Journal of VLSI Circuits and Systems, ISSN: 2582-1458 Vol. 6, No. 2, 2024 (pp. 115-121)
RESEARCH ARTICLE
WWW.VLSIJOURNAL.COM

VLSI Architecture-Based Implementation of Motion Estimation Algorithm for Underwater Robot Vision System

Aarti Hemant Tirmare[1*], Priyadarshani Shivakumar Mali[2], Amardeep Anandrao Shirolkar[3], Ganesh Rajaram Shinde[4], Vikas Dattatray Patil[5], Hemant Appa Tirmare[6]

*1*Assistant Professor, Department of Electronics And Telecommunication Engineering, Bharati Vidyapeeth's college of Engineering, Kolhapur, Maharashtra*

[2]Assistant Professor, Department of Electronics And Telecommunication Engineering, Bharati Vidyapeeth's college of Engineering, Kolhapur, Maharashtra

[3]Assistant professor, Electronics and Telecommunication Engineering Department of Technology, Shivaji University, Kolhapur, Maharashtra

[4]Assistant Professor, Mechanical Engineering , Department of Technology, Shivaji University, Kolhapur, Maharashtra

[5]Assistant Professor, Department of Electronics And Telecommunication Engineering, Bharati Vidyapeeth's college of Engineering, Kolhapur, Maharashtra

[6]Assistant Professor, Computer Science and Technology , Department of Technology, Shivaji University, Kolhapur, Maharashtra

KEYWORDS:
Motion Estimation,
Underwater Robot Vision,
Computer vision,
FPGA,
Velocity

ARTICLE HISTORY:
Received : 09.09.2024
Revised : 18.10.2024
Accepted : 06.11 .2024

DOI:
https://doi.org/10.31838/jvcs/06.02.13

Abstract

Motion Estimation (ME) is a computationally intricate issue often affected by non-exact solutions. Consequently, it is imperative to embrace many methodologies and assumptions. Velocity is an essential variable for Underwater Robot Vision (URV) navigation, yet its estimation poses significant challenges. Conventional techniques, such as the Global Positioning System (GPS), are ineffective underwater, while alternative approaches need to calculateocean currents' velocity or employ sound sensors. Nonetheless, these systems possess inherent limits. A viable method to calculate the velocity of URVs is through the utilization of Computer Vision (CV) systems when a marine structure is inside the URVs' field of view, as these sensors are not affected by the same issues as acoustic routing or the influence of ocean currents. This study devised an algorithm that estimates a vehicle's rotational velocity (RV) and axial velocity (AV) by utilizing the motion field derived from the optical ME technique. The algorithms were evaluated in simulated settings, and the findings were shown using several approaches for ME. The ME method was found to function as predicted, with a maximum error of 2.23% in the conducted tests, and showed greater robustness under challenging underwater situations. This work also presented a hardware architecture for low-end Field Programmable Gate Arrays (FPGAs) to enable real-time computation of dense optical motion for Video Graphics Array (VGA) images, utilizing the ME method.

Author's e-mail: aartitirmare9@gmail.com, priyadarshaniomali@gmail.com, aas_tech@unishivaji.ac.in, grs.50449@unishivaji.ac.in, vikaspatils@gmail.com, hat_tech@unishivaji.ac.in,
Author's ORCID: A H Tirmare 0009-0004-5567-0711, P.S.Mali 0000-0003-3279-299X, A.A.shirolkar -0009-0002-0933-6494, G.R.Shinde- 0009-0000-0498-3057, V.D.Patil 0000-0001-5700-1360, H.A.Tirmare 0000-0001-8451-935X

How to cite this article: Tirmare AH, Mali PS, Shirolkar AA, Shinde GR, Patil VD, Tirmare HA. VLSI Architecture-Based Implementation of Motion Estimation Algorithm for Underwater Robot Vision System, Journal of VLSI Circuits and System Vol. 6, No. 2, 2024 (pp. 115-121).

Introduction

The resilience of the human visual system in ME across many visual contexts is remarkable since it executes extensive computational tasks constantly, reliably, effectively, and easily. The academic world has extensively researched ME from image sequences because of its significance in several visual tasks. The three-dimensional image is displayed on the image plane, resulting in each point generating a two-dimensional trajectory with an explicit vector velocity. The 2D velocities for all observable

surface locations are typically called a 2D motion field. The ultimate objective of optical flow (OF) estimation is to calculate an estimate of the motion field based on time-varying picture intensity. A variety of real-time methodologies for ME have been suggested,[1] which may be initially categorized into matching domain estimates, energy designs, and gradient methods.[2, 3] Despite the variety of models and methods, none adequately addresses all the challenges related to real-world processing, including noise, variations in lighting, secondary motion, and occlusions.

