Received January 31, 2022, accepted February 26, 2022, date of publication March 2, 2022, date of current version March 10, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3156271

Sonar FoV Segmentation for Motion Estimation Using DL Networks

JOSÉ E. ALMANZA-MEDINA, (Graduate Student Member, IEEE), BENJAMIN HENSON, (Member, IEEE), AND YURIY V. ZAKHAROV, (Senior Member, IEEE)
Department of Electronic Engineering, University of York, York YO10 5DD, U.K.

Corresponding author: José E. Almanza-Medina (jeam502@york.ac.uk)

The work of José E. Almanza-Medina was supported by the Mexican National Council of Science and Technology (CONACyT). The work of Benjamin Henson and Yuriy V. Zakharov was supported in part by the U.K. Engineering and Physical Sciences Research Council (EPSRC) under Grant EP/R003297/1 and Grant EP/V009591/1.

ABSTRACT Accurate navigation of an autonomous underwater vehicle is important for its reliable operation. However, this task is challenging due to limitations of radio wave propagation and poor visibility in the aquatic environment. Underwater navigation techniques based on analysing sonar images facilitated by machine learning have shown promising results. However, previously proposed techniques are still complicated for real-time applications. This paper investigates low complexity techniques for the motion estimation based on the use of images obtained by a sonar looking down to the seafloor. The sonar can use multiple beams within a field of view (FoV). Various configurations of beams are considered according to portions of the FoV covered and two estimation approaches are investigated. In one approach, the sonar images are directly processed by a deep learning (DL) network, whereas in the other, the images are converted into (reduced size) vectors before applying them to a DL network. The vector approach shows a significantly lower computation time (about 10 times faster), which makes it suitable for real-time applications. Both the approaches show a similar estimation accuracy, about 10% of the maximum magnitude of the motion. The vector technique has been used to estimate a simulated trajectory and compare the estimate with the ground truth, which showed a good match. It has also been applied to estimate the trajectory of an imaging sonar from a real data set from a ship's hull inspection. The estimated trajectory has successfully been used to build a mosaic by merging the sonar images from the real data set.

INDEX TERMS Deep learning, motion estimation, sonar, underwater navigation.

I. INTRODUCTION

The underwater environment covers a large part of the planet. However, only a small portion has been explored [1]. Autonomous Underwater Vehicles (AUVs) and Remotely Operated Vehicles reduce risks to which humans are exposed and give significant assistance in recognition tasks, playing an important role in underwater exploration [2], [3]. However, AUVs need accurate navigation for reliable operation [1], [4].

Underwater navigation is the process of acquiring the position of an underwater platform in absolute or relative coordinates with respect to the environment [5]. Applications requiring underwater navigation include exploration of the deep ocean [6], naval surveillance [7], aquatic habitat surveys [8], dam, harbor and ship inspections, military actions [9], etc. However, autonomous underwater navigation is still an open problem. The radio signals from satellites used in systems such as Global Navigation Satellite Systems (GNSS) are highly attenuated underwater [10]. To solve this problem, [11] considers deploying buoys equipped with GNSS. The buoys transmit their positions to users or devices underwater to calculate the user position through triangulation. Similarly, the work [12] proposes a method that uses a constellation of underwater acoustic devices with known positions to estimate the position of underwater platforms. In [13], a method is presented for navigation, which consists of reducing the images from a 3D multi-beam sonar to a depth map that is compared with a pre-registered reference map of the bottom.

Traditional methods for underwater navigation are based on using an inertial navigation system (INS) [14], [15] and

The associate editor coordinating the review of this manuscript and approving it for publication was Byung-Gyu Kim.

Doppler velocity log (DVL) [16]. However, an INS is highly affected by drift errors and need the vehicle to return to the surface for resetting its position [12], [15]. The DVL is highly sensitive to the external environment conditions [16], it is expensive and not suitable for mounting on many vehicles due to its size [17]. Two techniques for navigation based on acoustic ranging are Long Baseline (LBL) [18] and Ultra Short Baseline (USBL) [19]. LBL systems have a restricted area of operation and are difficult to deploy and calibrate [20]. USBL systems require an array of hydrophones on a ship to give the position information to the underwater vehicle and have a high energy consumption [21].

Visual odometry [22], [23] is often used as a complement for other sensors, but visual based navigation methods alone are inaccurate when the visibility is poor [24], [25]. Acoustic imaging techniques are not affected by these conditions and techniques for motion estimation using sonar images have been developed. For instance, Forward-Looking Sonar (FLS) images are used for underwater target tracking [26] and motion estimation approaches with deterministic [10], [25], [27], [28] or deep learning (DL) techniques [29].

Combinations of the above mentioned methods have been explored in order to improve the navigation accuracy. The work [30] proposes mixing the information from a sonar and an inertial measurement unit (IMU). In [31], an IMU, DVL and USBL are combined. Furthermore, in [32], even more methods are combined, by processing data from IMU, DVL, LBL, and pressure and attitude sensors. In [33], authors propose a method that combines visual odometry with IMU measurements. The work [1] proposes combining the information from aerial images with the images obtained by an FLS.

The combination of multiple methods can alleviate the problems associated with the original methods but at the cost of increasing the whole system complexity, making it expensive and harder to deploy. Some systems attempt to reduce the complexity and cost, for example [34] presents the use of two low-cost inertial sensors and a sonar (altimeter). In [35], FLS images and an IMU are used with an adaptive Kalman filter to replace a DVL. In a similar way, [36] uses an INS and replaces the DVL by extracting the dynamic model of the vehicle using Newton-Euler equations.

The aim of this work is to develop a motion estimation method that can be suitable for real-time applications, keeping a high accuracy. Following the previous work [29], the basis of this technique is to perform the motion estimation using a DL network trained with sonar images generated by a simulator. However, instead of using an FLS, the source of information is a looking down sonar whose measurements are pre-processed in different ways. This simplifies the DL network reducing the size of data that the DL network processes. This is done in two steps: (i) a portion of the sonar field of view (FoV) is selected and used to generate images; (ii) the images are converted into vectors and grouped together as part of the pre-processing.

There are two contributions in this paper:

1) The first contribution is a technique that consists in converting sonar images into vectors by adding up the pixels in each row. This reduces the size of the DL networks that are used to estimate the motion.
2) The second contribution is the division of the FoV of a simulated sonar into four quadrants. An image is generated from each quadrant. This is combined with the vector technique by converting the images into vectors and grouped together as the input of the DL network.

The proposed vector-based technique has shown good results, maintaining the millimeter accuracy level when validated using simulated data. To evaluate the method using real data, sonar images were processed by the vector technique (taking into account the specific of the data available from a real sonar). The result is presented by recovering the full trajectory using the real data set and plotting a mosaic by merging all the sonar images.

The rest of this paper is organized as follows. In Section II, the DL network, a sonar simulator and the data sets to train the network are described. In Section III, the proposed motion estimation methods with different portions of the sonar FoV are described. Section IV presents results and discussion for the DL network validation and trajectory estimation with simulated and real data. Finally, conclusions are given in Section V.

II. DL NETWORK AND SONAR SIMULATOR

This section presents the DL network for motion estimation, the simulator which is used to generate the training data sets, as well as the description of the data sets and the simulated underwater environment.

A. DL NETWORK ARCHITECTURE USING SONAR IMAGES AS THE INPUT

In the previous work [29], the performance of numerous DL networks is evaluated, estimating the motion of an underwater platform using sonar images acquired from a FLS mounted on the platform. The PoseNet network [37] was found to be the most efficient for this application after optimizing the network parameters such as the number of convolutional layers, the size of kernels, etc. The resulting optimized PoseNet for motion estimation with sonar images was named PoseNet-Normx10. The architecture of the PoseNet-Normx10 can be seen in Figure 1 (for simplicity, in the rest of this paper, it will be called PoseNet).

The input of the network is an image obtained by concatenating two consecutive sonar images. The input image passes through a series of 9 convolution layers. Parameters of the convolutional layers can be seen in Table 1. The first 8 convolutional layers have a ReLU activation function and batch normalization. An average pooling layer with an averaging window of size 4 is connected to the last convolutional layer. The values in the column "Output size per channel" depend on the input image size. In Table 1, the values are

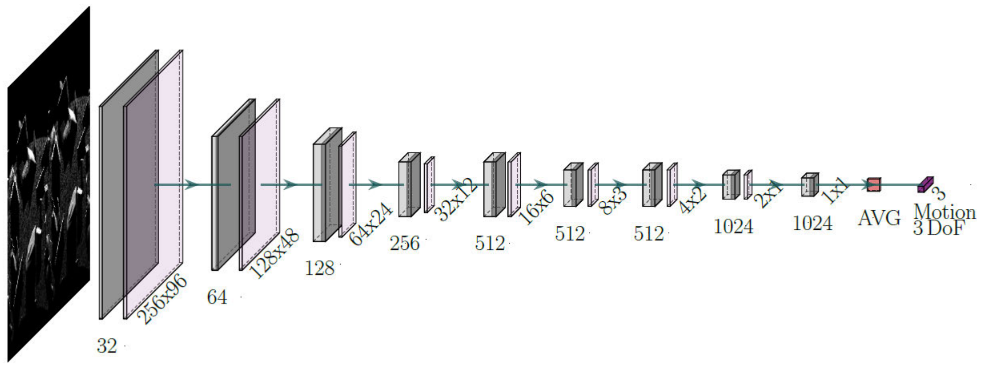


FIGURE 1. Architecture of the DL network PoseNet as presented in [29]. The grey rectangles represent convolutional layers and the semitransparent purple rectangles represent the ReLU function and batch normalization. The average (AVG) pooling layer is represented with a red rectangle and the output regression layer is the purple rectangle at the very right of the diagram.

