Stixel-based Free Space Estimation for USVs using Stereo Camera and LiDAR

Johannes Robert Skarø[1] Trym Anthonsen Nygård[1] Rudolf Mester[2] Annette Stahl[1] Edmund Førland Brekke[1]

***Abstract*—Unmanned surface vehicles (USVs) require robust situational awareness to navigate safely in complex maritime environments. A critical element of this is to identify the free navigable space around the USV. Free water regions can be derived from water segmentation in the image. However, these segmented regions must be transformed into a bird's eye view (BEV) representation to be utilized effectively in motion planning. This paper proposes a novel approach to estimate free navigable space in a BEV format by integrating a stereo camera and light detection and ranging (LiDAR). The proposed method uses water segmentation to delineate the water surface and represents the closest obstacles in the USV line of sight using vertical planar rectangles known as Stixels. The depth of these Stixels is derived from LiDAR data, ensuring precise positioning in space. The effectiveness of the approach is demonstrated through experiments conducted on real-world data collected from the milliAmpere 2 (MA2) autonomous ferry prototype in Trondheim, Norway. Qualitative evaluations focusing on accuracy and temporal consistency confirm its ability to reliably detect free navigable areas in complex maritime environments.**

***Index Terms*—Situational awareness, stereo camera, LiDAR, maritime autonomy, fusion**

I. Introduction

In recent years, USVs have gained significant attention due to their potential to provide efficient, profitable, and safer operations at sea. A cornerstone of safe navigation for USVs is maintaining a precise and continuously updated understanding of their surroundings. For motion planning systems, it is essential to determine where the USV can navigate safely. Mapping free navigable space can be challenging in confined waterways, where multiple static and dynamic objects of varying sizes and structures are present. A common approach is to use camera-based obstacle detection, leveraging the rich visual information from images. However, detecting individual objects can be difficult in high-traffic areas due to occlusions and clutter, and it might be insufficient to accurately map the free space. Instead, free water regions can be detected directly from water segmentation, providing a visual clue of the boundaries of the free space.

In order to autonomously navigate, the free water regions detected need to be transformed into a common geographic frame, requiring depth information. Projective geometry transformations can be used as a reconstruction method, but provide inaccurate depth data at large ranges. To this end, this paper proposes a novel approach that uses both stereo camera and LiDAR measurements to reconstruct the boundary of free space in a BEV representation. Specifically, the contributions are: (1) a robust water segmentation pipeline that uses temporal filtering and ego-motion compensation to improve reliability. (2) A Stixel representation adapted to maritime environments, created using water segmentation and visual clues from the RGB and disparity image. (3) Accurate positioning of the Stixels using depth measurements from LiDAR data projected onto the camera plane. An example of the output of the method is illustrated in Figure 1.

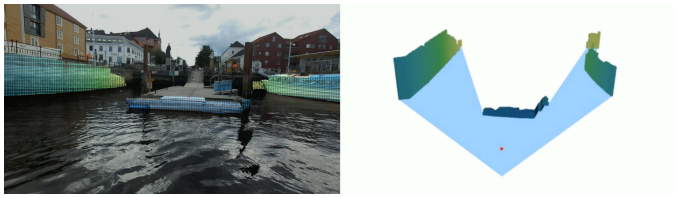


Fig. 1: Image of a dock with Stixels overlaid (left) and corresponding 3D view of the estimated free space (right).

II. Related Work

Several methods for water segmentation have been developed. One approach divides images into smaller subregions classified as water, partial water, or non-water using a convolutional neural network (CNN)-based model [1]. Another method uses horizon estimation from an inertial measurement unit (IMU) as a prior for CNN-based water segmentation [2]. A cross-model fusion technique that combines camera and LiDAR data was proposed by [3] to improve segmentation. [4] proposes a hybrid approach for robust water segmentation by combining Fast Segment Anything Model (FastSAM) [5], a recent image segmentation network, with a Random Sample Consensus (RANSAC)-based plane fitting method.

In the automotive domain, machine learning-based methods have been developed to estimate free space directly in a BEV format using camera data [6] and camera-LiDAR fusion [7], [8].

The occupancy grid map, first introduced by [9], can also provide detailed spatial information about the free space. The environment is discretized into a grid of cells, each assigned a probability of being occupied or not. Grid maps have been created for the roads in [10], which introduced a dynamic programming approach to compute the optimal boundary of the free space. More recent advances include dynamic occupancy grids developed for object detection, velocity estimates, and drivable area detection [11], [12]. However, despite their rich

[1]Department of Engineering Cybernetics, NTNU, Trondheim, Norway.
[2]Department of Computer Science, NTNU, Trondheim, Norway.

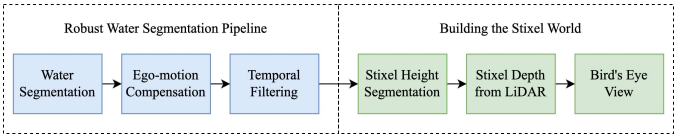


Fig. 2: Overview of the free space estimation method. The output is a BEV representation of the free space.

spatial information, occupancy grid maps require significant memory and processing power, making real-time implementations challenging. They require high resolution grids to be accurate, and scaling them to large areas leads to excessive computational overhead.