Methods for measuring OF frequently aim to balance precision and efficiency.[4] This trade-off occurs because the most accurate approaches typically need greater processing resources. With equivalent processing resources, certain strategies provide more exact estimations but at a slower pace, and others achieve less accurate movement predictions more rapidly. Accuracy enables the evaluation of result quality, whereas efficiency pertains to the duration required for data acquisition and the computing resources utilized. In ME, achieving precise findings efficiently in real-time is essential, employing methodologies that may be adapted to practical issues. This discussion pertains to systems associated with ME accelerators, including those utilizing graphics hardware as specialized circuits and FPGAs.[5]

ME may be accomplished by several methods, both vision and non-vision-based, according to the specific application. If other sensors are present, vision-based ME may be superfluous. In underwater vehicles, the presence of a superior USBL), inverse USBL (iSBL), and GPS intelligent beacons (GIB) may render vision-based ME unnecessary, particularly when the robot operates at significant distances from the seafloor.[6] However, if a Remotely Operated Vehicle (ROV) is near the seabed and lacks additional high-precision sensors, utilizing an optical system to estimate its velocity is an effective approach. Other sensors, such as the Doppler Speed Logger (DSL), can complement a vision-based strategy; nevertheless, they are insufficient due to inherent drift accumulation. Numerous studies in the literature employ vision-based guidance in both URVs and ROVs.[7]

The acquisition of the velocity vector necessitates a real-time system, as an acceleration vector that surpasses its temporal validity is ineffective in a control loop. Determining OF and retrieving vehicular velocity is a computationally difficult activity that may not align with the restricted processing capabilities of low-power embedded computer devices often found in tiny URVs. Consequently, acquiring a velocity vector within a practical timeframe becomes exceedingly challenging.[8-9]

This paper aims to design a system that uses OF approaches to estimate the speed of an URV compared to a structure observed by the camera, including the sea bottom. A low-resolution picture is utilized to mitigate the constraints imposed by the substantial computing complexity of existing OF methods. Intense OF is intended to be immediately calculated by a proprietary computer system implemented in an FPGA-based architecture.

Bibliographic review

The navigation and positioning of a URV is a complex endeavor. Modern navigational tools above water depend on radio or spread-spectrum transmission and GPS. But underwater, the sea's tremendous absorption of electromagnetic energy renders radio transmission inaccessible to URVs. This indicates that conventional RF-based systems are unsuitable for URV navigation. Numerous studies in the literature employ CV-based routing in both URVs and ROVs. CV-based alternatives may be categorized as Power, Gradient, OF, and Matching techniques.[10] Specifically, many real-time implementations are available in FPGAs and, more recently, Graphics Processing Units (GPUs). Specifically, real-time OF ME sensors deployed in FPGAs are discussed in.[11] An instantaneous ME for Micro Aerial Vehicles with FPGA was demonstrated by.[12] A contemporary parallel design for OF prediction demonstrated exceptional performance. An intriguing bio-inspired methodology integrating modest primitives, including OF, with intermediate applications, like segmentation and surveillance, has been introduced in.[13] Recently, a gradient-based method utilizing a GPU was proposed. Additional instances of GPU use for matching attributes and monitoring, a potential solution to the ME issue, were shown in.[14]

GPUs are more prevalent and highly beneficial for the real-time execution of URVs. For example,[15] uses a GPU to handle Synthetic Aperture Sonar (SAR) signals onboard a URV. SARs provide high-quality sonar pictures that need substantial computer resources. Consequently, GPUs are crucial for sustaining the URV's real-time operational capacity. The GPU provides a significant advantage for Simultaneous Location and Mapping (SLAM), which underpins reconstruction alongside stereo images. In the future, these technologies will gain further popularity as the sector advocates for enhanced autonomy and durability of automobiles. Visual odometry refers to the method of ascertaining robot placement through the analysis of sequential camera pictures. The methods for acquiring visual odometry are OF and geometry from motion.[6] The

algorithms employed in this navigation method typically function with other systems, including inertial and audio systems. This approach facilitates navigation, enhancing accuracy when near the object of interest. The study[16] employed forward and downward cameras to rectify inaccuracies in dead-reckoning navigation situations. One technique employed in optical directions is the OF derived from the URV's built-in cameras. Reference[17] presents a docking and management system that employs OF for managing and docking a URV. OF is utilized in unmanned robotic vehicles to monitor targets and velocity estimates.[18]