TABLE 1. Parameters of PoseNet layers as presented in [29].

Layer	Kernel	Channels	Stride	Output size per channel	ReLU & Batch norm
Conv 1	7×7	32	2×2	256×1000	Yes
Conv 2	5×5	64	2×2	128×500	Yes
Conv 3	3×3	128	2×2	64×250	Yes
Conv 4	3×3	256	2×2	32×125	Yes
Conv 5	3×3	512	2×2	16×63	Yes
Conv 6	3×3	512	2×2	8×32	Yes
Conv 7	3×3	512	2×2	4×16	Yes
Conv 8	3×3	1024	2×2	2×8	Yes
Conv 9	3×3	1024	2×2	1×4	No
Average	4×4	1024	1×1	1×4	No

shown for the image size of 512 × 2000 pixels (two concatenated images of 512 × 1000 pixels), which is the size of the first configuration described below in Section III. An output regression layer is connected to the average layer to generate motion estimates for the three degrees of freedom (3 DoF). For simplicity, the scope of this work is to perform motion estimation in the 3 DoF: forward/backward motion, sideways motion and yaw rotation, denoted by $\Delta = [\Delta_x, \Delta_y, \Delta_\theta]$, respectively. It is considered that the platform maintains a constant height from the seafloor when it moves and roll and pitch rotations are too small to be considered [25], [28]. The loss function in the regression layer is the Mean Squared Error (MSE).

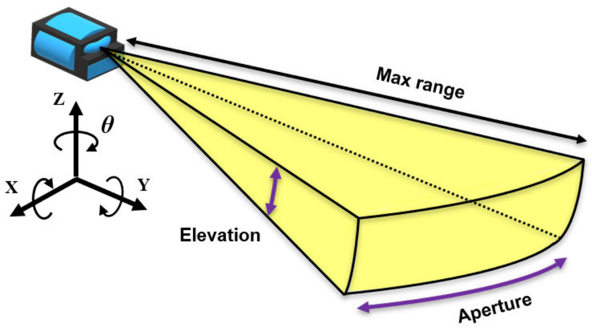


FIGURE 2. The sonar in the simulator is characterized by the aperture and elevation angles, and the maximum range that it can measure.

B. SONAR SIMULATOR

The sonar simulator described in [38] uses ray-tracing to generate sonar images. The underwater vehicles and sonar sensors are implemented in the development platform Unity [39]. The MATLAB programming platform [40] is used to generate images from data provided by Unity. The simulator allows the adjustment of the image noise level, number of pixels, pixel intensity, etc.

A simulated sonar sends multiple rays in a predefined FoV specified by aperture and elevation angles, and the maximum range (see Figure 2).

The rays are equally separated in the elevation and aperture angles. The ray separation is a parameter that can be set in the simulator and it is independent for the aperture and elevation dimensions (see Figure 3). When any of the rays hits an object, the sonar measures the range to the hit point and stores this value along with the angle of incidence on the surface of the hit object, a reflectivity value, the angle values in the elevation and aperture dimensions and the position and orientation of the sonar sensor in the underwater environment. To generate a sonar image, range intensities are

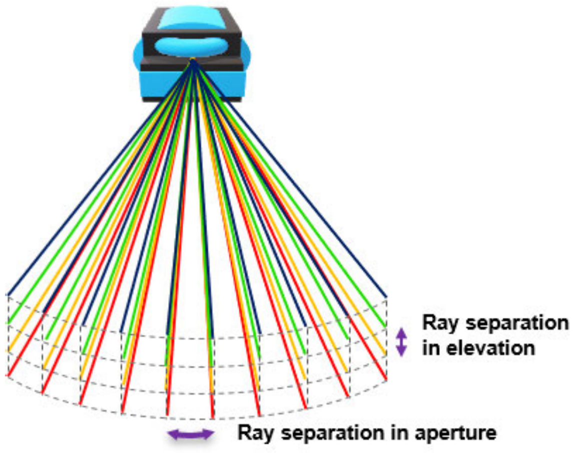


FIGURE 3. The simulated sonar FoV is made of multiple rays that are sent from the sonar with a defined separation in the aperture and elevation dimensions.

'binned' based on range values. The intensities of rays along the elevation are summed together. Rows and columns of the resulting image represent the range and aperture of the sonar, respectively.

The simulator uses a hop-and-generate process, which means that when the platform with the sonar follows a trajectory, it stores the data to generate an image in a certain position in the scenario, then it "hops" to another position according to the trajectory by following a rule of *first rotate then translate*. In the new position, it collects the data for the next image and it hops again. The steps are repeated until completing the trajectory.

The noise if needed is added to the final image. The added noise consists of two parts [41]: (i) Gaussian noise is added to the pixels of acoustic shadows in the images; (ii) to represent the scattering noise, the Rayleigh noise is added to the remaining pixels.

The simulator is used to generate large volumes of data with ground truth information for the position and orientation of the sonar platform. Simulated data is used to train a DL network effectively, since such training requires large amount of ground truth data, which would be almost impossible to collect in real sea experiments. In this work, new sonar FoV configurations are considered and implemented in the simulator.

C. GENERATION OF DATA SETS FOR TRAINING THE DL NETWORK

The use of the simulator to generate training and validation sets represents an essential part of the method described in this work. It provides the ground truth data that prevents possible labeling errors introduced when trying to use real data for training. This eliminates data uncertainty of the platform motion and the information collected from the underwater environment to generate the sonar images. Furthermore, a fair validation of a method and comparison of different methods can only be possible using the simulated data accompanied with the ground truth positions.

The training data sets are generated by concatenating pairs of consecutive sonar images into a single image. The images obtained from the simulator are represented in polar coordinates, they are not transformed to cartesian coordinates. The PoseNet described in subsection II-A is trained using supervised learning. Therefore, labels with information on the platform displacement need to be associated with the training samples. The parameters to be estimated by the DL network are $\Delta = [\Delta_x, \Delta_y, \Delta_\theta]$, where Δ_x, Δ_y and Δ_θ are the displacements in x- and y-translation and z-rotation, respectively. The actual displacement of the platform between two consecutive positions is provided by the simulator. However, the units for the translation in x and y axes are in meters, while the rotation units are in degrees (°). In order to give the same weight to the three DoFs, the corresponding displacements are normalized by considering the maximum displacement possible between two platform positions. These normalized values are used as the displacement labels, denoted by $\Gamma = [\Gamma_x, \Gamma_y, \Gamma_\theta]$. The labels are associated with the concatenated images and together they form the data set.

To generate a data set, the simulated underwater vehicle moves in a random scenario. For all the experiments in this work, the maximum displacement between two positions (corresponding to two consecutive sonar images, and assuming the maximum platform velocity 0.42 m/s) is ±20 mm for translation in one direction and the maximum rotation is ±0.45° [29]. The maximum translation value is selected according to the specification of the Bluefin Robotics Hovering Autonomous Underwater Vehicle [42] and an approximation to the observed translation in the real data set described in subsection IV-E. Each scenario is generated as described in [29]. A scenario consists of a flat surface representing the seabed, covered with randomly placed rocks. The size of each rock is randomly defined and it is not bigger than 0.45 m for each of its three dimensions (length, width, and height). An example of a simulated scenario is shown in Figure 4. Given that the trajectory is also randomly generated, it is unlikely that the platform passes through the same position and orientation more than once. However, to avoid any possible similarities, several different scenarios are used. Each data set is generated using 17 scenarios. The data sets are shuffled and split into 95% and 5% for the training and validation, respectively.

III. PROCESSING WHOLE SONAR IMAGES AND IMAGES COMPRESSED INTO VECTORS

A sonar which looks down with a FoV of 100°×100° is simulated. The height (distance from the sonar to the seafloor) is 2.5 m as shown in Figure 5. It can be seen that the platform forward direction corresponds to the x-axis, the sideways direction to the y-axis and the upward direction to the z-axis, which is the rotation axis associated with θ. This coordinate system will be used in the rest of this paper to analyze and validate the proposed techniques.

TABLE 2. Parameters of PoseNetVec layers for the compressed images.

