evolving environmental conditions. Integrating image moment analysis into UUV control systems amplifies their effectiveness in exploration, research, surveillance, and various underwater applications. However, the computational demands of moment calculation algorithms pose challenges for real-time implementation, particularly with higher moment orders. In this study, we propose a novel structure based on systolic arrays, leveraging pipeline technique, to compute moments up to the 14th order of grayscale images in real time. Implemented in 45nm CMOS technology, our design demonstrates impressive performance, with each cell capable of computing the 14th order moment of a 1024×1024 image at a remarkable rate of 954 fps. Moreover, our design boasts low power consumption, registering at only 3.254 mW, demonstrating its potential for enhancing UUV control systems for diverse underwater applications.

1. Introduction

Unmanned Underwater Vehicles (UUVs) are autonomous robotic vessels designed for underwater operations without direct human control [1]. Equipped with sensors and propulsion systems, they navigate underwater environments, collect data, and perform tasks across various fields including oceanographic research, underwater mapping, marine biology, defense, infrastructure inspection, and search and rescue operations. UUVs offer a versatile platform for exploration, research, security, and industrial applications, contributing to our understanding and utilization of the world's oceans [2].

Image moments are indispensable tools in the realm of UUVs, playing a pivotal role in various aspects of their operation [3]. These moments are integral to extracting meaningful information from onboard camera images, enabling UUVs to recognize objects, navigate autonomously, and adapt to changing environmental conditions. The incorporation of image moment analysis into UUV control systems significantly enhances their capabilities for exploration, research, surveillance, and various underwater applications [3].

In feature extraction, image moments serve as essential mathematical descriptors derived from pixel intensities, providing valuable insights into an image's shape, size, orientation, and object distribution [4]. This feature extraction process is particularly crucial for UUV navigation, where visibility is often limited, making it imperative to extract pertinent features from onboard camera imagery for effective navigation and obstacle avoidance.

Furthermore, image moments facilitate object recognition and tracking, empowering UUVs to identify and monitor objects of interest in their surroundings [5]. This capability is essential for tasks such as detecting underwater structures, monitoring marine life, and identifying potential hazards. By accurately recognizing and tracking objects, UUVs can autonomously navigate through complex underwater environments while adeptly avoiding obstacles. Additionally, image moments contribute significantly to localization and mapping efforts. Through the analysis of moments extracted from successive images, UUVs can estimate their position and orientation relative to known landmarks or predefined maps. This information is crucial for ensuring precise navigation and achieving mission objectives, especially in environments where GPS signals may be unreliable or inaccessible [6].

Real-time implementation of image moments is crucial for UUVs because it enables swift decision-making, autonomous navigation, and obstacle avoidance in underwater environments [6]. By analyzing visual data

in real-time, UUVs can recognize objects, track their movements, and adjust their course accordingly, ensuring efficient and safe operation. Additionally, real-time processing enhances mission efficiency by enabling prompt responses to changing conditions and optimizing task execution. This capability not only enhances the autonomy and reliability of UUVs but also contributes to their overall safety and effectiveness in diverse underwater applications [7].

An approach for swiftly computing Krawtchouk moments in grayscale images, alongside rapid calculation techniques for binary images is presented in [8]. By decomposing 8-bit/pixel grayscale input images into corresponding bitplanes represented by image blocks, lower-order bitplanes exhibit a resemblance to a half-intensity image. Consequently, only the moments of higher-order bitplanes are computed, with the least significant bitplanes replaced by half-intensity images to significantly accelerate moment computation while maintaining an acceptable error margin between original and reconstructed images. Experimental findings confirm the effectiveness of substituting the lower 5 bitplanes with half-intensity images, resulting in minimal error in reconstructed images and notable acceleration in moment computation. Additionally, it demonstrates rapid performance, achieving real-time operation under typical conditions in pattern recognition scenarios requiring a moderate number of moment calculations in small-sized images.

In [9], researchers introduce an algorithm for computing discrete image moments based on first-order moments, achieved through a simple mathematical deduction, enabling the utilization of a fast algorithm for their calculation. Unlike conventional methods relying on moment kernel polynomials' properties, this approach allows for the computation of any discrete image moments, offering advantages such as a straightforward computation structure, no need for multiplication operations, independence from image intensity distribution, and applicability to all discrete image moment families. Additionally, they present a discrete image moment computation structure based on a systolic array, leveraging its characteristics for implementation using very large-scale integration (VLSI) technology.

