GNSS-Independent Maritime Navigation Using Monocular Camera Images with Digital Elevation Map

1st Maxime Roedelé
Department of Engineering Cybernetics,
Norwegian University of Science and Technology,
Trondheim, Norway
maxime.roedele@gmail.com

2nd Kjetil Vasstein
Department of Engineering Cybernetics,
Norwegian University of Science and Technology,
Trondheim, Norway
0000-0003-4993-3146

3rd Tor Arne Johansen
Department of Engineering Cybernetics,
Norwegian University of Science and Technology,
Trondheim, Norway
tor.arne.johansen@ntnu.no

***Abstract*—The threat of GNSS service disruption implies the need for GNSS-independent navigation solutions. We introduce a framework that estimates the position of a maritime vessel in a littoral zone through monocular camera images and existing knowledge of the vessel's environment. A set of algorithms extract and match image features from a camera aboard the vessel with synthetic images rendered from a digital elevation model (DEM). The image features, together with the knowledge of the vessel's intended position, are used to infer a 3-dimensional feature inside the DEM. A motion-only bundle adjustment optimization problem is then posed, seeking to estimate the actual position of the vessel by minimizing the reprojection error of the DEM feature. From 57 independent estimation problems with a variance of 70m, the average error results in approximately 25m, demonstrating the technique as a potential candidate for GNSS-independent position estimation.**

***Index Terms*—GNSS-independent position estimation, Computer Vision, Vision-based navigation**

I. Introduction

A. Motivation

The Global Navigational Satellite System (GNSS) is crucial to the everyday operability of the maritime sector. Given its prominence in one of the world's most vital socio-economic markets, reliance on satellite navigation exposes a substantial security risk. This risk is accentuated by the increasing prevalence of known threats, including jamming, spoofing, and other Denial of Service (DoS) attacks [1], [2]. Consequently, an important research challenge that is addressed in this paper is GNSS-denied navigation for maritime vessels.

Extensive literature already exists studying Alternative Positioning Navigation and Timing (APNT) systems, utilizing everything from radio beacons to visual markers [3], [4] to infer the state of maritime vessels. However, where the necessary infrastructure for such solutions are not available, the use of standalone sensors is considered a viable alternative. In this paper, we present a novel, knowledge-based, data driven, algorithmic framework covered extensively in [5]. An optimization problem is posed by the correlations between semantic segmentations of visible terrain, captured by a monocular camera aboard the vessel and synthetic imagery generated through a known Digital Elevation Model (DEM). The approach draws parallels to other systems such as the skyline-based heading estimation approach in [6], and systems employing Visual Odometry (VO), extracting different visual features, like the 3-dimensional peaks proposed in [7] or the lunar craters proposed in [8]. Similarly, radars can be used to locate shorelines, buoys and beacons such as in [9], [10], whilst LiDARs can be used as in [11] to estimate the depths of potential features.

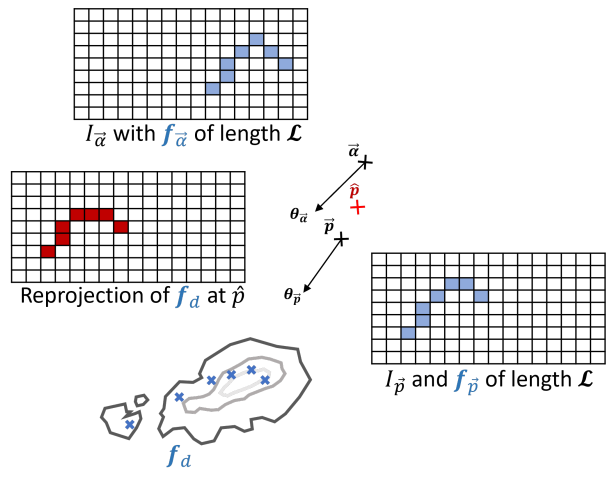
This research was partly funded by the Research Council of Norway through the Center of Excellence AMOS (grant number 223254) and the TRUSST project (grant number 313921).