Layer	Kernel	Channels	Stride	Output size per channel	ReLU & Batch norm
Conv 1	7×2	32	2×1	256×2	Yes
Conv 2	5×2	64	2×1	128×2	Yes
Conv 3	3×2	128	2×1	64×2	Yes
Conv 4	3×2	256	2×1	32×2	Yes
Conv 5	3×2	512	2×1	16×2	Yes
Conv 6	3×2	512	2×1	8×2	Yes
Conv 7	3×2	512	2×1	4×2	Yes
Conv 8	3×2	1024	2×1	2×2	Yes
Conv 9	3×2	1024	2×1	1×2	No
Average	4×4	1024	1×1	1×2	No

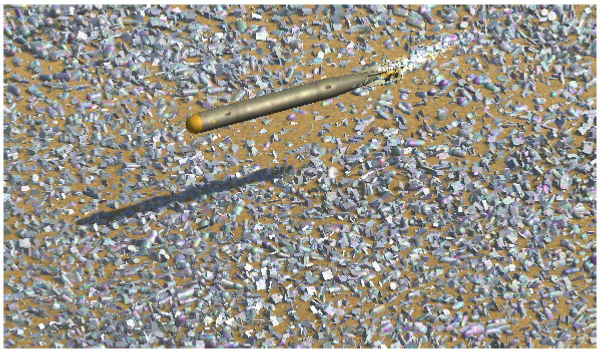


FIGURE 4. Example of the simulated underwater environment with a moving vehicle.

Two techniques for motion estimation are considered:

1) **Processing the whole image (intensity over the aperture-range plane):** Images are generated by compressing the FoV in elevation. This is done by adding up all reflections from objects which are at the same aperture angle and range from the sonar. This process eliminates the elevation dimension.
2) **Processing a compressed image, i.e., a vector (intensity over the range):** In addition to the FoV compression in the elevation dimension, the image is also compressed in the aperture dimension. The result of this compression is a vector with the summation of all the reflections from objects that are located at the same distance from the sonar. The aim of this technique is to reduce the amount of data representing the sonar image and thus reduce the complexity and computing times of the motion estimation system.

A. DL NETWORK ARCHITECTURE USING VECTORS AS THE INPUT

When using the vector technique, every sonar image is converted into a vector, therefore, the input of the DL network is the concatenation of two vectors (one vector obtained from each image), rather than two images. The kernel and stride of each convolutional layer is now changed to avoid a reduction of the number of columns, where each column corresponds to a vector. The summary of the DL network is shown in Table 2. This modified version of the PoseNet is called *PoseNetVec*.

The next subsections describe the use of different portions of the sonar FoV to find a configuration that gives the best estimates of the platform motion.

B. COMPRESSION OF THE FoV INTO A SINGLE VECTOR

The $100^\circ \times 100^\circ$ sonar FoV described above is generated by multiple rays that are sent by the simulated transducer. The separation between rays for the transducer in the simulation is 0.1° in each direction. The value of 0.1° was observed to give a moderate computation time with a good resolution.

To produce the images, the range dimension is discretized into 512 values while the aperture dimension is discretized into 1000 values. The size of the resulting image is 512×1000 pixels (range by aperture). An example of a generated image can be seen in Figure 6. The pixels in the first row of the image correspond to the range of 2 m from the sonar and the pixels in the last row correspond to the range 6.1 m. These values are selected because out of these boundaries there is no information collected by the sonar. For the vector technique, a vector of size 512 is computed by adding up all the pixel values on each row of the image. This produces a vector of 512 range levels, where the first and last levels correspond to ranges (distance from the sonar) of 2 m and 6.1 m, respectively.

C. COMPRESSION OF A HALF OF THE FoV INTO A VECTOR

Given that the sonar is looking down and that images are generated by adding up the data in the elevation angle, the resulting images have objects from both sides of the platform, by overlapping the objects that are at the same distance and aperture angle. An example of overlapping can be seen in the third image in Figure 7. If the rays of only one single side of the platform are used to generate the image, the overlapping can be avoided. For example, the first and second images in Figure 7 are generated with the rays of the right and left side of the platform, respectively. If both images are combined, the result is the third image.

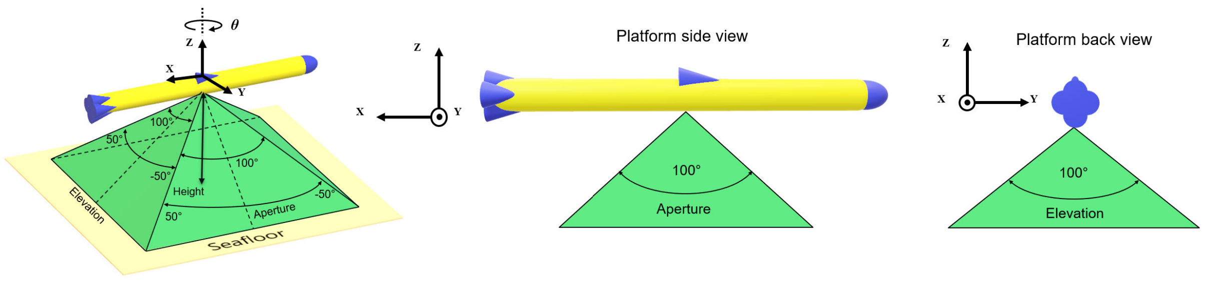


FIGURE 5. Sonar with a looking down FoV of 100° × 100°. The sonar is mounted on a platform whose forward direction corresponds to the negative direction of the *x*-axis.

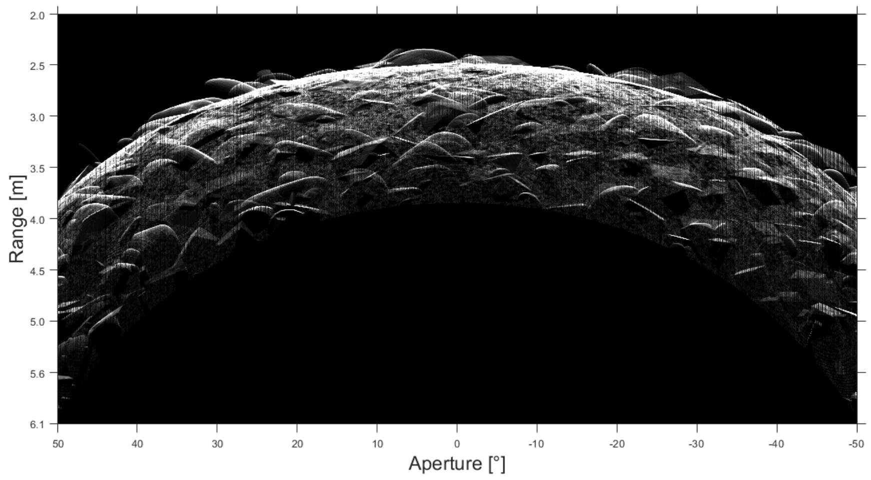


FIGURE 6. Image generated using the FoV of 100° × 100° and a ray separation of 0.1°. This image is compressed in elevation.

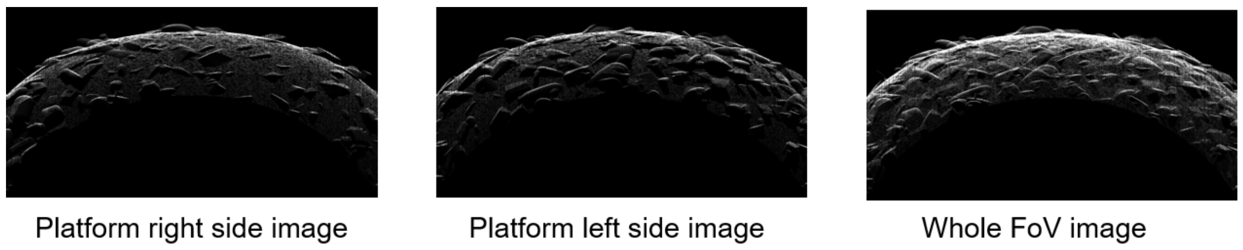


FIGURE 7. Images generated using elevation angles (from left to right) [-50, 0], [0, 50] and [-50, 50]. In the third image, which is the sum of the first two images, it can be seen the overlapping of objects from the first two images.

The overlapping causes the following: when the platform rotates, half of the objects in the images are displaced to one side of the image and the other half of the objects are displaced to the other side of the image, which introduces an ambiguity in the estimation process.

To avoid the overlapping, images are generated using the rays of only one side of the platform. Figure 8 highlights with a blue line the corresponding portion of the FoV. The size of the generated images and vectors are the same as in the case of using the whole FoV.

D. COMPRESSION OF FOUR FoV QUADRANTS INTO FOUR VECTORS

Rather than separating the sonar FoV into two segments, a new configuration that splits the whole FoV into four quadrants is now used. Figure 9 shows the quadrant segmentation.

The images are still generated by adding up the information in the elevation angle corresponding to each quadrant. An example of an image generated using the quadrant segmentation can be seen in Figure 10, which corresponds to Q4.