In contrast, the Stixel World provides an efficient mid-level scene representation by grouping pixels into thin rectangular segments defined by a base point, height and depth [13]. Initially introduced as a compact representation of first layer obstacles in traffic scenes [14], it was later extended to the dynamic Stixel World, where Stixels were tracked to capture the motion of dynamic objects [15]. The framework was further extended to model entire scenes using a multilayer Stixel representation [16]. Over time, several improvements have been made, including color integration [17] and semantic labeling [18], [19]. Stixel processing has been integrated into broader autonomous systems, leveraging its structured representation for instance segmentation [20] and deep learning-based free space segmentation in driving scenarios [21].

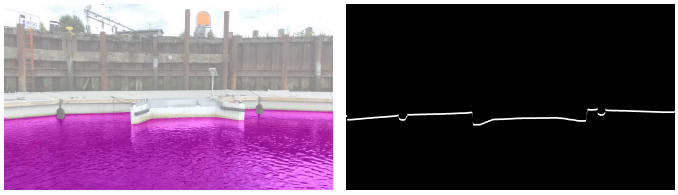
Despite its success in automotive applications, Stixel World has received little attention in the maritime domain. To the authors' knowledge, the only existing works in this area are [22] and [23], where Stixels are generated using stereo camera disparity information to identify free navigable areas in urban waterways. This paper builds on that approach by introducing a new Stixel creation pipeline that integrates disparity, RGB information, and LiDAR measurements to accurately position the Stixels in a BEV representation.

III. Overview of the Free Space Estimation Method

An overview of the free space estimation method is illustrated in Figure 2. The pipeline begins with water segmentation, described in Section IV-A, to delineate free water regions and identify the boundary between navigable areas and obstacles. Accurate water segmentation is crucial, as errors at this stage directly affect subsequent free space estimation.

To enhance the robustness of the water segmentation, temporal filtering is performed. This involves aligning recent water masks and performing a pixel-wise majority voting to predict whether a pixel corresponds to water or not. To align previous frames, the ego-motion of the USV is accounted for by warping the previous water masks to match the current frame. Temporal filtering helps mitigate sudden frame losses or single frame segmentation errors, improving overall performance. The details of these steps are described in Sections IV-B and IV-C.

Once an accurate water mask is obtained, the next step is to locate the boundary between the water regions and the first line



(a) (b)

Fig. 3: (a): Image of a dock from the MA2 dataset with FusedWSS segmentation result overlaid (pink). (b): The extracted free space boundary mask from the water mask.

of sight obstacles. This boundary is defined as the first pixel in each column that is not labeled as water when scanning upward from the bottom edge of the image. Even if areas further above in the water mask are labeled as water, they are not considered free navigable space in this context. Stixels are then generated to represent the closest obstacles in the scene, with details of this process provided in Section V-A.

After creating the Stixels, they are used to filter the LiDAR point cloud. The point cloud is first projected onto the image plane, and only the points that fall within the boundaries of the Stixels are kept. The depth information from these LiDAR points is then used to position the Stixels in 3D world coordinates as described in Section V-B. Finally, a 2D bird's eye view map is generated of the Stixel positions, providing a structured representation of the free navigable space.

IV. Robust Water Segmentation Pipeline

A. Water Segmentation

Several water segmentation methods exist, and any method that produces a binary water mask can be used. This paper uses Fused Water Surface Segmentation (FusedWSS) [4], which combines a RANSAC-based water plane segmentation method with the FastSAM instance segmentation model. A detailed explanation of the FusedWSS method is omitted, and the reader is referred to [4] for further details.

Using the computed water mask, the free space boundary is defined as the first non-water pixel in each column when scanning upward from the bottom of the image. Figure 3 demonstrates the FusedWSS method and the resulting free space boundary extraction.

B. Temporal Filtering

The water segmentation should be consistent and always yield reliable results. To achieve this, the temporal filtering pipeline maintains a history of the most recent masks, up to a specified frame limit K. Using these past masks, a prediction of the current water mask is generated. The stored masks allow for majority voting, where each pixel is evaluated based on its occurrence as water throughout the sequence of past frames. This voting mechanism helps reduce noise and segmentation failure by only classifying pixels as water if they appear as such in a sufficient number of recent frames, effectively filtering out sporadic or isolated loss of water detection.

Fig. 4: A sequence of water masks from obtained from the MODD2 dataset, warped into the current frame k, along with the predicted water mask for the current frame. Yaw rotation between frames has been compensated to preserve the shape of the boat.

Each water mask is represented as a binary matrix where 1 indicates a water pixel and 0 indicates non-water. The last K masks are summed element-wise to create an aggregated matrix. A pixel is predicted to be water if its value in the aggregated matrix exceeds the threshold $\lfloor 2K/3 \rfloor$ where $\lfloor \cdot \rfloor$ is the floor operator. The threshold requires that at least two-thirds of past frames label a pixel as water, balancing false positives with segmentation variability. Finally, the predicted water mask is merged with the current water mask through a pixel-wise logical OR operation, enhancing the overall robustness of the segmentation.

For a meaningful comparison of pixels between previous frames, the frames must be aligned and corrected for the USV ego-motion. Without this compensation, a lag effect occurs due to camera movement between frames.

C. Ego-Motion Compensation

Two images of the same planar surface, viewed from different angles, are related by homography. Here, the planar surface corresponds to the water plane. By computing the homography between the past and current frames, the past water segmentation mask is warped into the perspective of the current frame.

Due to ego-motion, the camera center shifts between frames. To warp points from the previous frame at time $k-1$ (denoted $c, k-1$) into the current frame c, k, the homogeneous transformation $\mathbf{T}_{c,k-1}^{c,k}$ must be computed.