In [10], the Hu Moment concept serves as a detection algorithm implemented on an FPGA system through hardware description language. This algorithm effectively recognizes target shapes within test images. The implementation involves utilizing two finite state machines to compute Hu Moments and the NLM filter. The findings demonstrate that the hardware implementation significantly outperforms software in the computation of image moments, showcasing the efficiency of FPGA-based processing for such tasks.

In [11], the hardware implementation of a Shape Recognition Algorithm based on Invariant Moments is outlined, aiming for optimized execution speed and resource utilization. Manual optimization, combined with pragma commands, results in the most efficient version in the third implementation, fully utilizing available resources. The study underscores the importance of specific optimizations for improved performance, as High-Level Synthesis (HLS) and programmable logic alone do not guarantee efficiency. Results indicate significant time efficiency gains using HLS tools and certain optimization methods compared to software implementation, with manual optimization being particularly effective. However, a trade-off between performance and chip utilization in FPGA technology persists, as demonstrated by compromises between processing speed and the ability to handle multiple matrices simultaneously across different implementations.

In this article, we introduce two novel structures designed for real-time computation of two-dimensional moments up to the 14th order (q=0,1,2,...,7, p=0,1,2,...,7) for grayscale images. These structures utilize a systolic parallel array employing pipelining techniques and a compressor. Computation is performed in floating-point format and implemented using 45nm CMOS technology.

2. Image Moments

Image moments are mathematical descriptors used in image processing and computer vision to characterize various properties of an image, such as its shape, size, orientation, and intensity distribution. These moments provide valuable information for tasks such as object recognition, tracking, and analysis. The formula for computing image moments depends on the specific moment being calculated. However, the general formula for computing moments of an image function is given by Eq. 1:

$$M_{pq} = \sum_{x=1}^{m} \sum_{y=1}^{n} x^p y^q f(x,y) \qquad x \le m, y \le n \qquad (1)$$

where M_{pq} represents the (p,q)-th moment, and the integration is performed over the entire region R. This equation captures the essence of moment computation, wherein pixel intensities f(x,y) are multiplied by the spatial coordinates x^p and y^q, and then integrated over the region of interest.

Image moments find widespread utility across diverse domains, serving as essential mathematical descriptors used in image processing and computer vision [12]. They characterize various properties of an image, including its shape, size, orientation, and intensity distribution. These moments play a crucial role in numerous applications, such as object recognition, shape analysis, image registration, segmentation, and texture analysis. They facilitate tasks like object recognition and classification by extracting discriminative features capturing an object's centroid, orientation, and size. Additionally, moments aid in shape analysis, enabling comparisons between shapes and facilitating tasks like template matching and shape similarity measurement. In image registration,

moments are instrumental in aligning and matching images from different sources, while in segmentation, they contribute to partitioning images into meaningful regions based on their moment properties [13]. Furthermore, image moments play a crucial role in texture analysis, capturing statistical properties of image textures for tasks such as texture classification and discrimination [14]. Overall, image moments serve as foundational elements in image analysis and computer vision, empowering a broad spectrum of applications across various fields.

Low-order moments are utilized for extracting features and determining the center of gravity, position, and orientation of an object within an image [15]. On the other hand, higher-order moments capture additional intricate features of the image and are commonly employed in pattern recognition and image reproduction tasks. However, directly computing the moments according to Eq. 1 necessitates $m \times n \times (p+q+2)$ addition operations, where m and n denote the dimensions of the image. This poses significant challenges when dealing with a large number of calculation operations, including $m \times n$ multiplication operations, particularly in scenarios requiring real-time moment computation.