With the quadrant configuration, two variants are considered to train the DL network:

- **Single quadrant:** The PoseNet is trained using images from quadrant Q4.
- **Four quadrants:** The PoseNet is trained using images from all 4 quadrants. This variant is equivalent to 4 sonars pointing to different directions. For motion estimation, 8 images (2 from each quadrant) are concatenated into a single image which is the input to the

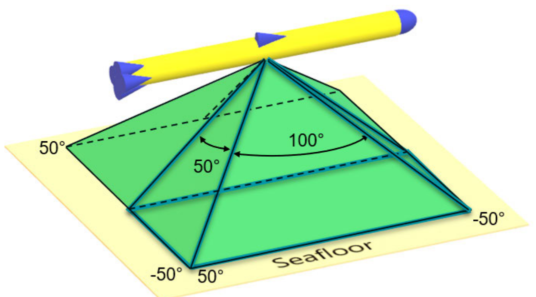


FIGURE 8. The FoV for one side of the underwater platform. The size of the FoV is 100° × 50°.

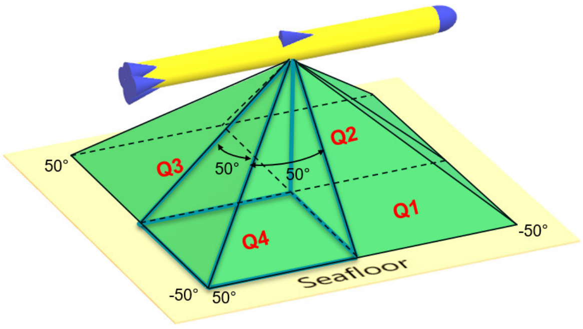


FIGURE 9. Segmentation of the FoV into four quadrants. Each quadrant points to a different direction and has a portion of the FoV of 50° × 50°.

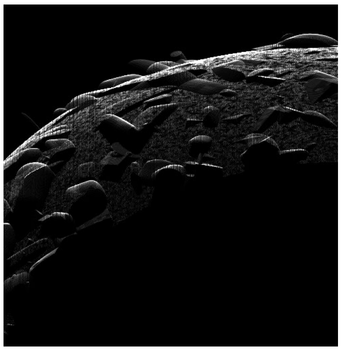


FIGURE 10. Image generated using the information from segment Q4.

DL network. An image example with the concatenation of 8 images and its order is shown in Figure 11. For the vector technique, the DL network receives as input, 8 concatenated vectors (2 from each quadrant), which is equivalent to a 512 × 8 matrix.

E. COMPRESSION OF CIRCULAR SEGMENTS OF THE FoV QUADRANTS INTO VECTORS

The four segments defined in subsection III-D are now cut to a circular shape to form a cone as shown in Figure 12. The rays within the cone are used to generate the sonar images. An example of an image obtained with the circular segment (Q4) is shown in Figure 13. The image is very similar to the one for a quadrant in Figure 10, with the difference that there is slightly less data, as seen in the rounded edges of the illuminated area. Similar to the quadrant configuration, each circular segment produces an image that is concatenated with the consecutive image obtained for the same circular segment, to transfer a total of 8 concatenated images into one.

IV. RESULTS AND DISCUSSION

In this section, results of validating the use of the sonar FoV portions described above are presented.

A. COMPARISON OF THE FoV PORTIONS USING SIMULATED DATA

To evaluate the use of different FoV portions detailed in section III, for each techniques, a data set with 1000 pairs of images/vectors is generated with their motion labels in the three DoFs. The data set is split into 950 training samples and 50 validation samples. The following notation will be used:

- **Whole FoV:** Images and vectors are generated using all the rays in the FoV.
- **Single side:** Images and vectors are generated using the rays of one side of the platform. The selected side corresponds to the segment of the FoV from $-50°$ to $50°$ in aperture and from $-50°$ to $0°$ in elevation as shown in Figure 8.
- **Single quadrant:** The rays from Q1 quadrant are used to generate the images and vectors of the data set.
- **Four quadrants:** All four quadrants are used to generate the images and vectors.
- **Circular segments:** Circular segments from all four quadrants are used to generate the images and vectors.

The root mean square error (RMSE) for the motion estimation is obtained individually for each of the 3 DoF. The DL networks are trained using noiseless images. However, the validation is based either on noiseless images or on images with a high-level noise added to the pixels. Following the considerations specified in subsection II-B, the noise is added with parameters: (i) the Gaussian distribution has a mean and standard deviation of 13.7% and 3.1% of the maximum pixel value, respectively; (ii) the Rayleigh distribution has a scale parameter of 13.7% of the maximum pixel value.

Every time a PoseNet and a PoseNetVec are trained in this work, the following setup is considered. The learning rate at the start of the training is set to 10^{-4} and it is reduced to 0.5×10^{-4} at epoch 12. At epoch 16, it is reduced to 10^{-5}. The training is stopped at epoch 24 or when the validation loss converges. The Adam optimization algorithm is used [43]. As in [29], the MSE loss function is given by

$$\mathcal{L} = \frac{1}{2SR}\sum_{k=1}^{S}\|\widehat{\Gamma_k} - \Gamma_k\|^2, \tag{1}$$

where $\widehat{\Gamma_k} = \left[\widehat{\Gamma_{x_k}}, \widehat{\Gamma_{y_k}}, \widehat{\Gamma_{\theta_k}}\right]$ are estimates obtained by the DL network for each DoF, k is the index that refers to the training samples in the mini-batch, S is the mini-batch size and R is the number of parameters to estimate. For this work, S and R are set to 4 and 3, respectively.

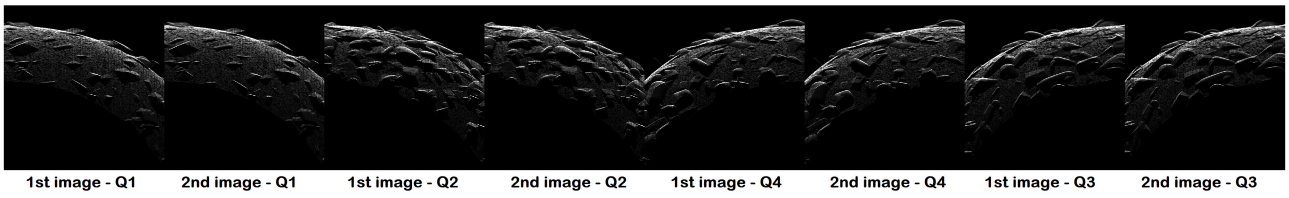


FIGURE 11. Concatenation of eight images (two consecutive images from each quadrant) as input to the DL network.

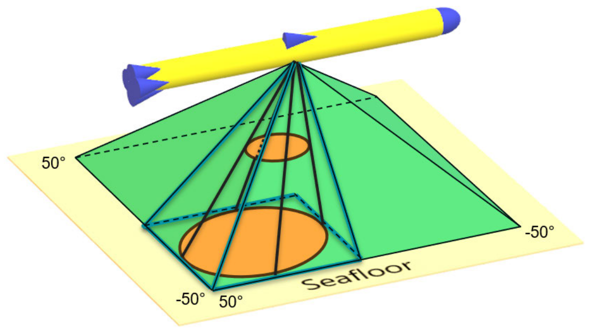


FIGURE 12. Representation of the circular segment.

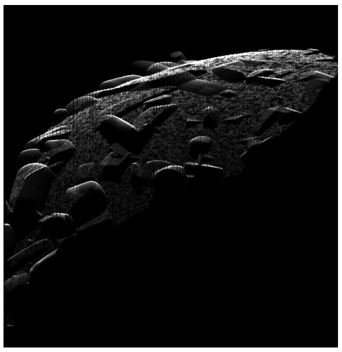


FIGURE 13. Image generated using the information from a single circular segment.

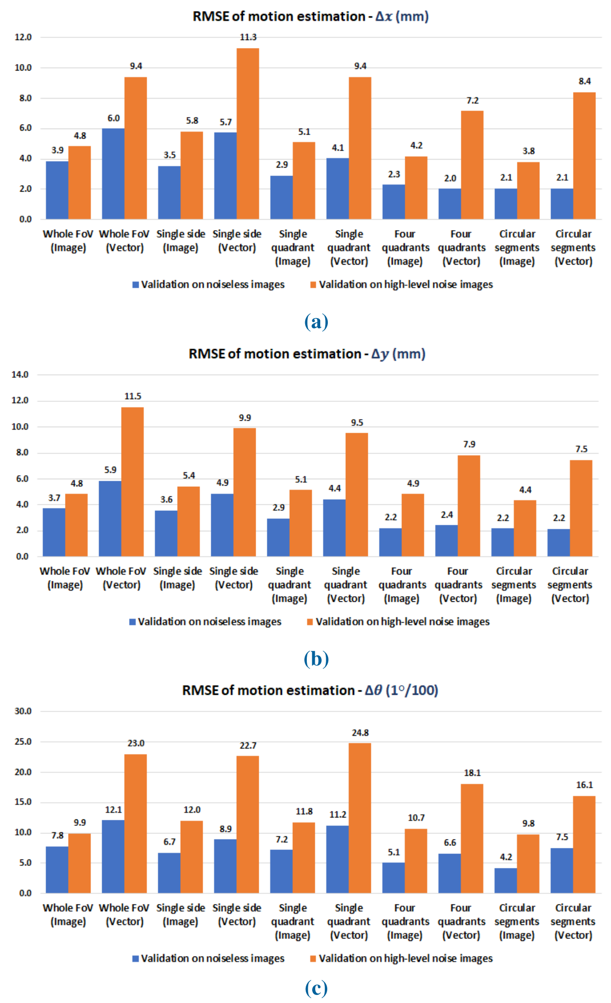


FIGURE 14. Validation RMSE for the motion parameters Δ_x, Δ_y and Δ_θ, shown in (a), (b), and (c), respectively.