While optical technologies possess advantages compared to acoustic systems, they also present challenges. URVs are subject to power limitations, as they cannot carry substantial batteries for energy supply and must remain submerged for extended durations. Furthermore, URVs have spatial constraints that limit their capacity to house extensive equipment for various functions, resulting in the use of compact, low-power computer systems for image processing, which restricts performance.

ME FOR URV AND VLSI IMPLEMENTATION

Calculating the URV velocity is a complex endeavor since the 3D movement of an item in the surrounding plane when utilizing a monocular camera, is reduced to a 2D projected movement in the camera plane. This indicates that the profundity of the movement is diminished. Furthermore, the motion of an item comprises two elements: axial and rotational. The previously reported OF techniques provide the pixel shift between two movie frames. This movement may be readily transformed into velocity by dividing it by the frame duration (1=fps), as seen in equation (1).

$$V\left(\frac{pixels}{s}\right) = (u, v)/\left(\frac{1}{fps}\right) \quad (1)$$

It is feasible to convert pixel velocity to velocity inside the benchmark frame of the URV. The velocity is inversely related to the camera's proximity, a phenomenon referred to as the parallax impact.

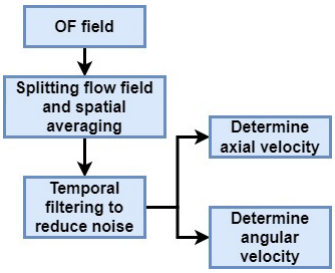


Fig. 1: Flow diagram detailing the procedures employed to extract the URV's velocity from the OF field

Fig. 1 illustrates a flow diagram detailing the procedures to extract the URV's velocity from the OF field. The initial stage involves splitting the OF field into parts to facilitate subsequent computations and eliminate incorrect vectors. The subsequent stage involves applying a low pass filter in the temporal field, which mitigates the perceived motion of the URV and removes high-frequency noise. Subsequently, the assessment of axial velocity occurs alongside the evaluation of angular velocity.

Temporal filtration and velocity transformation

The slower kinematics of a URV constrain the pace at which its velocity and trajectory may change. Consequently, a low pass filter (LPF) was applied to the velocity vectors (VV) from every movement area. This LPF is a straightforward moving average (SMA) in the time domain. This mitigates rapid changes in direction and velocity caused by vibrations and minor unexpected motions. Unless otherwise indicated, the SMA size has been empirically determined to be four frames. However, it may be adjusted according to changing navigation circumstances. At this phase, equation (1) is employed, transforming the movement vectors of each sector into VV. Given the short duration among the frames, one can consider these vectors instantaneous VV.

Estimation of AV

The velocity in both directions of motion constitutes the AV. The movement field is condensed to 16 VV from a 4 x 4 grid, with the vector denoting the URV's velocity being the average of all 16 VV. The averaging procedure also mitigates the rotation of the URV around the z-axis. After calculating the vector denoting AV in pixels per second (pps), it has been converted that velocity to m/s within the URV basis frame. The procedure assumes the employment of a pinhole camera equipped with a symmetrical lens, as seen in Fig. 2.

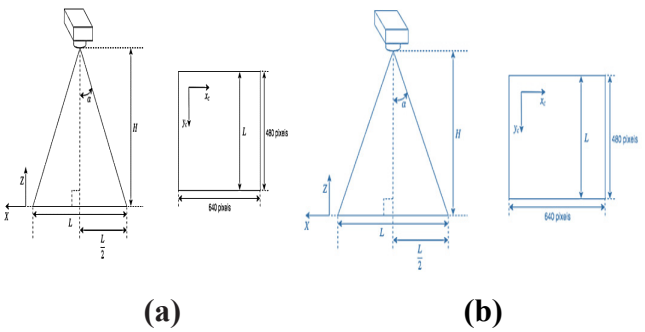


Fig. 2: (a) cross-sectional depiction of the camera's view angle and (b) The pixel dimensions of the frame

Fig. 2(a) is the cross-sectional depiction of the camera's view angle (VA). The angle is 50% of the camera's VA.

represents the distance from the lens to the object, utilized for calculating the velocity. represents the actual dimension of the VA along the x-axis of the camera. The pixel dimensions of the frame are indicated in Fig. 2(b).