The position $\mathbf{t}_{b,k}^{w}$ and orientation $\mathbf{R}_{b,k}^{w}$ of the USV's body frame b at time k, expressed in the world coordinate frame w, are obtained from the Inertial Navigation System (INS) and Global Navigation Satellite System (GNSS) on the USV. These are combined into a homogeneous transformation as

$$\mathbf{T}_{b,k}^{w} = \begin{bmatrix} \mathbf{R}_{b,k}^{w} & \mathbf{t}_{b,k}^{w} \\ \mathbf{0}_{1\times 3} & 1 \end{bmatrix}. \tag{1}$$

Additionally, the static extrinsic calibration provides the transformation between the body frame b and the camera frame c, denoted as $\mathbf{T}_{c}^{b}$. Using this, the relative transformation between consecutive camera poses is given by

$$\mathbf{T}_{c,k-1}^{c,k} = \left(\mathbf{T}_{b,k}^{w}\mathbf{T}_{c}^{b}\right)^{-1}\mathbf{T}_{b,k-1}^{w}\mathbf{T}_{c}^{b}\,. \tag{2}$$

From this transformation, the relative rotation $\mathbf{R}_{c,k-1}^{c,k}$ and translation $\mathbf{t}_{c,k-1}^{c,k}$ are extracted as the rotational and translational components of the 4×4 matrix.

With the relative camera motion known, the homography between the two views can be computed. Consider camera poses at time k and $k-1$, both observing a 3D point $\mathbf{P}_i$ lying on a plane π. In homogeneous pixel coordinates, the projection of $\mathbf{P}_i$ in the previous frame is given by $\mathbf{u}_i^{c,k-1} = \begin{bmatrix} u_i^{k-1} & v_i^{k-1} & 1 \end{bmatrix}^\top$. Its projection in the current frame is related by planar homography, up to scale, as

$$\mathbf{u}_i^{c,k} \sim \mathbf{K}\mathbf{H}_{c,k-1}^{c,k}\mathbf{K}^{-1}\mathbf{u}_i^{c,k-1}, \tag{3}$$

where $\mathbf{K}$ is the camera intrinsic matrix [24]. The homography matrix is defined as

$$\mathbf{H}_{c,k-1}^{c,k} = \mathbf{R}_{c,k-1}^{c,k} - \frac{\mathbf{t}_{c,k-1}^{c,k}\mathbf{n}^\top}{d}, \tag{4}$$

where $\mathbf{n}$ is the normal vector of the plane π, and d is the orthogonal distance from the current camera center to the plane, i.e., the camera's height above the water surface. Since the water surface is assumed to be flat and horizontal, $\mathbf{n} = \begin{bmatrix} 0 & -1 & 0 \end{bmatrix}^\top$ is used. The resulting homogeneous coordinates are converted to pixel positions by dividing each by its third component.

Although the derivation above considers the previous frame $k-1$, the same procedure applies to any earlier frame by computing the relative transformation to the current frame k. The resulting homography is applied to the pixels in the water masks from previous frames, producing warped versions aligned with the current frame. Finally, temporal filtering is performed as before, resulting in a smoother and more consistent water segmentation.

The effectiveness of ego-motion compensation and temporal filtering is tested using synchronized stereo camera and IMU data from the Multi-Modal Marine Obstacle Detection Dataset

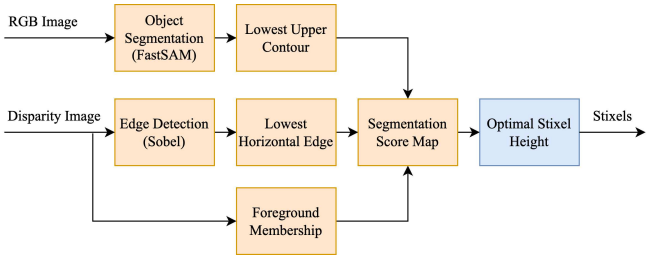


Fig. 5: The Stixel height segmentation pipeline. RGB and disparity images are processed separately to extract object contours, disparity edges and foreground membership score. These contribute to a score map used to determine the optimal Stixel heights.

2 (MODD2) [25]. Figure 4 shows the water masks of a sequence of images from the MODD2 dataset overlaid on the current RGB image. The sequence captures a rigid inflatable boat with a yaw rotation of the camera causing the position of the boat to shift within the image. The predicted water mask, derived from the previous masks, is shown on the right. As demonstrated, ego-motion compensation improves alignment, ensuring that past masks better align with the current frame. This is evident in the way the shape of the boat is preserved when the previous water masks are overlaid on the current image. However, warping cannot fill areas that were not in the previous frame's view, so newly uncovered regions near the image edges remain unlabeled (appearing as non-water by default).

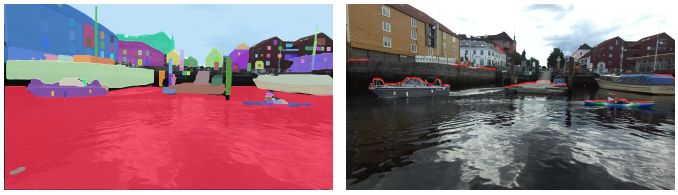
The temporal filtering pipeline performs well under small rotations, but loses accuracy with larger movements, leading to minor misalignment and lag in segmented water masks. Additionally, since it does not account for moving objects in the image, dynamic elements may introduce misalignment. However, since only a limited number of past frames are used, this effect remains minimal. Future work could improve the compensation technique to improve the robustness in such cases.