In [16], orthogonal moments are computed using recurrence relations for Legendre and discrete Chebyshev polynomials, aiming to achieve both speed and accuracy in computation. The study emphasizes minimizing errors in moments approximation to maintain their discriminative capability in classification tasks. Numerical experiments across six texture image databases reveal that moments computed via recursion formulas outperform those derived from closed-form representations. Additionally, moments extracted from the Gray-Level Co-occurrence Matrix (GLCM) demonstrate high classification accuracy, especially with a higher number of moments. The study also highlights the potential of weighted moments to enhance classification results. [17] presents a parallel algorithm effectively implemented to compute 2D and 3D Legendre moments for gray-level images and objects. This algorithm is assessed on multicore CPUs and GPUs, with experimental results closely aligning with theoretical-to-optimal speedup ratios for multicore CPUs and demonstrating significant acceleration for GPUs. Furthermore, [10] presents a hardware-based shape detection algorithm utilizing Hu's moments and a non-local means filter on an FPGA system. The paper outlines the algorithm's design stages, emphasizing speed and parallelism in signal processing, and showcases its effectiveness in recognizing target shapes within test images.

3. Systolic Array

The Systolic Array serves as a versatile solution for translating high-level computations into hardware structures. Its fundamental design involves processing data from memory in a sequential manner, with each cell performing computations before passing the data to the next cell in the array. Noteworthy advantages of the systolic system include its modularity, regular and streamlined data flow, rapid response time, and the ability to utilize uniform and simplistic cells [4].

According to Eq (1), the calculation of image moments involves repeated multiplication operations for each pixel of the image, followed by the summation of these calculated expressions. In real-time processing, it's essential to initiate the computation for each pixel as it emerges from the camera and continue until the arrival of the next pixel. Once the entire image has been captured, the processing concludes. If the camera output is in Raster Scan format, the expression $x^p y^q f(x, y)$ needs to be computed for individual pixels in the first row of the image. Subsequently, as each subsequent row is captured, the operation is repeated for that row. Given this process, employing a systolic array is well-suited for implementing the moment calculation algorithm, with numerous existing structures relying on this architecture [18],[19].

In this study, we utilize the systolic structure depicted in Figure 1. This structure comprises n+1 pipeline stages. The symbol "+" denotes the unit of calculation for the adder unit, while "P" signifies the unit for exponentiation and multiplication. The final adder combines the output of the preceding adder with the previously stored value. Each combination of "+" and "P" forms a moment processor element (MPE). Figure 2 illustrates the structure of MPE cell. The control unit receives data f(x, y) from the camera and transmits f(x, y) along with y and x values relevant to each cell. For brevity, the schematic of the control unit is omitted. In this configuration, all units operate simultaneously on the rising edge of the clock pulse. The x^p computing unit, also known as an exponentiation unit (EXP), is responsible for raising a given input value x to the power of P. This unit is essential in various computational tasks where exponentiation operations are required, such as in polynomial evaluations, signal processing, and numerical simulations. The unit typically consists of arithmetic and logic circuits designed to efficiently compute exponentiation operations with high accuracy and low latency.

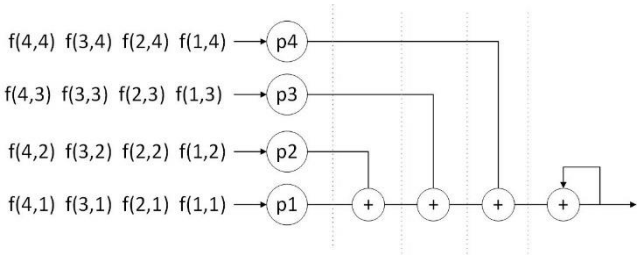

Figure 1. The systolic array structure for implementing the image moment calculation algorithm utilizes four processors.

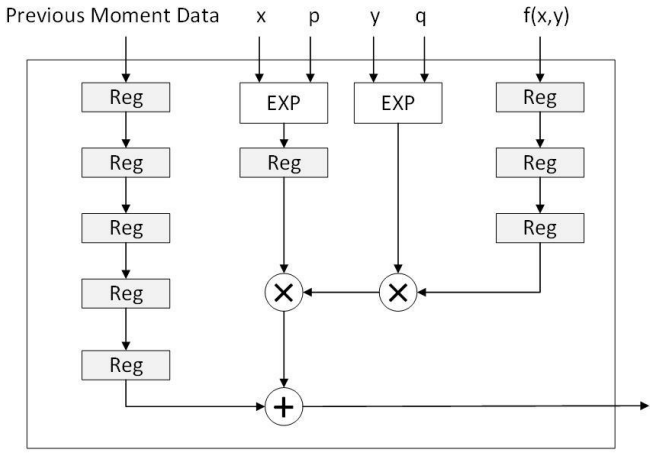

Figure 2. Moment processor element (MPE).