Figures 14a, 14b, and 14c show the RMSE of the estimates for the displacement in x and y directions and rotation around z-axis, respectively. The RMSE are shown in pairs, the blue bars correspond to validation with noiseless images and the orange bars correspond to validation with the high-level noise images. Also, Table 3 presents the computation time required to obtain one motion estimate. The time values in the table are averaged over 50 measurements for each of the DL networks. The comparison is done for a standard PC with i5-6500 CPU @3.0GHz processor, 8.0 GB of RAM and without a GPU.

From Figure 14 and the computation times in Table 3 and focusing on validation with noiseless images only, the following observations are made:

- For any FoV configuration, the image technique has a better estimation accuracy than the corresponding vector technique. However, the difference is not high, and, for the best estimation techniques using data from the four quadrants, the difference is negligible.
- In general, the motion estimation in x and y directions show a similar accuracy. This can be due to the same size of the FoV in the aperture and elevation dimensions.
- Splitting the whole FoV into smaller portions results in improvement of the estimation accuracy.
- The vector techniques require an order of magnitude smaller computation time to produce the measurement than the corresponding image technique.

TABLE 3. Average computation time required by the DL networks for one measurement.

Configuration	DL network time (ms)
Whole FoV (Image)	160
Whole FoV (Vector)	16
Single side (Image)	139
Single side (Vector)	17
Single quadrant (Image)	65
Single quadrant (Vector)	13
Four quadrants (Image)	309
Four quadrants (Vector)	27
Circular segments (Image)	307
Circular segments (Vector)	29

- The lowest estimation error is obtained with the configurations of 4 quadrants (for both the rectangular or circular segmentation). The image and vector techniques show an RMSE for translation in either x or y directions slightly above 2 mm and for rotation between 0.042° and 0.051°.
- Using the FoV for one side of the platform is preferable over both sides. It helps to avoid the ambiguity mentioned in subsection III-C.
- The single quadrant configuration further improves the accuracy in estimation of the translations, but not the rotation.
- The four quadrant configuration shows better estimation accuracy compared to the single quadrant and single side configurations, at the cost of a higher computation time.
- The circular segmentation provides results similar to the rectangular segmentation, despite some loss of information due to removing data from the corners of the quadrants.
- The results obtained with quadrant configurations are a hint to future work to develop a system for motion estimation using signals obtained from four single hydrophones facing down.

When validating with noisy images, the motion estimation accuracy presents higher error than the validation with noiseless images. This is expected, given that the networks are trained using noiseless data for both types of validation. However, the obtained RMSE that corresponds to each FoV configuration shows the same trend as in the case of validation with noiseless data described in the points above.

B. RESULTS FOR AN INCREASED DISTANCE OF THE SONAR PLATFORM FROM THE SEAFLOOR

To investigate how the distance can affect the accuracy of the motion estimation, a new data set of 1000 samples is generated using the circular segments in the four quadrants. The difference is that the distance of the sonar to the seafloor is now 10 m rather than 2.5 m. An example of an image obtained with this distance from the seafloor is shown in Figure 15. The pixels in the first row of the image correspond to the range of 9 m from the sonar and the pixels in the last row correspond to the range 24.5 m.

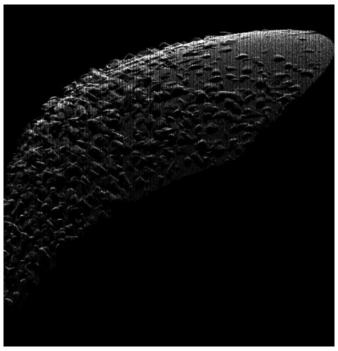


FIGURE 15. Image generated using the information from only one circular segment at a distance of 10 m from the seabed.

Additionally, to reduce the estimation error for validation with noisy images, the image and vector techniques were trained with a data set containing noisy images. For comparison, the training with noisy images was done using the data sets for the cases of 2.5 m and 10 m heights. The noisy images have a high-level noise with the same parameters as described in subsection IV-A.

Figures 16a, 16b, and 16c show validation results for training with noiseless and noisy data at two different heights from the seafloor. It can be seen that the RMSE is significantly reduced when training with high-level noise images compared to training with noiseless data and validating with noisy data. The obtained RMSE is comparable with the case of training and validating with noiseless data for the image and vector techniques. The results give a better idea of what is the approximate error when using real data, given that real images are noisy. Regarding the results of having the sonar at 2.5 m and 10 m from the seafloor, the distance does not affect significantly the estimation accuracy.

C. DEPENDENCE OF THE RMSE ON THE RANGE QUANTIZATION

The number of range levels (quantization levels) in the vectors can be tuned to improve the estimation accuracy. For this, data sets with different number of range levels are generated. When generating the images, the intensities collected by the sonar are spread along range dimension of the image. The chosen number of range levels are a power of 2 (16, 32, ..., 512, and 1024). For validating each number of range levels, a data set of 1000 samples is generated and used to train a PoseNetVec. The data sets are generated for the sonar at a height of 10 m and the circular segment configuration.

Figure 17 presents the RMSE obtained using different number of range levels to generate the images. The curves show that for the vector technique, the best cases are with 128 and 256 range levels, where the error is minimized. The increase of the RMSE while decreasing the number of levels can be due to the poor range resolution when having a small

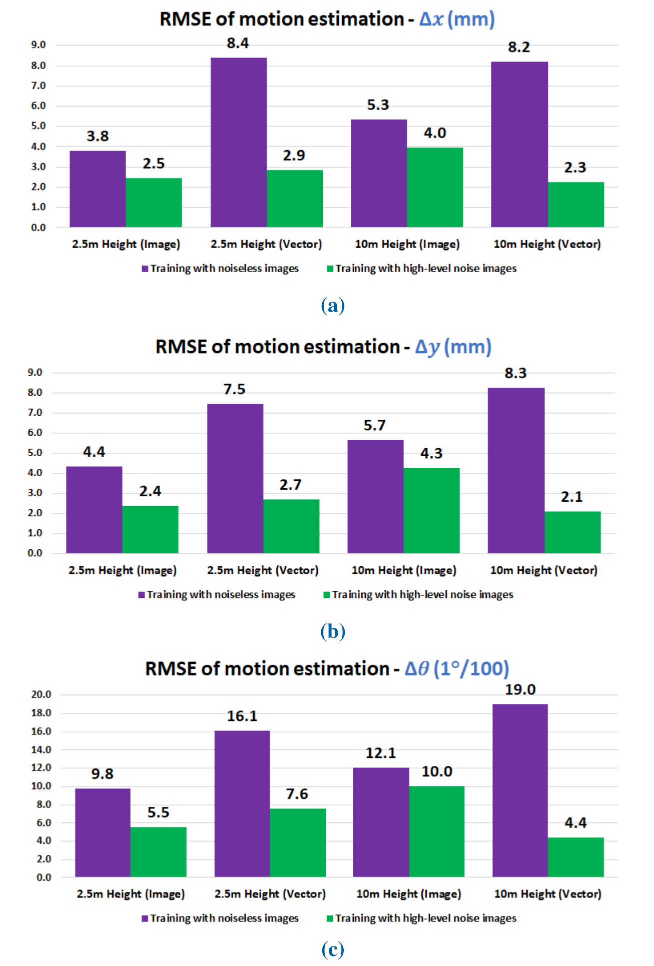


FIGURE 16. Validation RMSE when training with noiseless images and high-level noise images and validating with noisy images for the motion parameters Δ_x, Δ_y and Δ_θ, shown in (a), (b), and (c), respectively.

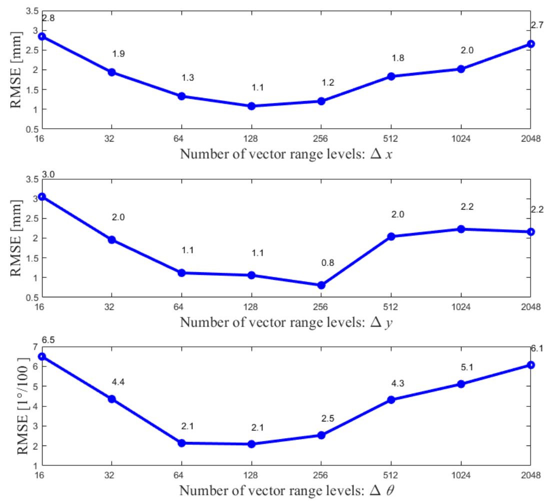


FIGURE 17. RMSE obtained for each motion estimation parameter against the number of range levels for the vector technique with the circular segmentation. The platform's distance from the seabed is 10 m. The training and validation are using noiseless images.

number of levels. This is equivalent to adding some noise to the data. The RMSE increase with an increased number of levels can be caused by the large number of internal parameters to be trained by the DL network.