V. Building the Stixel World

Based on the RGB image, the disparity image and the free space boundary, a Stixel World representation of the scene is generated. The Stixel creation algorithm is inspired by the approach introduced in [14] and approximates first layer obstacles as Stixels. These objects are in contact with the water surface and are located directly above the free space boundary. To construct the Stixel representation, the image is vertically divided into N columns. Within each column, the goal is to determine the base row and top row of the Stixel. The base row is set as the median row position of the free space boundary within the current Stixel column.

A. Height Segmentation

The height of the Stixels is determined by identifying the optimal segmentation that separates the first layer objects and the background. This is achieved by computing a segmentation score map (SSM), where higher values indicate a stronger likelihood of being a valid segmentation boundary. The upper boundary of the objects is then obtained by applying dynamic programming to the SSM, ensuring an optimal segmentation path. Figure 5 shows the height segmentation pipeline.



(a) (b)

Fig. 6: (a): The FastSAM masks of the object segmentation. (b): The resulting upper contours of the lowest object masks (red) overlaying the RGB image.

The SSM is constructed from three separate inputs: (1) visual clues from the RGB image obtained using FastSAM, (2) edge information from the disparity image extracted by an edge detector, and (3) a foreground membership score that reflects the likelihood of each row within a Stixel column belonging to a foreground object. The SSM has the same number of rows as the RGB image and consists of N columns, corresponding to the number of Stixels. For each input source, an individual score map S_i is created. The final SSM is computed as a weighted sum of the three input score maps.

$$\mathrm{SSM}(n,v) = a_1 S_1(n,v) + a_2 S_2(n,v) + a_3 S_3(n,v), \quad (5)$$

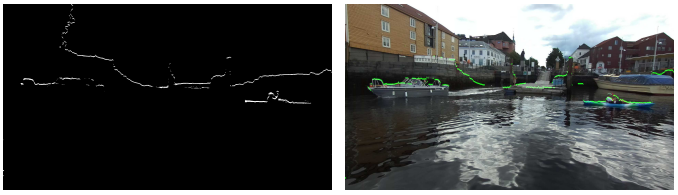
where $a_{1,2,3}$ are weights.

The first score map, denoted S_1, is derived from FastSAM [5], which generates segmentation masks for all objects in the scene, providing information on the height of objects. From these masks, only the upper contour lines are extracted. Using the free space boundary as a reference, only the lowest upper contour line above the waterline is selected, representing the upper boundary of objects that protrude from the water. All extracted contour lines are then combined into a single binary mask C. To compute the score map S_1, the mean value of each row within every Stixel in C is assigned to its corresponding element in S_1. For a given Stixel n, Stixel width w and row v, the score is computed as

$$S_1(n,v) = \mathrm{mean}\left(C[nw : (n+1)w, v]\right). \quad (6)$$

Here, $C[nw : (n+1)w, v]$ denotes slicing across image columns. Figure 6a shows the object segmentation by FastSAM, and Figure 6b illustrates the lowest upper contours overlaid on the RGB image.

The second score map, denoted S_2, is derived from horizontal edges detected in the disparity image using the Sobel operator. Edges in the disparity image serve as strong indicators of object boundaries, making them valuable for identifying transitions between surfaces. The upper boundaries are mostly horizontal, so the horizontal edges are extracted. The detected



(a) (b)

Fig. 7: (a): The horizontal edges of the disparity image obtained by the Sobel operator. (b): The resulting lowest edges above the waterline (green) overlaying the RGB image.

edges are then validated against a threshold to produce a binary mask. Using the free space boundary as a reference, only the lowest horizontal edges above the waterline are selected to form the mask H. This mask is integrated into S_2 similarly to the upper contours from FastSAM. Specifically, the mean value of each row within every Stixel in H is assigned to its corresponding element in S_2.

$$S_2(n, v) = \text{mean}\left(H[nw : (n+1)w, v]\right). \tag{7}$$

Figure 7a illustrates the horizontal edges detected in the disparity image, and Figure 7b shows the lowest edges above the waterline overlaid in the RGB image.

The third score map, denoted S_3, is a foreground membership score calculated from the disparity image. Each row within a Stixel column contributes to determining whether it belongs to the foreground object. Starting from the base row and moving upward, the median disparity value of each row is appended to a list. As new disparity values are added, the algorithm continuously computes the variance of all the values in the list. For each row v within a given Stixel n, the foreground membership is defined as

$$S_3(n, v) = 2^{(1-k\sigma^2_{n,v})} - 1, \tag{8}$$

where $\sigma^2_{n,v}$ is the variance of the list that contains all the median disparity values from the previous rows, and k is a tunable parameter. This approach is designed to detect sudden changes in disparity robustly. Under the assumption that objects consist of planar surfaces with approximately constant depth, a significant change in disparity suggests a transition to a different object. Such jumps in disparity increase the variance of the list, reducing the membership score accordingly. The parameter k controls the scaling of the variance $\sigma^2_{n,v}$, which effectively determines the sensitivity of the membership score to changes in the disparity. In these experiments, $k = 2$ was empirically found to perform well.

The foreground membership score has the effect of highlighting foreground objects in the scene. If a FastSAM contour line differs in height from a disparity edge line, the line falling within the foreground region will have a higher segmentation score and will be selected as the Stixel height. Furthermore, segmentation scores for rows without valid disparity data are set to zero, preventing unreliable depth measurements from influencing the computed Stixel heights. The weighting of the inputs to the SSM used in this paper is $a_1 = 2, a_2 = 1, a_3 = 1$, placing more emphasis on the FastSAM contour lines, which tend to be slightly more reliable and less noisy than the disparity edge lines.