When multiple MPEs are connected sequentially, they form a systolic array of cells, each responsible for processing specific portions of the image data. In Figure 3, we observe a structural arrangement composed of four cells. In this setup, MPE1 initially processes pixels associated with the kth column of the image. Following this, MPE2, MPE3, and MPE4 consecutively handle calculations for the subsequent columns. Upon completing computations for these four columns, the result from MPE4 is fed back to MPE1, initiating a continuous processing cycle. Subsequently, pixels from the next set of four columns are directed to MPE1 through MPE4, thus repeating this sequence iteratively for all columns in the image. This feedback loop ensures seamless and efficient processing of the entire image dataset through the systolic array structure.

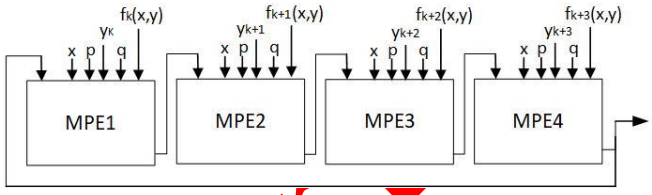

Figure 3. Structure of a Systolic Array Consisting of Four Cells for Image Moment Computation.

3.1 Computatonal units

Choosing the number format is crucial in designing the moment calculation structure, as it dictates the bit width of computing units and transmission lines. A 164-bit number is necessary to represent the maximum value of the 14th-order moment in binary format. However, employing binary numbers for moment calculation up to the 14th order would require 164-bit arithmetic units, increasing complexity, power consumption, and gate count. To mitigate this, floating-point number formats are utilized. In this research, the maximum moment order calculated is 5.8688×10^{48}. Using 8 bits for the exponent and 10 bits for the mantissa ensures sufficient range and precision. The decimal equivalent of the maximum number represented by this number of bits is equal to 1.16×10^{77}, which exceeds the maximum value of M77.

Since the calculation of moments involves only positive numbers, the sign bit is omitted. Furthermore, to enhance accuracy, the normalized mode of floating-point numbers is utilized. The maximum relative rounding error in calculating moments up to the 14th order using this format is 3.27%, which is acceptable for practical applications.

3.1.1 Adder

In this design, a Carry Select Adder (CSA) is constructed using a combination of multiplexers (muxes) and full adders (FAs). The CSA architecture facilitates efficient addition of two multi-bit binary numbers by distributing the carry bits across multiple stages, reducing the critical path delay and enhancing performance [20]. The design commences with the computation of the least significant bit (LSB) sum (Sum[0]) and carry-out (C-out) using a single full adder. Subsequently, the next bit of the sum (Sum[1]) is computed alongside the carry-in (C-in) from the previous stage, once again employing a single full adder. To compute the next two bits of the sum (Sum[3:2]), a pair of serial full adders are utilized in conjunction with a 4:2 multiplexer, allowing for efficient carry propagation and selection. Similarly, the subsequent three bits of the sum (Sum[6:4]) are computed using a combination of 6:3 multiplexers and three serial full adders.

This architecture enables parallel computation of multiple bits of the sum while efficiently managing carry propagation. By distributing the addition process across multiple stages and utilizing serial adders where necessary, the design achieves reduced critical path delay and optimal performance. Additionally, to efficiently add two 18-bit floating-point numbers, the process involves comparing the numbers and adjusting the smaller number's mantissa based on the difference in their exponents. This addition operation requires a 10-bit adder, such as the ten-bit Carry Select Adder (CSA) utilized in this study. Ensuring optimal performance entails balancing the input paths to the CSA to maintain uniform delay across all layers. The design of a high-speed and low-power ten-bit CSA adder, crucial for achieving fast and energy-efficient computation, is illustrated in Figure 4. The CSA offers several advantages, including high-speed addition, low power consumption, compact hardware utilization, scalability to larger bit widths, and minimized critical path delay, making it a preferred choice for arithmetic units in various computing systems.