In the work [29], for the case of noiseless images, an RMSE of around 1 mm is achieved in the direction the FLS is looking, but in the sideways direction the RMSE is always higher than 2.3 mm. The vector technique proposed here with 128 and 256 range levels provides the RMSE for the translation motion (x and y) between 0.8 mm and 1.2 mm, which is an improvement in the estimation accuracy.

For the rotation, the RMSE in [29] in the best case is 0.047°. This is higher than the RMSE obtained by the vector technique, which is around 0.025°, which is again an improvement in the accuracy. These results show that the vector technique can provide a high estimation accuracy. The great advantage of the vector technique is reduction in the complexity of the DL network. This allows a high processing speed and makes the technique suitable for real-time applications.

D. TRAJECTORY RECONSTRUCTION USING SIMULATED DATA

As described in [29], a trained DL network can be used to estimate the displacements between each pair of consecutive images. With these estimates, the platform trajectory can be recovered. The points of the trajectory are represented as x_i and y_i and the orientation of the platform is represented as θ_i, where i is an index that refers to the positions where the images were obtained.

The initial position and orientation x_1, y_1 and θ_1 are set to zero and the other points are calculated using the equations [44]:

$$\begin{aligned}\theta_{i+1} &= \theta_i + \widehat{\Delta_{\theta_i}},\\ x_{i+1} &= x_i + \widehat{\Delta_{y_i}}\sin(\theta_i + \widehat{\Delta_{\theta_i}}) + \widehat{\Delta_{x_i}}\cos(\theta_i + \widehat{\Delta_{\theta_i}}),\\ y_{i+1} &= y_i + \widehat{\Delta_{y_i}}\cos(\theta_i + \widehat{\Delta_{\theta_i}}) - \widehat{\Delta_{x_i}}\sin(\theta_i + \widehat{\Delta_{\theta_i}}),\end{aligned} \quad (2)$$

where $\widehat{\Delta_{x_i}}$, $\widehat{\Delta_{y_i}}$ and $\widehat{\Delta_{\theta_i}}$ are the x- and y-translation and z-rotation estimates for each pair of consecutive images, respectively.

Two examples of trajectories generated with simulated data are shown in Figures 18 and 19. Both trajectories have most of the motion in sideways direction with no rotation. The motion estimates are done using the vector technique. The platform height from the seafloor is 10 m as in the cases described in the previous subsection with two numbers of range levels. The first trajectory is made of 51 platform positions (50 pairs of vectors for estimation) and the number of range levels is 256. The second trajectory is made of 55 positions using vectors of 512 range levels.

It can be seen that both estimated trajectories follow the ground truth trajectory with most of the motion in sideways direction. The recovered trajectory for the case of 256 range levels seems smoother when the platform moves backwards almost at the end of the trajectory, whilst the

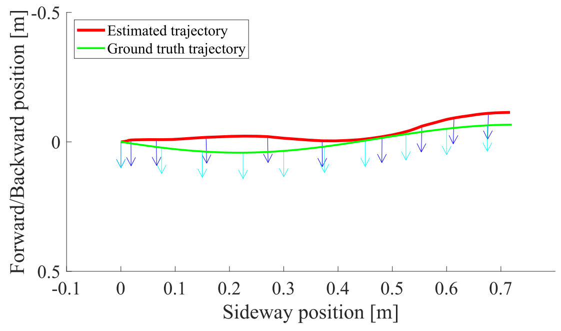


FIGURE 18. First simulated trajectory (256 range levels): Ground truth trajectory (green line), ground truth orientation (cyan arrows), estimated trajectory (red line), estimated platform orientation (blue arrows) using 50 pairs of vectors of simulated data.

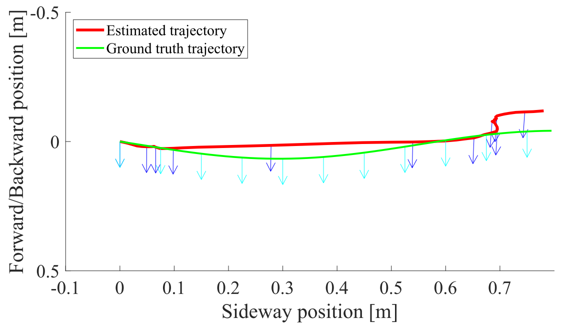


FIGURE 19. Second simulated trajectory (512 range levels): Ground truth trajectory (green line), ground truth orientation (cyan arrows), estimated trajectory (red line), estimated platform orientation (blue arrows) using 54 pairs of vectors of simulated data.

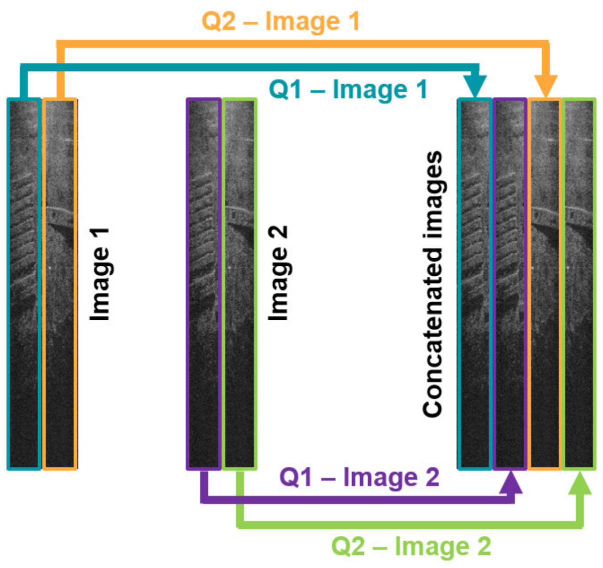


FIGURE 20. Each of two sonar images is split into two parts, representing two quadrants. The quadrants are re-ordered as shown and are converted into vectors to be used as the input of the DL network.

recovered trajectory presents abrupt changes in the same part of the trajectory. The better performance using 256 levels was expected, given that the motion estimation with 256 levels is slightly more accurate than the case of 512 as shown in Figure 17.

It can be seen that, even with this simple trajectory recovery procedure described above, the estimated trajectory is close to the ground truth. More sophisticated recovery techniques based on regularized spline interpolation and/or Kalman filtering could improve the trajectory estimation. This will be a subject of our future research.

E. TRAJECTORY RECONSTRUCTION AND MOSAIC BUILDING USING REAL DATA

The accuracy of the vector technique is evaluated using real data. Therefore a data set obtained by a real sonar sensor is used. The selected data set has a total of 4464 images. It shows several passes along a ship's hull during its inspection. From the full data set, 520 images are extracted. These images correspond to one single pass from end to end of the ship. The sonar employed for this inspection is the DIDSON 300 [45] mounted on a Bluefin Robotics Hovering Autonomous Underwater Vehicle [42]. The images from the sonar have a size of 512×96 pixels, and intensity values in the range of 0 to 255. The FoV of this sonar in aperture and elevation angles is $29^\circ \times 14^\circ$, respectively. The sonar generates 21 images per second (21 fps).

A data set generated from a simulated environment that resembles a ship's hull is used for training [29]. The data set contains 76000 training samples. The images have the same aperture and elevation angles as images obtained by the DIDSON sonar. The DIDSON sonar is an FLS, therefore the vertical axis on the image corresponds to the forward motion of the sonar and the horizontal axis corresponds to the sideways motion. To train the network, noise is added to the images given that the real images are noisy. The noise parameters were adjusted to give the best estimates with the real data set and they are as follows: For the Gaussian distribution the mean and standard deviation are set to 4% and 2% of the maximum pixel value, respectively. And for the Rayleigh distribution, the scale parameter is set to 4% of the maximum pixel value.

The images obtained from the DIDSON sonar cannot be taken from 4 quadrants and the aperture and elevation angles are different from the data sets used in previous subsections. Therefore, the images are modified by separating them in azimuth in two halves, where each half represents a quadrant. Using pairs of consecutive images, the input of the PoseNetVec is the concatenation of 4 vectors of size 512. The first 2 vectors correspond to the left columns of the first and second images and the other 2 vectors correspond to the right columns of the images as shown in Figure 20.