Based on the SSM, the optimal height is computed. Following the approach in [14], a graph $G(V, E)$ is constructed. V is the set of vertices, with each vertex corresponding to an element on the SSM. E is the set of edges that connect every vertex in a column to every vertex in the following column. The objective is to find an optimal path through the graph that minimizes a cost function using dynamic programming. The total cost to minimize consists of two terms. A negative version of the SSM, ensuring that higher segmentation scores correspond to lower costs, and a smoothness term that penalizes abrupt variations in vertical position between adjacent columns. The cost of an edge connecting two vertices $V_{n,v0}$ and $V_{n+1,v1}$ is given by

$$c_{n,v_0,v_1} = S(n, v_0, v_1) - \text{SSM}(n, v_0), \tag{9}$$

where $\text{SSM}(n, v_0)$ is the segmentation score map at position (n, v_0). $S(n, v_0, v_1)$ is the smoothness term, defined as

$$S(n, v_0, v_1) = C_s \left|v_0 - v_1\right| \cdot \max\left(0, 1 - \frac{|z_n - z_{n+1}|}{N_z}\right), \tag{10}$$

where C_s is a parameter that scales the difference between rows v_0 and v_1. z_n and z_{n+1} represent depth values, obtained by converting disparity values to depth using the median disparity along the free space boundary within a Stixel. The last term relaxes the smoothness constraint when the depth difference between adjacent Stixels is larger than N_z. This ensures that Stixels with small depth variations are penalized more heavily. Based on empirical tuning, the parameters were set to $N_z = 5$ and $C_s = 2$.

Additionally, to maintain consistent obstacle representation, any computed Stixel height falling below a predefined minimum is adjusted to that minimum height. In these experiments, this minimum height is set to 20 pixels.

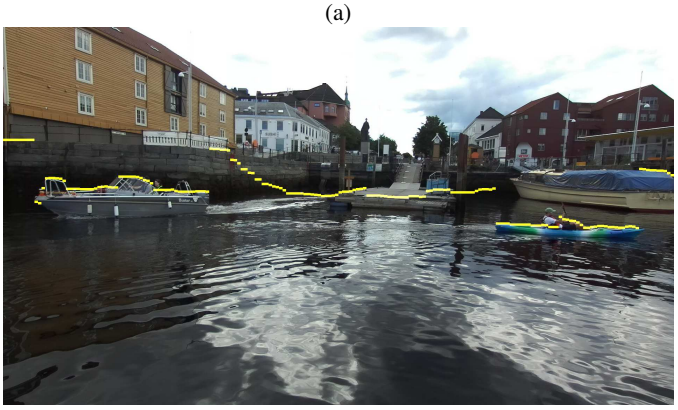
The results of the Stixel height segmentation are shown in Figure 8, where Figure 8a shows the segmentation score map and Figure 8b shows the resulting Stixel heights from the dynamic programming step. Notably, the Stixel heights to the left of the dock appear too small due to invalid disparity data in that area, causing the segmentation scores to be set to zero.

B. Filtering LiDAR Point Cloud with Stixels

After obstacles are represented as Stixels, LiDAR data are integrated to provide precise depth measurements. The process begins by projecting the LiDAR point cloud onto the image plane, expressing each point in pixel coordinates.

First, each LiDAR point $\mathbf{p}^l$ in homogeneous coordinates is transformed from the LiDAR frame l into the camera frame c using extrinsic calibration.

(a)



(b)

Fig. 8: (a): The SSM of the same frame as in Figures 6 and 7, resized to match the width of the RGB image for better illustration. (b): The resulting Stixel height segmentation (yellow).

$$\mathbf{p}^c = \mathbf{T}_l^c \, \mathbf{p}^l \, , \tag{11}$$

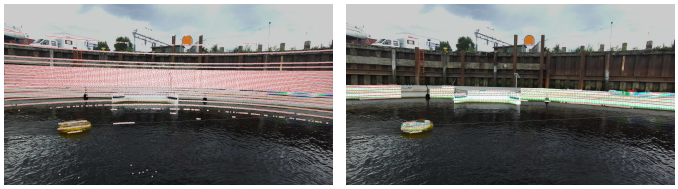
where $\mathbf{T}_l^c$ is the rigid body transformation from the LiDAR to the camera frame. Second, the resulting 3D point in the camera frame $\mathbf{p}^c$ is projected into 2D pixel coordinates $\mathbf{u}$ using the camera intrinsic matrix $\mathbf{K}$.

$$\mathbf{u} \sim \mathbf{K}\mathbf{p}^c, \tag{12}$$

where pixel coordinates are obtained by normalizing the third component.

The LiDAR points are then filtered by identifying those that fall within the boundaries of the Stixels. This results in a refined point cloud, where each Stixel contains a specific set of points. By doing so, only the relevant LiDAR points are extracted, those that correspond to the closest obstacles in the scene. An example is shown in Figure 9, where noisy water returns are effectively filtered out.

If multiple LiDAR measurements fall within a single Stixel, the median depth measurement can provide a robust estimate by filtering outliers. However, the assumption that the object is a vertical planar surface with constant depth is sometimes inaccurate. A better alternative could be to use the LiDAR measurement with the smallest depth to identify the closest part of the object. An even more refined approach is to use the 10th percentile of the depth values, which helps eliminate outliers while still detecting the closest part of the object.



(a) (b)

Fig. 9: (a:) LiDAR point cloud (red) projected into the image. (b): Filtered points retained by the Stixels (green).