To cover the many weaknesses of standalone sensors, multi-sensor fusion is becoming ever more prevalent. [12] propose using cameras and automatic identification systems in junction to improve the situational awareness of Autonomous Surface Vehicles (ASVs), whilst [13] fuse visual, acoustic and inertial data using an error-state Kalman filter for navigation of maritime vessels in narrow waterways. [3], [4] are further examples of how multi-sensor fusion is being investigated in improving Simultaneous Localization And Mapping (SLAM) solutions, allowing for even greater autonomy of unmanned vehicles.

B. Problem Description

We propose an algorithmic framework to estimate the 2-dimensional position $\vec{p}$ of a maritime vessel operating within a littoral zone, by using an assumed initial guess of the position $\vec{\alpha}$ with heading angle $\theta_{\vec{\alpha}}$. The maritime vessel is assumed to be equipped with a calibrated monocular camera, a heading reference system, as well as an accurate Digital Elevation Model (DEM) of its surroundings. Three sets of features are extracted and matched to form a motion only bundle adjustment optimization problem [14], which can be minimized to find a suitable estimation $\hat{p}$ of $\vec{p}$.

An illustrative example of the three feature sets $\mathbf{f}_{\vec{p}}$, $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_d$ can be seen in Fig. 1. All features illustrate the same peak



$I_{\vec{\alpha}}$ with $f_{\vec{\alpha}}$ of length $\mathcal{L}$
$\vec{\alpha}$
$\hat{p}$
$\theta_{\vec{\alpha}}$
$\vec{p}$
$\theta_{\vec{p}}$
Reprojection of f_d at $\hat{p}$
$I_{\vec{p}}$ and $f_{\vec{p}}$ of length $\mathcal{L}$
f_d

Fig. 1: Illustration of the captured image feature $\mathbf{f}_{\vec{p}}$, the synthetically generated feature $\mathbf{f}_{\vec{\alpha}}$ and the DEM feature $\mathbf{f}_d$. Estimation is done by aligning the reprojection of $\mathbf{f}_d$ with the target image feature $\mathbf{f}_{\vec{p}}$.

visible in $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_{\vec{p}}$. They are used to minimize the *reprojection error* between the set of 3-dimensional DEM-features $\mathbf{f}_d$ (1)

$$\mathbf{f}_d = \begin{bmatrix} x_{d_0} & y_{d_0} & z_{d_0} \\ x_{d_1} & y_{d_1} & z_{d_1} \\ \vdots & \vdots & \vdots \\ x_{d_{\mathcal{L}-1}} & y_{d_{\mathcal{L}-1}} & y_{d_{\mathcal{L}-1}} \end{bmatrix}, \tag{1}$$

and the set of 2-dimensional features $\mathbf{f}_{\vec{p}}$ captured by the camera aboard the maritime vessel at $\vec{p}$ (2)

$$\mathbf{f}_{\vec{p}} = \begin{bmatrix} x_{p_0} & y_{p_0} \\ x_{p_1} & y_{p_1} \\ \vdots & \vdots \\ x_{p_{\mathcal{L}-1}} & y_{p_{\mathcal{L}-1}} \end{bmatrix}, \quad \mathbf{f}_{\vec{\alpha}} = \begin{bmatrix} x_{\alpha_0} & y_{\alpha_0} \\ x_{\alpha_1} & y_{\alpha_1} \\ \vdots & \vdots \\ x_{\alpha_{\mathcal{L}-1}} & y_{\alpha_{\mathcal{L}-1}} \end{bmatrix}. \tag{2}$$

Conceptually, motion only bundle adjustment aligns the *reprojection* of $\mathbf{f}_d$ with $\mathbf{f}_{\vec{p}}$ by updating $\hat{p}$ from its initial value $\vec{\alpha}$. Implicitly, if no feature $\mathbf{f}_{\vec{p}}$ can be extracted, the nonlinear optimization is rendered infeasible.