In Figure 21, the full estimated trajectory and orientation of the platform are displayed with red line and blue arrows, respectively. According to the reconstructed trajectory, a mosaic is built by merging the 520 images in the data set. When the images are combined, the pixel intensities are

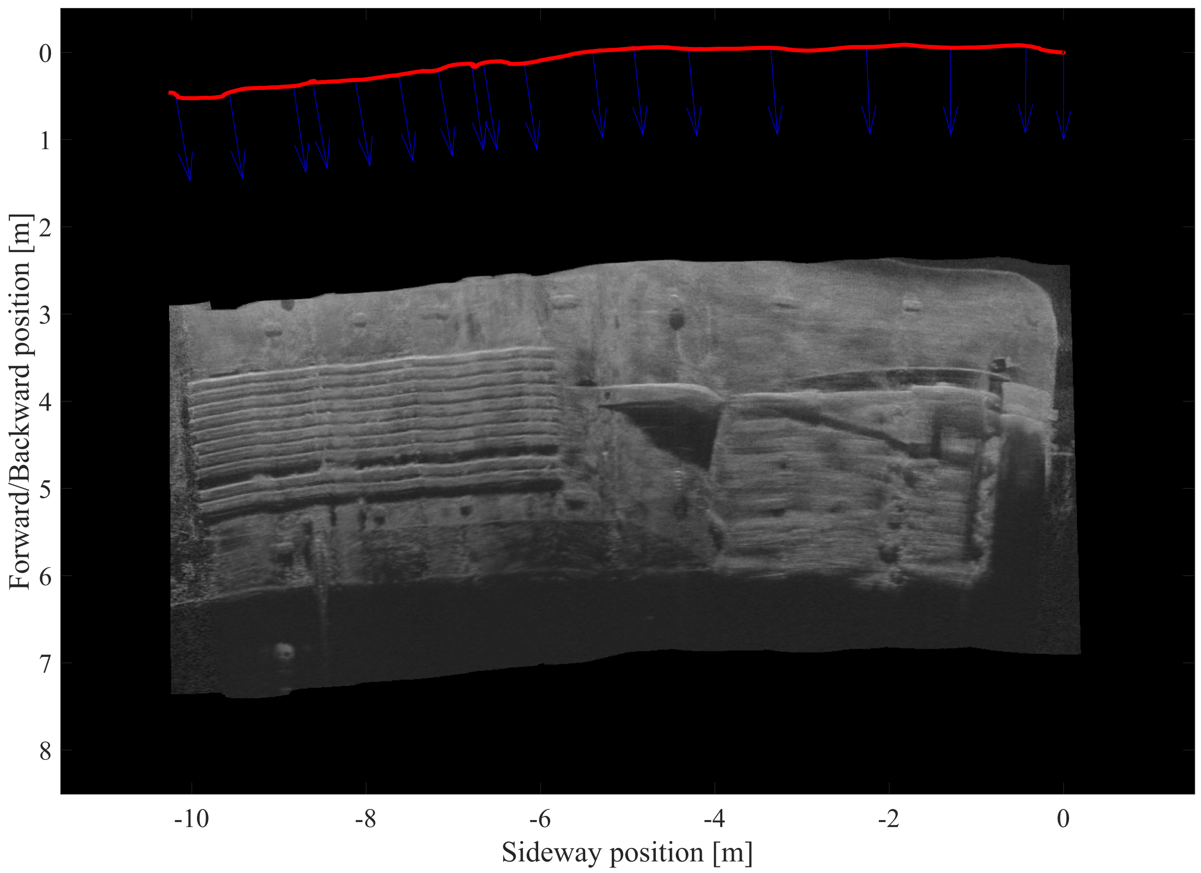


FIGURE 21. Reconstructed platform trajectory (red line), platform orientation (blue arrows) and mosaic built using 520 sonar images of the ship's hull real data. From the mosaic, multiple parts of the ship's hull are clearly recognizable, like the sacrificial anodes, the propeller and the keel.

averaged in the overlapping images. It can be seen that during the ship's hull inspection, the sonar mostly moves in the sideways direction with small motions in the forward/backward direction and rotation. This can be seen in the estimated trajectory, while the small rotation can be seen pointing forward most of the time.

The estimated trajectory and mosaic look accurate compared to similar pictures in the other works [25], [28]. Furthermore, the time to obtain the motion estimates using the computer described in subsection IV-A was about 12 ms per estimate. This computing time is considerably smaller than for the methods in [28] and [29], whose computing times for each pair of images are 25.69 s and 56.9 ms, respectively. Moreover, the computing time is much smaller than the time that it takes for the DIDSON sonar to generate an image with a frame rate of 21 fps (an image every 48.62 ms). This shows that the vector technique is suitable for accurate motion estimation in real-time.

V. CONCLUSION

Two main contributions for underwater motion estimation are presented in this paper. The first contribution is a technique that consists in converting sonar images into vectors by adding up the pixels in each row. The use of vectors reduces the size of the DL networks that are used to estimate the motion. The second contribution is the division of the beams in the FoV of a simulated sonar into four groups (quadrants). An image is generated from each group. This is combined with the vector technique by converting the images into vectors and grouped together as the input of the DL network. The network is trained using simulated data. The proposed vector technique using the four images shows better accuracy and faster computation times compared to other similar methods from the literature.

Validation of the vector technique with simulated and real data is presented. For the case of real data, sonar images from an FLS are used by adapting the quadrants with the vector technique to the features of the images. The application on real data shows good results and the potential to perform low-cost and fast motion estimation in real time.

These results suggest that it might be possible to design a motion estimation system with a few hydrophones possessing a low space directivity. In our future work, we are going to investigate such designs and compare them with known techniques.

ACKNOWLEDGMENT

The authors would like to thank Prof. Y. Petillot, School of EPS, Heriot-Watt University, and Dr. N. Hurtós for providing the ship's hull data set.

REFERENCES

[1] M. M. Dos Santos, G. G. De Giacomo, P. L. J. Drews, and S. S. C. Botelho, "Underwater sonar and aerial images data fusion for robot localization," in *Proc. 19th Int. Conf. Adv. Robot. (ICAR)*, Belo Horizonte, Brazil, Dec. 2019, pp. 578–583.

[2] A. Grządziel, "Results from developments in the use of a scanning sonar to support diving operations from a rescue ship," *Remote Sens.*, vol. 12, no. 4, p. 693, Feb. 2020.

[3] M. Specht, A. Stateczny, C. Specht, S. Widźgowski, O. Lewicka, and M. Wiśniewska, "Concept of an innovative autonomous unmanned system for bathymetric monitoring of shallow waterbodies (INNOBAT System)," *Energies*, vol. 14, no. 17, p. 5370, Aug. 2021.

[4] N. Modalavalasa, "An efficient implementation of tracking using Kalman filter for underwater robot application," *Int. J. Comput. Sci., Eng. Inf. Technol.*, vol. 2, no. 2, pp. 67–78, Apr. 2012.

[5] K. Fang, S. Zhang, and J. Liu, "Principle and error analysis of Doppler acoustic omni-directional beacon," in *Geo-Informatics in Resource Management and Sustainable Ecosystem*. Berlin, Germany: Springer, 2013, pp. 384–392.

[6] L. Whitcomb, D. R. Yoerger, H. Singh, and J. Howland, "Advances in underwater robot vehicles for deep ocean exploration: Navigation, control, and survey operations," in *Robotics Research*, J. M. Hollerbach and D. E. Koditschek, Eds. London, U.K.: Springer, 2000, pp. 439–448.

[7] M. B. Loc, H.-S. Choi, J.-M. Seo, S.-H. Baek, and J.-Y. Kim, "Development and control of a new AUV platform," *Int. J. Control, Autom. Syst.*, vol. 12, no. 4, pp. 886–894, Aug. 2014.

[8] M. Salhaoui, J. C. Molina-Molina, A. Guerrero-González, M. Arioua, and F. J. Ortiz, "Autonomous underwater monitoring system for detecting life on the seabed by means of computer vision cloud services," *Remote Sens.*, vol. 12, no. 12, p. 1981, Jun. 2020.

[9] X. Yuan, J.-F. Martínez-Ortega, J. Fernández, and M. Eckert, "AEKF-SLAM: A new algorithm for robotic underwater navigation," *Sensors*, vol. 17, no. 5, p. 1174, May 2017.

[10] J. Li, M. Kaess, R. M. Eustice, and M. Johnson-Roberson, "Pose-graph SLAM using forward-looking sonar," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2330–2337, Jul. 2018.

[11] A. Grosch, C. Enneking, L. A. Greda, D. Tanajewski, G. Grunwald, and A. Ciećko, "Theoretical concept for a mobile underwater radio-navigation system using pseudolite buoys," *Remote Sens.*, vol. 12, no. 21, p. 3636, Nov. 2020.

[12] P. N. Mikhalevsky, B. J. Sperry, K. F. Woolfe, M. A. Dzieciuch, and P. F. Worcester, "Deep ocean long range underwater navigation," *J. Acoust. Soc. Amer.*, vol. 147, no. 4, pp. 2365–2382, Apr. 2020.

[13] A. Stateczny, W. Błaszczak-Bąk, A. Sobieraj-Żłobińska, W. Motyl, and M. Wisniewska, "Methodology for processing of 3D multibeam sonar big data for comparative navigation," *Remote Sens.*, vol. 11, no. 19, p. 2245, Sep. 2019.

[14] R. Panish and M. Taylor, "Achieving high navigation accuracy using inertial navigation systems in autonomous underwater vehicles," in *Proc. OCEANS*, Santander, Spain, Jun. 2011, pp. 1–7.

[15] Z. Song and K. Mohseni, "Long-term inertial navigation aided by dynamics of flow field features," *IEEE J. Ocean. Eng.*, vol. 43, no. 4, pp. 940–954, Oct. 2018.