If no LiDAR measurements fall within a single Stixel, then a LiDAR measurement from the previous LiDAR scan is used. If two successive scans lack hits, depth is taken from the nearest Stixel with a LiDAR measurement.

VI. Experimental Platform and Data

The milliAmpere 2 dataset was collected using the autonomous ferry prototype MA2 at Ravnkloa in Trondheim, Norway [26]. It consists of 12 episodes that capture realistic and varied urban harbor navigation scenarios, including interactions with static and dynamic targets, such as a kayak, an inflatable tube, and a speedboat. MA2 is equipped with mono and stereo cameras, LiDAR, INS, and GNSS. In these experiments, only data from the ZED 2 stereo camera and the LiDAR mounted on the aft port side are used.

The ZED 2 stereo camera captures video at a resolution of 1920 $\times$ 1080 pixels with a frame rate of 15 frames per second. It has a baseline of 120 mm, a field of view of 110° horizontally and 70° vertically, and a depth range of up to 20 m where accuracy decreases beyond this distance. The LiDAR is an Ouster OS1-32. It features 32 channels, a 360° horizontal field of view, and a 45° vertical field of view. It has a maximum depth range of 120 m, provides 655,360 points per second, and operates with a data rate of 66 Mbits^{-1} [27].

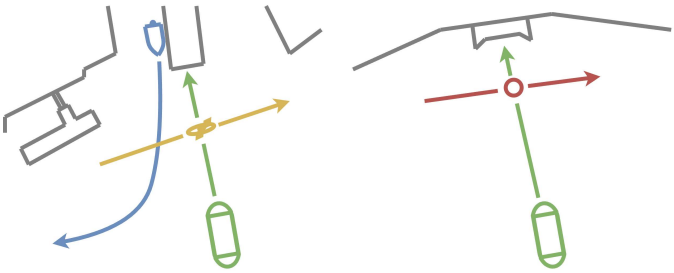
Figure 10 visualizes the two test episodes used in these experiments, named episode 1 and episode 2.

VII. Evaluation

To evaluate the accuracy of the free space estimation method, the results are visualized in both image and spatial domains. Stixels are overlaid on RGB images to assess their alignment with obstacles, and 3D scene visualizations are provided to give insights into the depth accuracy. Finally, a BEV representation is used to qualitatively assess the global accuracy and consistency over time of the estimated free space.

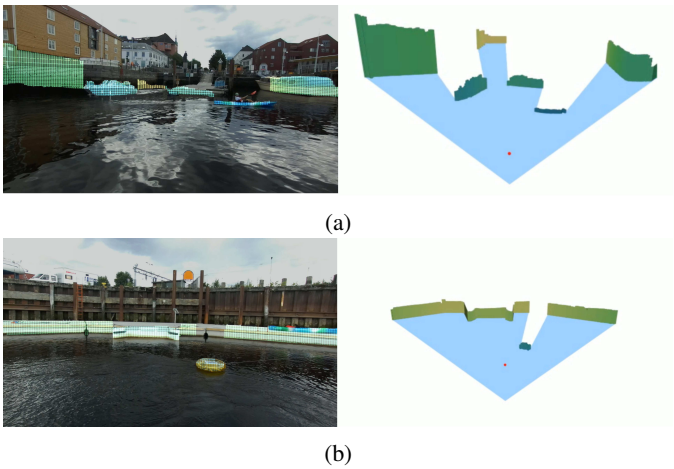
A. Stixel Alignment on Obstacles

Figure 11 shows the Stixel representation for selected frames from episodes 1 and 2. The results indicate that the Stixels align well with real-world objects, effectively representing first layer obstacles encountered by the USV. However,



(a) (b)

Fig. 10: Visualizations of MA2 episodes. (a) Episode 1: The ferry (green) is docking while a kayak (yellow) and a speedboat (blue) move in the scene. (b) Episode 2: The ferry (green) approaches the dock while an inflatable ring (red) crosses in front. The figure is based on illustrations in [4].



(a) (b)

Fig. 11: RGB images from episodes 1 and 2 overlaid with Stixels and filtered LiDAR points, alongside corresponding 3D visualizations. Free space is shown in blue, and obstacles are color-coded by depth. The red dot represents the center of the camera frame.

in some cases, as in Figure 11a, the Stixels to the right of the speedboat appear too short due to poor disparity data in that region.

B. Global Accuracy

The global accuracy is assessed using a BEV representation in world coordinates, as shown in Figure 12. The estimated free space is overlaid on shoreline data, with the ego vessel and midpoints of the Stixels represented as dots. The Stixels are labeled as boats in red according to YOLOv11 [29] object segmentation, while all other Stixels are assumed to be static objects, shown in dark blue. Overall, the estimated free space aligns well with open water areas, though some Stixels extend onto land and deviate from the shoreline contours. Additionally, the depth of the Stixels to the left of the dock appears incorrect due to the absence of LiDAR hits on these Stixels. This results in neighboring Stixel depth values being used instead.