Compute with values of and is given as,

$$\propto = \arctan\left[\frac{\frac{L}{2}}{H}\right] \quad (1)$$

With the angle established, it is now feasible to calculate the value of , denoted as , since it may vary with the value for any given , represented as :

$$L(t) = 2 * \tan(\propto) * H(t) \quad (2)$$

With , one can establish a direct correlation between a pixel length and its position in the external standard frame. In this instance, pertains to the x-direction, which measures 480 pixels in dimension.

$$transition\ factor = \frac{L(t)}{480}\ m/pixel \quad (3)$$

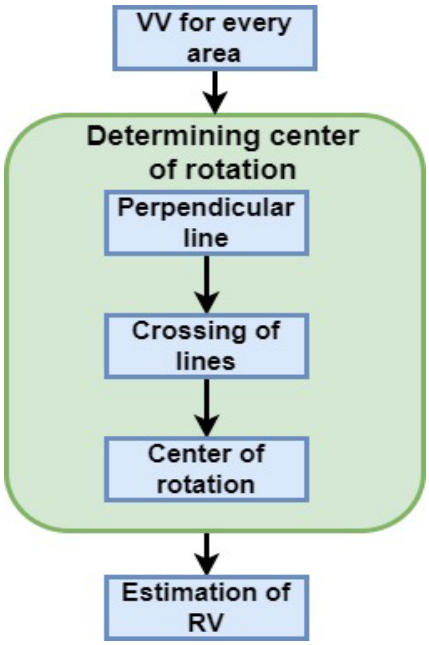
The VV in only needs to be multiplied by the transition factor at this point to transform the velocity in pixels to meters, irrespective of the velocity of the pixels and the camera. The conversion of this velocity, specifically the camera framework, to the URV reference frame can be achieved by:

$$x = y_c \quad (4a)$$

$$y = x_c \quad (4b)$$

Estimation of RV

The RV of the URV denotes the velocity along the z-axis. The movement field was segmented into regions, with each area characterized by a VV that denotes its mean speed. Subsequently, temporal filtering was applied to mitigate the noise. When examining the motion sequence of the URV during rotation and disregarding movable and deformable maritime systems, it is permissible to consider a rigid structure rotation, indicating that the rotation can be represented as a planar transition. In this instance, the body is the reference framework for calculating velocity. Another aspect to remember is that the VV of every area is expressed in the lens's standard frame, , , and . Therefore, after calculating the RV, it must be converted to the URV's standard frame. This can be executed by reversing the RV at the conclusion, i.e., . In a planar shift in a rigid structure, the RV can be calculated as outlined in the process diagram of Fig. 3 (a).



(a)

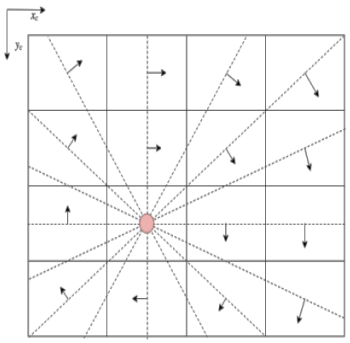


Figure 4.10: The red dot is the centre of rotation which coincides with the intersection of the perpendicular lines to the velocity vectors of each region.

(b)

Fig. 3: (a) Flow diagram detailing the procedures employed to estimate RV (b)The circle in red represents the center of rotation, aligning with the point that intersects the perpendicular lines to the VV of every area.

The initial step in calculating RV is identifying the center of rotation, specifically the instantaneous center of a planar movement. This entails calculating the perpendicular line for every movement vector, represented by the dotted line in Fig. 3(b) and subsequently determining the point of intersection of every line with every other line. In an ideal scenario, the crossing of the line yields a singular moment that denotes the center of rotation, illustrated by the red circle in Fig. 3(b). However, due to the non-

ideal nature of the VV, an accumulation of intersecting points will manifest at the center of rotation. The center of rotation can be determined by calculating the centroid of the point cluster. Subsequently, the RV can be calculated by determining the separation between the center of rotation and the beginning of the VV, followed by estimating the mean of all angular speeds.