Contrary to the feature set $\mathbf{f}_{\vec{p}}$, the feature set $\mathbf{f}_{\vec{\alpha}}$ is generated synthetically by simulating the projection of DEM data to a camera located at $\vec{\alpha}$, with heading $\theta_{\vec{\alpha}}$. Although not partaking in the final estimation, this allows for the extraction of $\mathbf{f}_d$.

C. Paper structure

The structure of this paper follows the flow of data throughout the framework, illustrated by the diagram in Fig. 2. Section II covers the target optimization problem, whilst Section III covers the compound algorithms **ExtractImageFeatures** and **ExtractDemFeature**, with the embedded algorithms **SkylineContour**, **LocatePeaks** and **SubdivideFrustum**. Sections IV and V finally present and analyse the estimator's performance within a simulator environment.

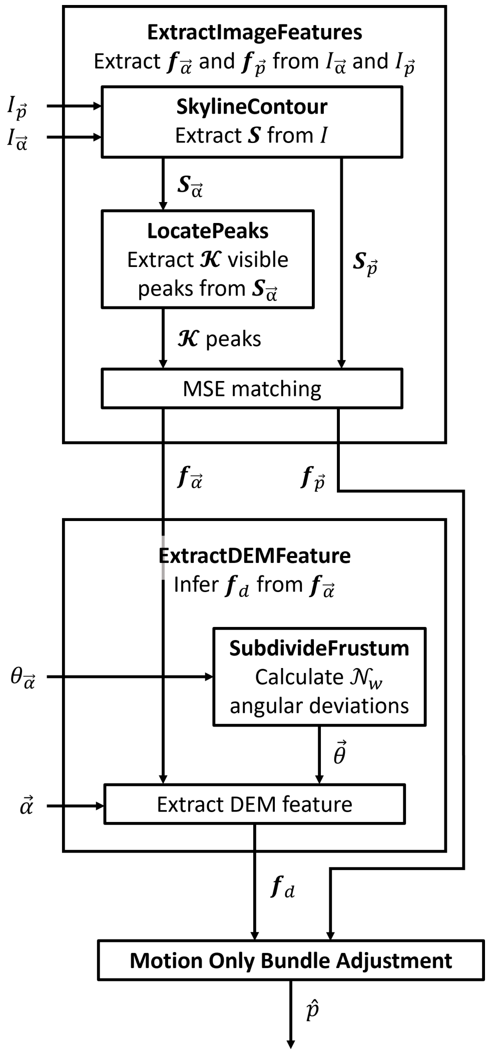


Fig. 2: Structure of the proposed algorithmic framework. 2 compound algorithms **ExtractImageFeatures** and **ExtractDEMFeatures** extract a set of features $\mathbf{f}_{\vec{\alpha}}$, $\mathbf{f}_{\vec{p}}$ and $\mathbf{f}_d$ from the semantic segmentations of visible terrain $I_{\vec{\alpha}}$ and $I_{\vec{p}}$. These are used to infer a position estimate $\hat{p}$ through a Motion Only Bundle Adjustment optimization problem.

II. Optimization Problem

The matrix $\mathbf{f}_d$ of 3-dimensional points in the DEM is projected onto the image plane of a pinhole camera mounted onboard the maritime vessel by the known, linear projection matrix $\mathbf{P}$, which is a compound transform of the camera's intrinsic parameter matrix $\mathbf{K}$ and extrinsic transform matrix $\mathbf{E}$. The Euclidean distance between all resulting 2-dimensional projections and $\mathbf{f}_{\vec{p}}$ can then be minimized to find $\hat{p}$

$$\hat{p}^* = \arg\min_{\hat{p}} e(\hat{p}) \tag{3}$$

where

$$\begin{aligned} e(\hat{p}) &= \textstyle\sum_{k=0}^{\mathcal{L}-1} ||\mathbf{P}\mathbf{f}_{d_k} - \mathbf{f}_{\vec{p}_k}||_2 \\ &= \textstyle\sum_{k=0}^{\mathcal{L}-1} ||(\mathbf{K}\mathbf{E})\mathbf{f}_{d_k} - \mathbf{f}_{\vec{p}_k}||_2 \end{aligned} \tag{4}$$