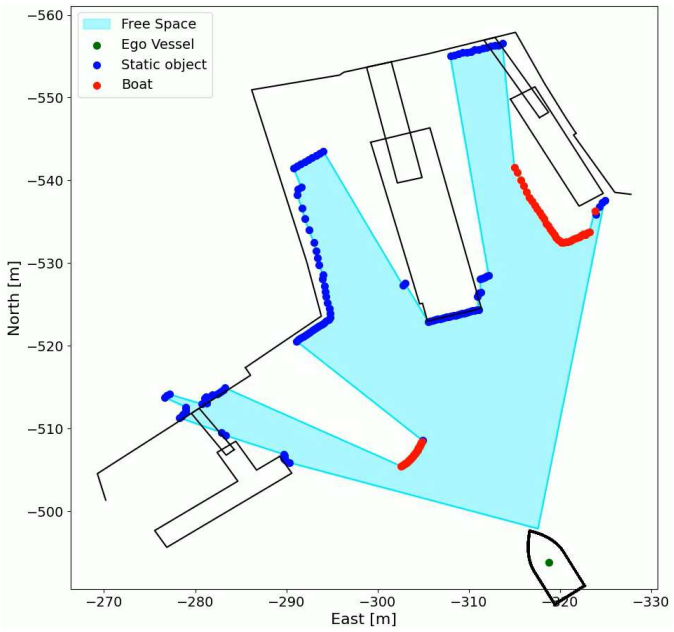


Fig. 12: BEV of the free space (light blue) for a frame in episode 1. Dots represent Stixel midpoints, labeled as boats (red) based on YOLOv11 or as static objects (dark blue). Map data: N5 Kartdata from Norwegian Mapping Authority [28].

C. Consistency in Depth over Time

The consistency of Stixel depth over time is evaluated by plotting Stixels from 200 consecutive frames in a BEV representation, as shown in Figure 13. Stixel positions are transformed to the latest camera frame using pose estimates from the INS and GNSS. The selected frames are from episode 2, where a floating tube is dragged from left to right, forming the long cluster closest to the camera. Stixels corresponding to static structures, such as the dock and pier, remain stable over time as they consistently overlap. Some noisy Stixels are also present. A slight drift is observed on the right side, indicated by a wider band of Stixels due to variations in depth.

VIII. Conclusion

This paper presented a novel method for estimating free navigable space in a BEV representation by fusing stereo camera and LiDAR data. The pipeline uses water segmentation to identify free water regions in RGB images and temporal filtering to make the water masks more consistent over time. The closest obstacles in the USV line of sight are represented as Stixels using clues from FastSAM object masks, edges in the disparity map, and a foreground membership score. The Stixels are positioned in space using precise depth information from LiDAR measurements projected onto the image plane. A BEV representation is then generated from the Stixels, suitable for motion planning.

Experimental results show that the method effectively aligns Stixels with obstacles and estimates free space over time. Although limitations were noted in cases with insufficient LiDAR data, the method successfully transforms segmented

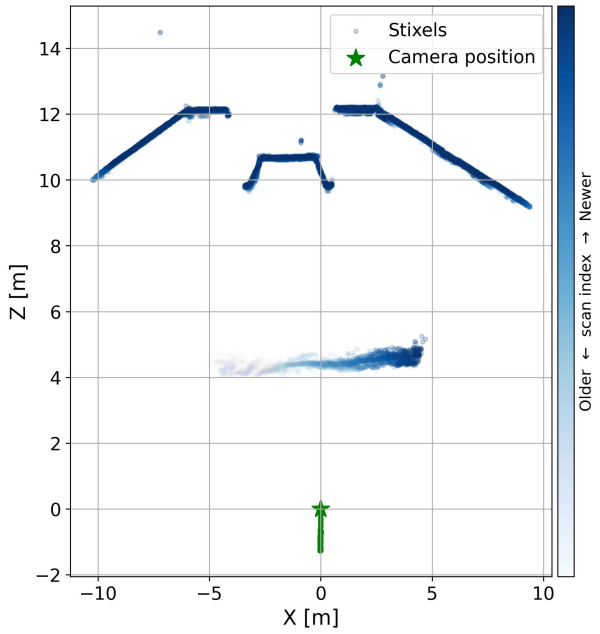


Fig. 13: The XZ-positions of Stixels from 200 consecutive frames, plotted in the latest camera frame. Newer Stixels appear darker and more opaque, while older ones gradually fade. The camera's position is marked by a star symbol, and its past trajectory is shown in green

water regions in the image into actionable spatial information, facilitating motion planning. Future work will focus on improving the spatial consistency of Stixels and robustly handling missing LiDAR data.

References

[1] M. K. Plenge-Feidenhans and M. Blanke, "Open water detection for autonomous in-harbor navigation using a classification network," *IFAC-PapersOnLine*, vol. 54, no. 16, pp. 30–36, 2021.

[2] B. Bovcon and M. Kristan, "Wasr: A water segmentation and refinement maritime obstacle detection network," *IEEE Transactions on Cybernetics*, vol. 52, no. 12, pp. 12 661–12 674, 2022.

[3] J. Gao, J. Zhang, C. Liu, X. Li, and Y. Peng, "Camera-lidar cross-modality fusion water segmentation for unmanned surface vehicles," *Journal of Marine Science and Engineering*, vol. 10, no. 6, p. 744, 2022.

[4] J. T. Grini, R. Mester, T. A. Nygård, N. Dalhaug, E. F. Brekke, and A. Stahl, "Fusedwss: Water surface segmentation fusing machine learning and geometric cues," in *2024 27th International Conference on Information Fusion (FUSION)*, 2024, pp. 1–8.

[5] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang, "Fast segment anything," *arXiv preprint arXiv:2306.12156*, 2023.

[6] T. Scheck, A. Mallandur, C. Wiede, and G. Hirtz, "Where to drive: free space detection with one fisheye camera," in *Twelfth International Conference on Machine Vision (ICMV 2019)*, vol. 11433. SPIE, 2020, pp. 777–786.