RESULTS

During the initial phases of ME, the movement field was partitioned into 16 identical areas, and the median VV was calculated for every area, employing a threshold of one pixel, thereby considering only movements exceeding one pixel. A SMA of the four recent values was employed for temporal filtering. An image with a VGA resolution measuring 640x480 pixels was utilized in this experiment. The visual sequences employed in these experiments were generated using Matlab.

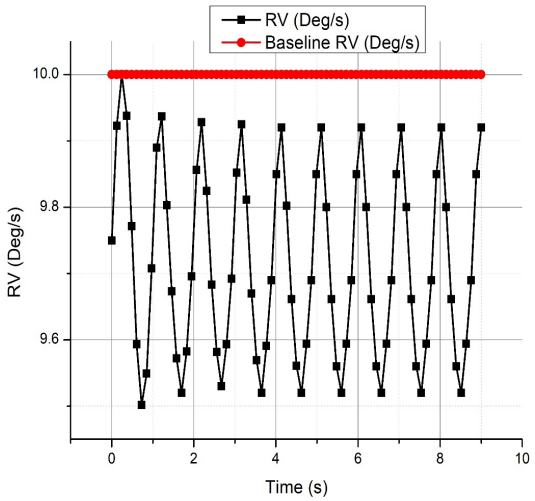
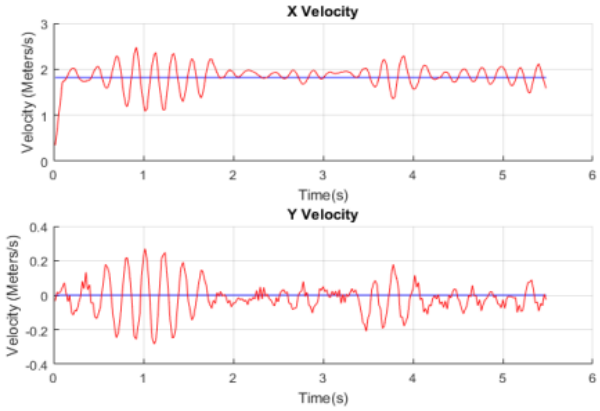


Fig. 4: Computation of RV

Fig. 4 illustrates the RV calculated with the center of rotation outside the camera's VA. The red line at 10 /s represents the baseline RV of the camera, while the black line denotes the determined RV. Fig. 4 indicates that the error in the estimation of RV has risen to 2.23%, corroborating previous statements. As the center of rotation is inaccurately estimated, it affects the estimation of RV. This error arises because maintaining a consistent RV while distancing the center of rotation from the image's center results in an increased velocity of the pixels within the frame. Consequently, the precision of OF diminishes, resulting in increased noise and, therefore, a heightened error in RV at elevated AVs. As velocity rises, this estimation begins to yield a greater error.



(b) Computed velocity of the car in the x and y direction (red) and true velocity (blue).

Fig. 5: Computation of AValong x and y directions

Fig. 5 illustrates that the trajectory is predominantly accurate. As previously stated, the URV traversed a distance of 12 m along the x-axis and 0 m along the y-axis; consequently, the final computed distances are 11.12 m along the x-axis and 0.15 m along the y-axis, yielding an average error of 0.45%. Fig. 5 illustrates the elements of the URV's velocity. Numerous fluctuations in this element can be attributed to the camera's vibrations. Nonetheless, the calculated velocity is proximate to the actual velocity, depicted in blue.

FPGA implementation

This study introduced a hardware design tailored for low-end FPGAs to facilitate actual-time processing of intense optical movement for VGA images, employing the ME method.

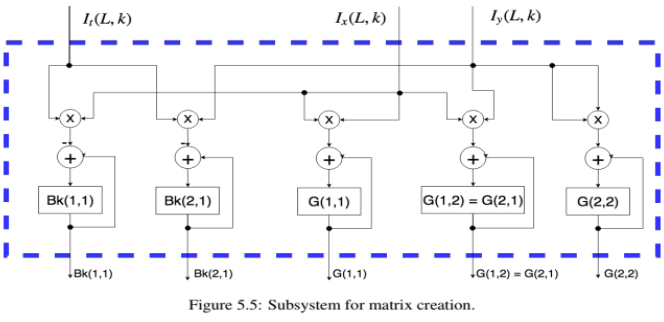


Figure 5.5: Subsystem for matrix creation.