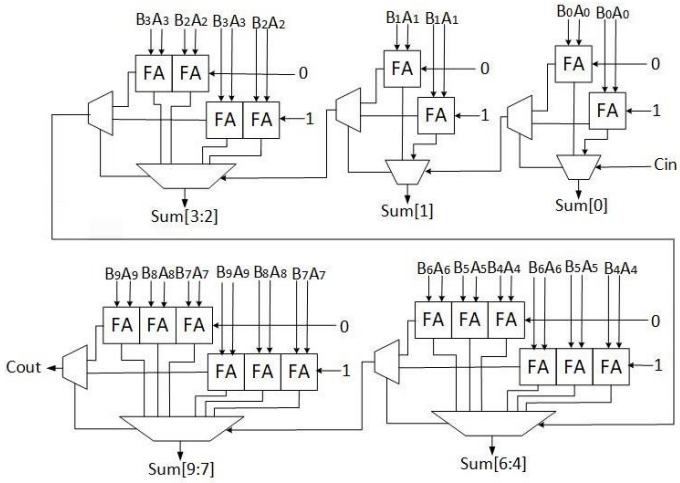

Figure 4. Ten bit CSA.

3.1.2 Multiplier

In this study, the 10*10 Dadda multiplier is employed as the multiplier unit, acknowledged for its superior speed compared to similar types. Leveraging a modified Booth encoding technique, the Dadda multiplier facilitates efficient and rapid multiplication operations by reducing the number of partial products, thereby enhancing computational performance. Operating in stages, each stage of the Dadda multiplier comprises a specific number of rows of partial products. The height of each stage is determined by working backward from the final stage, typically consisting of two rows of partial products. The design incorporates a strategy to reduce the number of partial products, utilizing two layers of full adders. Each full adder input carries a Cin, representing the Cout of the previous full adder. By utilizing several full adders, a logical circuit is created to add multiple-bit numbers, with each carry bit "rippling" to the next full adder. This architecture necessitates employing the Ripple Carry Adder (RCA) procedure. Data is transmitted to adders, with the carry of each stage added to the next two data inputs in the same stage. Ultimately, the ripple carry adder process is executed at the final stage, yielding product terms p1 to p8, as illustrated in Figure 5.

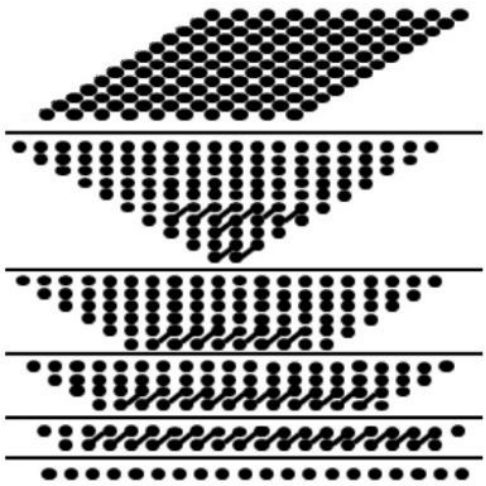

Figure 5. Dot diagram illustrating the 10×10 Dadda multiplication process.

3.1.3 Exponentiation Unit

To compute the powers of x^p and y^q, p and q clock cycles are required, respectively. However, as the order of moments increases, the processing time becomes extended. Figure 6 illustrates a more efficient structure for this operation. The proposed design comprises three pipeline stages, allowing the computation of the power of a number within three clock cycles. Notably, this time remains constant regardless of the value of p. This scheme supports calculations up to x^7. Based on the power value, the multiplexer selects and outputs the desired value.

The use of floating-point units for processing numbers leads to higher power consumption compared to binary number formats. Complex adder and multiplier units are required for binary formats, such as the 164-bit CSA which demands multiple layers of adders. In contrast, the hardware complexity of comparators and 10-bit shifters is significantly lower. Designing Dadda multipliers presents even greater complexity than CSA architectures.