[7] A. V. Gideon Maillette de Buy Wenniger, Tijn Schmits, "Identifying free space in a robot bird-eye view," in *Proceedings of the 4th European Conference on Mobile Robots, ECMR'09, September 23-25, 2009, Mlini/Dubrovnik, Croatia*, I. Petrovic and A. J. Lilienthal, Eds. KoREMA, 2009, pp. 13–18.

[8] B. Yu, D. Lee, J.-S. Lee, and S.-C. Kee, "Free space detection using camera-lidar fusion in a bird's eye view plane," *Sensors*, vol. 21, no. 22, p. 7623, 2021.

[9] A. Elfes, "Using occupancy grids for mobile robot perception and navigation," *Computer*, vol. 22, no. 6, pp. 46–57, 1989.

[10] H. Badino, U. Franke, and R. Mester, "Free space computation using stochastic occupancy grids and dynamic programming," in *Workshop on Dynamical Vision, ICCV, Rio de Janeiro, Brazil*, vol. 20. Citeseer, 2007, p. 73.

[11] M. Schreiber, V. Belagiannis, C. Gläser, and K. Dietmayer, "A multi-task recurrent neural network for end-to-end dynamic occupancy grid mapping," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 315–322.

[12] H. Jang, T. Kim, K. Ahn, S. Jeon, and Y. Kang, "Dynamic occupancy grid map with semantic information using deep learning-based bevfusion method with camera and lidar fusion," *Sensors*, vol. 24, no. 9, 2024.

[13] S. Gehrig and U. Franke, "Stereovision for adas," in *Handbook of Driver Assistance Systems*, H. Winner, S. Hakuli, F. Lotz, and C. Singer, Eds. Springer International Publishing, 2014, ch. 21, pp. 495–524.

[14] H. Badino, U. Franke, and D. Pfeiffer, "The stixel world-a compact medium level representation of the 3d-world," in *Pattern Recognition: 31st DAGM Symposium, Jena, Germany, September 9-11, 2009. Proceedings 31*. Springer, 2009, pp. 51–60.

[15] D. Pfeiffer and U. Franke, "Efficient representation of traffic scenes by means of dynamic stixels," in *2010 IEEE Intelligent Vehicles Symposium*. IEEE, 2010, pp. 217–224.

[16] ——, "Towards a global optimal multi-layer stixel representation of dense 3d data." in *BMVC*, 2011, pp. 51.1–51.11.

[17] W. P. Sanberg, G. Dubbelman, and P. H. de With, "Extending the stixel world with online self-supervised color modeling for road-versus-obstacle segmentation," in *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, 2014, pp. 1400–1407.

[18] L. Schneider, M. Cordts, T. Rehfeld, D. Pfeiffer, M. Enzweiler, U. Franke, M. Pollefeys, and S. Roth, "Semantic stixels: Depth is not enough," in *2016 IEEE Intelligent Vehicles Symposium (IV)*, 2016, pp. 110–117.

[19] F. Piewak, P. Pinggera, M. Enzweiler, D. Pfeiffer, and M. Zöllner, "Improved semantic stixels via multimodal sensor fusion," in *German Conference on Pattern Recognition*. Springer, 2018, pp. 447–458.

[20] T. Hehn, J. Kooij, and D. Gavrila, "Fast and compact image segmentation using instance stixels," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 1, pp. 45–56, 2021.

[21] W. P. Sanberg, G. Dubbelman, and P. H. de With, "Free-space detection with self-supervised and online trained fully convolutional networks," *arXiv preprint arXiv:1604.02316*, 2016.

[22] T. A. Nygård, N. Dalhaug, R. Mester, E. Brekke, and A. Stahl, "Stereo camera-based free space estimation for docking in urban waters," *Modeling, Identification and Control*, vol. 45, no. 2, pp. 51–63, 2024.

[23] T. A. Nygård, E. F. Brekke, R. Mester, and A. Stahl, "Dynamic scene representation for docking in urban waters using a stereo camera," *Journal of Physics: Conference Series*, vol. 2867, p. 012029, 2024, published under licence by IOP Publishing Ltd.

[24] R. Hartley and A. Zisserman, *Multiple view geometry in computer vision*. Cambridge University Press, 2003.

[25] B. Bovcon, J. Muhovič, J. Perš, and M. Kristan, "Stereo obstacle detection for unmanned surface vehicles by IMU-assisted semantic segmentation," *Robotics and Autonomous Systems*, vol. 104, pp. 1–13, 2018.

[26] E. F. Brekke, E. Eide, B.-O. H. Eriksen, E. F. Wilthil, M. Breivik, E. Skjellaug, Ø. K. Helgesen, A. M. Lekkas, A. B. Martinsen, E. H. Thyri *et al.*, "milliampere: an autonomous ferry prototype," in *Journal of Physics: Conference Series*, vol. 2311, no. 1. IOP Publishing, 2022, p. 012029.

[27] Ouster, "Os1 lidar sensor datasheet," 2025, accessed: 2025-05-30. [Online]. Available: https://data.ouster.io/downloads/datasheets/datasheet-revd-v2p0-os1.pdf

[28] Norwegian Mapping Authority, "N5 Kartdata," https://kartkatalog.geonorge.no/metadata/n5-kartdata/6bb353c3-2b21-42fe-b296-31e60f64f95d, 2025, accessed: 2025-05-30 via NTNU through Norge Digitalt collaboration.

[29] G. Jocher and J. Qiu, "Ultralytics yolo11," 2024, accessed: 2025-05-18. [Online]. Available: https://github.com/ultralytics/ultralytics