Fig. 6: Matrix generator architecture

The matrix can be calculated using the ME algorithm. Fig. 6 requires five additions, five multipliers, and five accumulators to implement the matrix generatorsystem iteratively.

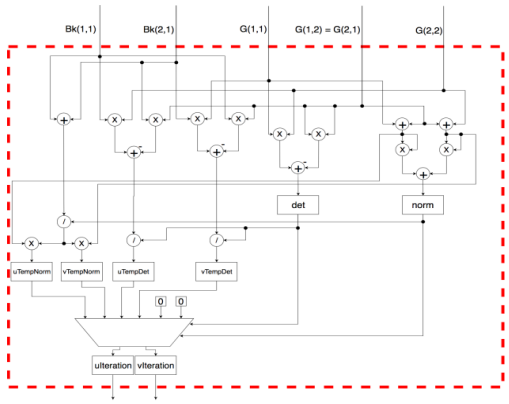


Figure 5.6: Architecture proposal for the solving the system.

Fig. 7: Matrix solver architecture

In the matrix solver shown in Fig. 7, the least squares problem must be addressed while concurrently calculating the determinant of matrix G and the mean of the gradient. It will require seven addition functions, ten multiplying factors, and three dividers. We must determine which computed value, derived from the two distinct methods, to utilize or if neither method successfully calculated the OF for that pixel. A multiplexer selects the value to be utilized at the conclusion based on the determinant and the average. Resource usage estimation has been shown in Table 1.

Table 1: Resource usage estimation

Resource	Matrix Solver	Matrix Generator	Gaussian Smoothing Kernel	Spatial Derivatives	Temporal Derivative	Image Warping Process	Total
Adders/Subtractors	7	5	24	8	1	2	47
Multipliers	10	5	20	-	-	-	35
Dividers	3	-	-	8	-	-	11
Memory (MBytes)	-	-	-	-	-	-	9.82

Memory Breakdown:

- **Images and Spatial Derivatives**: 4.9 MBytes for three images and derivatives (based on pyramid levels)
- **OF Storage**: 4.92 MBytes for two OF fields (u and v, current and previous frames)

Total memory estimation: **4.9 MByte + 4.92 MByte = 9.82 MBytes**

Conclusion

A feasible approach to determine the velocity of URVs is by employing CV systems when a marine structure is within the URVs' line of sight, as these sensors are unaffected by the challenges associated with acoustic routing or the impact of ocean currents. This research developed an algorithm that calculates a vehicle's RV and AV using the motion field obtained through the optical ME technique. The algorithms were assessed in simulated environments, and the results were presented using various methodologies for ME. The ME method performed as anticipated, exhibiting a maximum error of 2.23% in the tests conducted, and demonstrated enhanced robustness in demanding underwater conditions. This study introduced a hardware architecture tailored for low-end FPGAs to facilitate real-time computation of dense optical motion for VGA images, employing the ME method.

References

[1] Zheng, Y., Yu, Z., Wang, S., & Huang, T. (2022). Spike-based motion estimation for object tracking through bio-inspired unsupervised learning. *IEEE Transactions on Image Processing*, *32*, 335-349.

[2] Salman, S. A., Mustafa, S. S., &Gathban, S. A. (2020). Block Matching of Motion Estimation by using New Technique "Quintet Search Algorithm". *Journal of Engineering and Applied Sciences*, *15*(6), 1346-1350.

[3] Pradeep, K., & Veeraiah, N. (2021, June). VLSI implemetation of euler number computation and stereo vision concept for cordic based image registration. In *2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT)* (pp. 269-272). IEEE.

[4] Rajesh, M., Nagaraja, S.R., & Suja, P. (2024). Multi – Robot Exploration Supported by Enhanced Localization with Reduction of Localization Error Using Particle Swarm Optimization. *Journal of Wireless Mobile Networks, Ubiquitous Computing, and Dependable Applications (JoWUA)*, 15(1), 202-215.

[5] de Castro, M., Osorio, R. R., Vilariño, D. L., Gonzalez-Escribano, A., & Llanos, D. R. (2023). Implementation of a motion estimation algorithm for Intel FPGAs using OpenCL. *The Journal of Supercomputing*, *79*(9), 9866-9888.