[16] D. Li, J. Xu, H. He, and M. Wu, "An underwater integrated navigation algorithm to deal with DVL malfunctions based on deep learning," *IEEE Access*, vol. 9, pp. 82010–82020, 2021.

[17] S. Wirth, P. L. Negre Carrasco, and G. O. Codina, "Visual odometry for autonomous underwater vehicles," in *Proc. MTS/IEEE OCEANS*, Bergen, Norway, Jun. 2013, pp. 1–6.

[18] F. Dubrovin, Y. Vaulin, A. Scherbatyuk, D. Scherbatyuk, and A. Rodionov, "Navigation for AUV, located in the shadow area of LBL, during the group operations," in *Proc. Global Oceans*, Singapore, Oct. 2020, pp. 1–6.

[19] S. Fan, C. Liu, B. Li, Y. Xu, and W. Xu, "AUV docking based on USBL navigation and vision guidance," *J. Mar. Sci. Technol.*, vol. 24, no. 3, pp. 673–685, Sep. 2019.

[20] W. Chen and R. Sun, "Optimal distance for moving long baseline positioning system with distance-dependent measurement noise," *Adv. Mech. Eng.*, vol. 10, no. 6, pp. 1–9, 2018.

[21] M. V. Jakuba, J. C. Kinsey, J. W. Partan, and S. E. Webster, "Feasibility of low-power one-way travel-time inverted ultra-short baseline navigation," in *Proc. OCEANS*, Washington, DC, USA, Oct. 2015, pp. 1–10.

[22] V. A. Bobkov, A. P. Kudryashov, S. V. Mel'man, and A. F. Shcherbatyuk, "Autonomous underwater navigation with 3D environment modeling using stereo images," *Gyroscopy Navigat.*, vol. 9, no. 1, pp. 67–75, Jan. 2018.

[23] P. Dabove, V. Di Pietra, and M. Piras, "Monocular visual odometry with unmanned underwater vehicle using low cost sensors," in *Proc. IEEE/ION Position, Location Navigat. Symp. (PLANS)*, Portland, OR, USA, Apr. 2020, pp. 810–816.

[24] F. Ferreira, D. Machado, G. Ferri, S. Dugelay, and J. Potter, "Underwater optical and acoustic imaging: A time for fusion? A brief overview of the state-of-the-art," in *Proc. OCEANS*, Monterey, CA, USA, Sep. 2016, pp. 1–6.

[25] N. Hurtós, D. Ribas, X. Cufí, Y. Petillot, and J. Salvi, "Fourier-based registration for robust forward-looking sonar mosaicing in low-visibility underwater environments," *J. Field Robot.*, vol. 32, no. 1, pp. 123–151, Jan. 2015.

[26] W. Kazimierski and G. Zaniewicz, "Determination of process noise for underwater target tracking with forward looking sonar," *Remote Sens.*, vol. 13, no. 5, p. 1014, Mar. 2021.

[27] S. Negahdaripour, "On 3-D motion estimation from feature tracks in 2-D FS sonar video," *IEEE Trans. Robot.*, vol. 29, no. 4, pp. 1016–1030, Aug. 2013.

[28] B. T. Henson and Y. V. Zakharov, "Attitude-trajectory estimation for forward-looking multibeam sonar based on acoustic image registration," *IEEE J. Ocean. Eng.*, vol. 44, no. 3, pp. 753–766, Jul. 2019.

[29] J. E. Almanza-Medina, B. Henson, and Y. V. Zakharov, "Deep learning architectures for navigation using forward looking sonar images," *IEEE Access*, vol. 9, pp. 33880–33896, 2021.

[30] Y. Yang and G. Huang, "Acoustic-inertial underwater navigation," in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, Singapore, May 2017, pp. 4927–4933.

[31] N. Y. Ko, D. Jeong, and G. Song, "Navigation of a remotely operated underwater vehicle using IMU and DVL," in *Proc. 19th Int. Conf. Control, Autom. Syst. (ICCAS)*, Jeju, South Korea, Oct. 2019, pp. 1098–1100.

[32] P. A. Miller, J. A. Farrell, Y. Zhao, and V. Djapic, "Autonomous underwater vehicle navigation," *IEEE J. Ocean. Eng.*, vol. 35, no. 3, pp. 663–678, Jul. 2010.

[33] Y. Wang, X. Ma, J. Wang, and H. Wang, "Pseudo-3D vision-inertia based underwater self-localization for AUVs," *IEEE Trans. Veh. Technol.*, vol. 69, no. 7, pp. 7895–7907, Jul. 2020.

[34] Z. Xu, M. Haroutunian, A. J. Murphy, J. Neasham, and R. Norman, "An integrated visual odometry system for underwater vehicles," *IEEE J. Ocean. Eng.*, vol. 46, no. 3, pp. 848–863, Jul. 2021.

[35] M. Franchi, A. Ridolfi, and B. Allotta, "Underwater navigation with 2D forward looking SONAR: An adaptive unscented Kalman filter-based strategy for AUVs," *J. Field Robot.*, vol. 38, no. 3, pp. 355–385, May 2021.

[36] A. Karmozdi, M. Hashemi, H. Salarieh, and A. Alasty, "Implementation of translational motion dynamics for INS data fusion in DVL outage in underwater navigation," *IEEE Sensors J.*, vol. 21, no. 5, pp. 6652–6659, Mar. 2021.

[37] Z. Yin and J. Shi, "GeoNet: Unsupervised learning of dense depth, optical flow and camera pose," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Salt Lake City, UT, USA, Jun. 2018, pp. 1983–1992.

[38] J. E. Almanza-Medina, B. T. Henson, and Y. V. Zakharov, "Imaging sonar simulator for assessment of image registration techniques," in *Proc. OCEANS*, Seattle, WA, USA, Oct. 2019, pp. 1–7.

[39] *Unity*. Accessed: Aug. 8, 2021. [Online]. Available: https://unity.com

[40] *MATLAB—Mathworks*. Accessed: Aug. 8, 2021. [Online]. Available: https://www.mathworks.com

[41] F. Schmitt, M. Mignotte, C. Collet, and P. Thourel, "Estimation of noise parameters on sonar images," *Proc. SPIE*, vol. 2823, pp. 2–13, Oct. 1996.

[42] (2015). *Hovering Autonomous Underwater Vehicle Specifications*. Bluefin Robotics Corporation. Accessed: Aug. 8, 2021. [Online]. Available: https://gdmissionsystems.com/-/media/General-Dynamics/Maritime-and-Strategic-Systems/Bluefin/PDF/Bluefin-HAUV-Datasheet.ashx

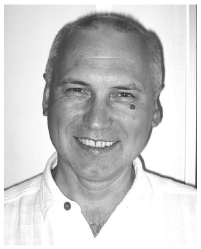
[43] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[44] S. Marschner and P. Shirley, *Fundamentals of Computer Graphics*. Boca Raton, FL, USA: CRC Press, 2018.

[45] *DIDSON 300 Standard Version Specifications*. Sound Metrics Corp. Accessed: Aug. 8, 2021. [Online]. Available: http://www.soundmetrics.com/products/DIDSON-Sonars/DIDSON-300m/DIDSON-300-Standard-Version-Specifications

JOSÉ E. ALMANZA-MEDINA (Graduate Student Member, IEEE) received the M.Sc. degree in electronics and telecommunications from the Ensenada Center for Scientific Research and Higher Education (CICESE), in 2015. He is currently pursuing the Ph.D. degree in electronic engineering with the Communication Research Group, Department of Electronic Engineering, University of York, U.K. From 2009 to 2013, he worked with Softtek IT Company, Mexico, as a Software Engineer and a Project Manager. His research interests include signal and image processing, and underwater acoustics.

BENJAMIN HENSON (Member, IEEE) received the M.Eng. degree in electronic engineering and the M.Sc. degree in natural computation from the University of York, U.K., in 2001 and 2011, respectively, and the Ph.D. degree in electronic engineering from the Communication Technologies Research Group, University of York, in 2018. From 2002 to 2008, he worked as an Engineer with Snell & Wilcox Ltd., on designing broadcast equipment. From 2008 to 2009, he worked at SRD Ltd., on imaging sonar designs. Then, from 2011 to 2013, he worked on laser measurement equipment for Renishaw plc. He is currently working as a Research Associate with the Department of Electronic Engineering, University of York. His research interests include signal and image processing, and acoustics.



YURIY V. ZAKHAROV (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical engineering from the Moscow Power Engineering Institute, Moscow, Russia, in 1977 and 1983, respectively. From 1977 to 1983, he was with the Special Design Agency, Moscow Power Engineering Institute. From 1983 to 1999, he was with the N. N. Andreev Acoustics Institute, Moscow. From 1994 to 1999, he was with Nortel as a DSP Group Leader. Since 1999, he has been with the Communications Research Group, University of York, U.K., where he is currently a Reader with the Department of Electronic Engineering. His research interests include signal processing, communications, and underwater acoustics.