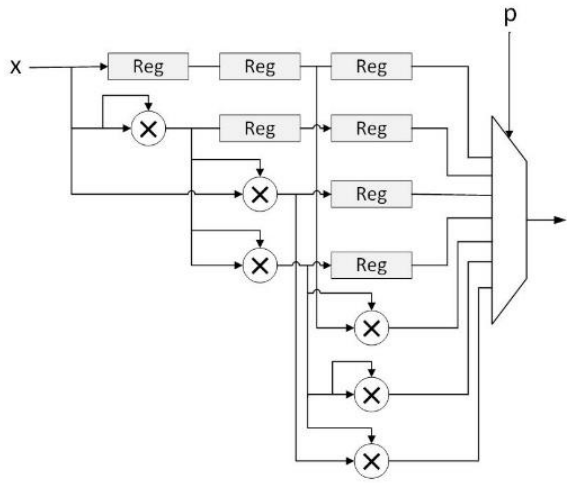

Figure 6. Pipeine exponentiation unit computing x^p.

4. Simulation Results

The physical attributes of the moment computation structure were analyzed by synthesizing them with the Nangate 45nm open cell library using the Synopsys Design Compiler. Power consumption was estimated using the Synopsys power analysis tool, based on a VCD file generated from post-synthesis simulations with 100,000 random inputs. All comparisons were conducted post-synthesis for all designs at the block level, following the practices of ASIC designers who thoroughly assess building blocks for potential integration into future chips.

Table 1 compares different architectures, including the proposed approach, based on speed, power consumption, and area. The proposed approach exhibits significantly higher speeds in gigabits per

second (618.243 Gbit/s) and frames per second (954.012 frames/s) compared to existing architectures, marking an improvement of approximately 571% and 1024%, respectively. Additionally, it demonstrates substantially lower power consumption (3.254 mW), showcasing a reduction of around 98.7% compared to other methods. However, it occupies a slightly larger area (0.734 mm²) than some of the other structures, representing a minor increase.

Table 1. Comparative analysis of performance metrics for various structures.

Architecture	[21]	[22]	[16]	Proposed
Max Moment Order	16	10	18	14
Transistor count	43894	143253	639804	543251
Speed (Gbit/s)	86.481	32.215	92.285	618.243
Speed (frame/s)	45.277	25.568	84.237	954.012
Power (mw)	254.863	185.657	19.586	3.254
Area (mm^2)	0.218	0.521	0.657	0.734

The combined utilization of systolic array, parallelization process, pipeline, and the implementation of fast multiplier and adder units in the proposed method has significantly enhanced its efficiency compared to other methods. By optimizing the performance of each block within the structure, such as employing the "Dada Multiplier", CSA adder, and a specially designed power supply unit, the operational speed of the entire system has been notably increased. This enhancement underscores the effectiveness of the proposed methodology in achieving superior performance metrics across various parameters.

Based on Figure 7, the proposed structure exhibits a notable decrease in delay, ranging from approximately 47% to 84% compared to [21], [22], and [16], indicating significantly improved efficiency across various moment orders.

Figure 7. Comparison of delay for various moment orders in different structures.

5. Conclusion

In this article, we introduced a novel structure utilizing systolic arrays and pipelines for the computation of high-order two-dimensional moments in grayscale images. Our proposed structure demonstrates remarkable efficiency, with each cell capable of computing the 14th order moment of a 1024×1024 image at an impressive rate of 954 frames per second (fps). Notably, our design achieves low power consumption, registering at only 3.254 mW. Leveraging these promising outcomes, future enhancements could focus on augmenting computational speeds by scaling up the number of computing cells, thereby further optimizing performance.

7. References

[1] Yuh, J. Design and control of autonomous underwater robots: A survey. *Autonomous Robots, 8*, 7-24. (2000)

[2] Terracciano, D. S., Bazzarello, L., Caiti, A., Costanzi, R., & Manzari, V. Marine robots for underwater surveillance. *Current Robotics Reports, 1*, 159-167. (2020)

[3] Liu, J., Gao, J., An, X., & Yan, W. (2019). *Autonomous Landing of an Unmanned Underwater Vehicle using Hybrid Visual Servoing Control with Image Moments and Quaternions.* Paper presented at the OCEANS 2019-Marseille.