[6] Pan, S., Xu, X., Zhang, L., & Yao, Y. (2022). A novel SINS/USBL tightly integrated navigation strategy based on improved ANFIS. *IEEE Sensors Journal*, *22*(10), 9763-9777.

[7] Sivayazi, K., & Giriraj, M. (2024). Dynamic Path Planning Algorithm for Mobile Robots: Leveraging Reinforcement

Learning for Efficient Navigation. *Journal of Internet Services and Information Security*, 14(2), 226-236.

[8] Gehrig, M., Millhäusler, M., Gehrig, D., &Scaramuzza, D. (2021, December). E-raft: Dense optical flow from event cameras. In *2021 International Conference on 3D vision (3DV)* (pp. 197-206). IEEE.

[9] Maurelli, F., Krupiński, S., Xiang, X., &Petillot, Y. (2022). URV localisation: a review of passive and active techniques. *International Journal of Intelligent Robotics and Applications*, *6*(2), 246-269.

[10] Xin, C., Wang, C., Xu, Z., Wang, J., & Yan, S. (2023). Marker-free fatigue crack detection and localization by integrating the optical flow and information entropy. *Structural Health Monitoring*, *22*(2), 1008-1026.

[11] Green, K., and R. Vrba. "Research on Nano Antennas for Telecommunication and Optical Sensing." National Journal of Antennas and Propagation 6.2 (2024): 1-8.

[12] Brebion, V., Moreau, J., &Davoine, F. (2021). Real-time optical flow for vehicular perception with low-and high-resolution event cameras. *IEEE Transactions on Intelligent Transportation Systems*, *23*(9), 15066-15078.

[13] Elouaret, T., Colomer, S., De Melo, F., Cuperlier, N., Romain, O., Kessal, L., & Zuckerman, S. (2023). Implementation of a Bio-Inspired Neural Architecture for Autonomous Vehicles on a Multi-FPGA Platform. *Sensors*, *23*(10), 4631.

[14] Xing, X., Yongjie, Y., & Huang, X. (2021, January). Real-time object tracking based on optical flow. In *2021 International Conference on Computer, Control and Robotics (ICCCR)* (pp. 315-318). IEEE.

[15] Gerg, I. D., Brown, D. C., Wagner, S. G., Cook, D., O'Donnell, B. N., Benson, T., & Montgomery, T. C. (2020, October). GPU acceleration for synthetic aperture sonar image reconstruction. In *Global Oceans 2020: Singapore–US Gulf Coast* (pp. 1-9). IEEE.

[16] Mu, X., Yue, G., Zhou, N., & Chen, C. (2022). Occupancy grid-based AUV SLAM method with forward-looking sonar. *Journal of Marine Science and Engineering*, *10*(8), 1056.

[17] Vu, M. T., Choi, H. S., Nhat, T. Q. M., Nguyen, N. D., Lee, S. D., Le, T. H., & Sur, J. (2020). Docking assessment algorithm for autonomous underwater vehicles. *Applied Ocean Research*, *100*, 102180.

[18] Masmitja, I., Martin, M., O'Reilly, T., Kieft, B., Palomeras, N., Navarro, J., & Katija, K. (2023). Dynamic robotic tracking of underwater targets using reinforcement learning. *Science robotics*, *8*(80), eade7811.

[19] Abdullah, Dahlan. "Recent Advancements in Nanoengineering for Biomedical Applications: A Comprehensive Review." Innovative Reviews in Engineering and Science 1.1 (2024): 1-5.

[20] Prasath, C. Arun. "Energy-Efficient Routing Protocols for IoT-Enabled Wireless Sensor Networks." Journal of Wireless Sensor Networks and IoT 1.1 (2024): 1-5.

[21] Muralidharan, J. "Optimization Techniques for Energy-Efficient RF Power Amplifiers in Wireless Communication Systems." SCCTS Journal of Embedded Systems Design and Applications 1.1 (2024): 1-5.

[22] Abraheem, Sudad, Nadia J. Fezaa Al-Obedy, and Amal A. Mohammed. "Some methods to approximate and estimate the reliability function of inverse Rayleigh distribution." Results in Nonlinear Analysis 7.3 (2024): 19-28.

[23] Kavitha, M. "Advances in Wireless Sensor Networks: From Theory to Practical Applications." Progress in Electronics and Communication Engineering 1.1 (2024): 32-37.