and the index k represents the k'th row-vector of the associated matrix. It is assumed that all features, whether located in image space or the DEM, contain $\mathcal{L}$ elements that are all uniquely comparable between features. In the proposed framework, these depict the same peak, visible in the images $I_{\vec{p}}$ and $I_{\vec{\alpha}}$. As all features and **K** are assumed constant, whilst the headings $\theta_{\vec{\alpha}}$ and $\theta_{\vec{p}}$ are assumed known, the only free parameters of (4) are the estimated spatial coordinates $\hat{p}$ of the maritime vessel embedded in **E**.

Given the nonlinearity of (3), the problem must be solved by a nonlinear solver. In this paper, an exhaustive grid search is adopted to form a baseline solution.

III. Algorithms

A. Matching image features

ExtractImageFeatures in Fig. 2 extracts a set of matching image features $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_{\vec{p}}$ from two semantic segmentations of images $I_{\vec{\alpha}}$ and $I_{\vec{p}}$ at positions $\vec{\alpha}$ and $\vec{p}$ with headings $\theta_{\vec{\alpha}}$ and $\theta_{\vec{p}}$. To accomplish this, a set of *skyline contours* $\mathbf{S}_{\vec{\alpha}}$ and $\mathbf{S}_{\vec{p}}$ in the form of 2-dimensional matrices are extracted from $I_{\vec{\alpha}}$ and $I_{\vec{p}}$ by the **SkylineContour** algorithm, covered in Section III-B. From $\mathbf{S}_{\vec{\alpha}}$, $\mathcal{K}$ visible peaks, which are sub-sequences of the contour, are extracted by the **LocatePeaks** algorithm, covered in Section III-C. **ExtractImageFeatures**, described in Algorithm 1, then employs a 1-dimensional similarity measure to find the best sub-sequence in $\mathbf{S}_{\vec{p}}$ matching one of the aforementioned $\mathcal{K}$ peaks. The optimal sequences are returned as the feature sets $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_{\vec{p}}$.

The employed heuristic is a 1-dimensional application of the Mean Square Error (MSE). The similarity between two arbitrary sub-sequences of $\mathbf{S}_{\vec{\alpha}}$ and $\mathbf{S}_{\vec{p}}$, $\mathbf{Q}_{\vec{\alpha}}$ and $\mathbf{Q}_{\vec{p}}$, both consisting of $\mathcal{L}$ rows, is computed as the average row-wise vertical offset (5)

$$MSE = \frac{1}{\mathcal{L}-1}\sum_{k=0}^{\mathcal{L}} |\mathbf{Q}_{\vec{\alpha}_k}(y) - \mathbf{Q}_{\vec{p}_k}(y)|. \qquad (5)$$

One should note that (5) is viable as a heuristic because of the assumption of negligent pitch and yaw of the maritime vessel, implied by the 2-dimensional optimization problem.

B. Skyline contour from semantic segmentation

SkylineContour in Algorithm 2 utilizes a variety of image-processing techniques to extract skyline contours $\mathbf{S}_{\vec{\alpha}}$ and $\mathbf{S}_{\vec{p}}$ from the semantic segmentations of visible terrain $I_{\vec{\alpha}}$ and $I_{\vec{p}}$. Firstly, morphological erosion ($\ominus$), dilation ($\oplus$), opening ($\circ$) and closing ($\bullet$) are used with *structuring elements* E_n and E_e to reduce pixel noise and extract a full contour of the visible terrain. Secondly, the coastline of the contour is removed by a simple iterative algorithm, leaving only the topmost skyline contour of the visible terrain.