[4] Eustice, R. M., Pizarro, O., & Singh, H. Visually augmented navigation for autonomous underwater vehicles. *IEEE Journal of oceanic Engineering, 33*(2), 103-122. (2008)

[5] Forouher, D., Hartmann, J., Klüssendorff, J. H., et al. (2012). *HANSE—A Low-Cost Autonomous Underwater Vehicle.* Paper presented at the Autonomous Mobile Systems 2012: 22. Fachgespräch Stuttgart, 26. bis 28. September 2012.

[6] Humais, M. A. Tools for Detection Tracking and Autonomous Operations by Unmanned Aerial Vehicles. (2020)

[7] Eren, F. (2015). *Pose detection and control of unmanned underwater vehicles (UUVs) utilizing an optical detector array.* University of New Hampshire.

[8] Karampasis, N. D., Spiliotis, I. M., & Boutalis, Y. S. Real-time Computation of Krawtchouk Moments on Gray Images Using Block Representation. *SN Computer Science, 2*, 1-15. (2021)

[9] Hua, X., Hong, H., Liu, J., & Shi, Y. A novel unified method for the fast computation of discrete image moments on grayscale images. *Journal of Real-Time Image Processing, 17*, 1239-1253. (2020)

[10] Raffaitin, C., Romero, J.-S., Romero, J.-S., & Procel, L.-M. (2019). *Hardware implementation of a shape recognition algorithm based on invariant moments.* Paper presented at the Proceedings of the 32nd Symposium on Integrated Circuits and Systems Design.

[11] Stančić, S. M., Popović-Božović, J. S., & Ponjavić, M. M. (2017). *Hls efficiency in the case of image moments algorithm implementation.* Paper presented at the 2017 25th Telecommunication Forum (TELFOR).

[12] Hjouji, A., El-Mekkaoui, J., Jourhmane, M., & Bouikhalene, B. New set of non-separable orthogonal invariant moments for image recognition. *Journal of Mathematical Imaging and Vision, 62*, 606-624. (2020)

[13] Manjunath, K., Prabhu, G., & Siddalingaswamy, P. A quantitative validation of segmented colon in virtual colonoscopy using image moments. *Biomedical Journal, 43*(1), 74-82. (2020)

[14] Majumdar, I., Chatterji, B., & Kar, A. (2020). *A moment based feature extraction for texture image retrieval.* Paper presented at the Information, Photonics and Communication: Proceedings of Second National Conference, IPC 2019.

[15] Shao, X., Liu, N., Wang, Z., Zhang, W., & Yang, W. Neuroadaptive integral robust control of visual quadrotor for tracking a moving object. *Mechanical Systems and Signal Processing, 136*, 106513. (2020)

[16] Di Ruberto, C., Putzu, L., & Rodriguez, G. Fast and accurate computation of orthogonal moments for texture analysis. *Pattern Recognition, 83*, 498-510. (2018)

[17] Salah, A., Hosny, K. M., & Abdeltif, A. M. (2023). A Generic Multicore CPU Parallel Implementation for Fractional Order Digital Image Moments *Recent Advances in Computer Vision Applications Using Parallel Processing* (pp. 1-12): Springer.

[18] Wang, B., Ma, S., Zhu, G., Yi, X., & Xu, R. A novel systolic array processor with dynamic dataflows. *Integration*, *85*, 42-47. (2022)

[19] Zhang, J., & Pan, C. (2020). *New algorithm and its systolic implementation for digital correlation by using first-order moment.* Paper presented at the MIPPR 2019: Parallel Processing of Images and Optimization Techniques; and Medical Imaging.

[20] Monajati, M. Underwater Image Enhancement Using FPGA-Based Gaussian Filters with Approximation Techniques. *International Journal Of Coastal, Offshore And Environmental Engineering (ijcoe)*, *8*(4), 49-58. (2023)

[21] Roma, N., & Sousa, L. (2000). *In the development and evaluation of specialized processors for computing high-order 2-D image moments in real-time*. Paper presented at the Proceedings Fifth IEEE International Workshop on Computer Architectures for Machine Perception.

[22] Iwashita, A., Komuro, T., & Ishikawa, M. An image-moment sensor with variable-length pipeline structure. *IEICE transactions on electronics*, *90*(10), 1876-1883. (2007)