C. Extract peaks from skyline contour

LocatePeaks in Algorithm 3 is tasked with finding $\mathcal{K}$ potential features in $\mathbf{S}_{\vec{\alpha}}$, on the form of visible peaks within the contour. One such peak **H** is a continuous sub-sequence of $\mathbf{S}_{\vec{\alpha}}$ and subject to a set of rigid conditions ensuring the non-ambiguity of the feature:

- The elements of **H** form a convex hull with negative semi-definite edges.
- The elements of **H** cannot form a slope.
- **H** cannot reach the horizontal boundaries of $I_{\vec{\alpha}}$.
- **H** has to be at least t_w pixels wide and t_h pixels high.
- There can be no overlap between two peaks, that is $\mathbf{H}_j \cap \mathbf{H}_k = \emptyset$, where $j, k \in [1, \mathcal{K}]$ & $j \neq k$.

The $\mathcal{K}$ peaks are extracted through a depth-first scheme, iterating through the rows of $I_{\vec{\alpha}}$ from top to bottom.

Algorithm 1 Match features between $I_{\vec{\alpha}}$ and $I_{\vec{p}}$

```
procedure ExtractImageFeatures($I_{\vec{\alpha}}$, $I_{\vec{p}}$)
    $\mathbf{H}_{\vec{\alpha}} \leftarrow$ LocatePeaks(SkylineContour($I_{\vec{\alpha}}$), ...)
    if $\mathbf{H}_{\vec{\alpha}} = \emptyset$ then
        Terminate: No visible peaks in $I_{\vec{\alpha}}$.
    end if
    $\mathbf{S}_{\vec{p}} \leftarrow$ SkylineContour($I_{\vec{p}}$)
    $\mathbf{H}^*, \mathbf{S}^*_{\vec{p}} \leftarrow$ []
    $MSE^* \leftarrow 0$
    for $\mathbf{H} \in \mathbf{H}_{\vec{\alpha}}$ do
        $j \leftarrow 0$
        while $j \leq len(\mathbf{S}_{\vec{p}}) - len(\mathbf{H})$ do
            $\mathbf{P} \leftarrow \mathbf{S}_{\vec{p}}[j : j + len(\mathbf{H}), :]$
            if $\mathbf{P}$ discontinuous then
                $j \leftarrow$ Index after discontinuity
                continue
            else
                $MSE \leftarrow$ MSE($\mathbf{H}, \mathbf{P}$)
                if $MSE < MSE^*$ then
                    Update $MSE^*$, $\mathbf{H}^*$ and $\mathbf{S}_{\vec{p}^*}$
                end if
                $j \leftarrow j + 1$
            end if
        end while
    end for
    return $\mathbf{H}^*, \mathbf{S}^*_{\vec{p}_t}$
end procedure
```

Algorithm 2 Skyline contour **S** from the image I

```
procedure SkylineContour($I$)
    $I_b \leftarrow$ BinaryThreshold($I$)
    $\mathbf{E}_n \leftarrow$ StructuringElement(($3 \times 3$), full)
    $I_b \leftarrow (I_b \bullet \mathbf{E}_n) \circ \mathbf{E}_n$
    $\mathbf{E}_e \leftarrow$ StructuringElement(($3 \times 3$), cross)
    $\mathbf{S} \leftarrow$ BITWISE_XOR($I_b$, $I_b \ominus \mathbf{E}_e$)
    for $x \in$ [0, len($\mathbf{S}_x$)] do
        $\vec{col} \leftarrow \mathbf{S}[x, :]$
        $\mathbf{S}$.Remove([$x$, min($\vec{col}$)])
    end for
    Return($\mathbf{S}$)
end procedure
```

Algorithm 3 Extract $\mathcal{K}$ visible peaks from $\mathbf{S}$

```
Require: Horizontal boundaries $I_{min}$, $I_{max}$ of $I$.
  procedure LocatePeaks($\mathbf{S}$, $t_w$, $t_h$, $I_{min}$, $I_{max}$)
      $peaks \leftarrow \{\}$
      for $y \in [\mathbf{S}_{y,max}, \mathbf{S}_{y,min}]$ do
          $\vec{row} \leftarrow \mathbf{S}[:, y]$
          if nonzero($\vec{row}$) $\neq \emptyset$ then
              $\mathbf{C} \leftarrow$ Continuous sequences in $\vec{row}$.
              for $\mathbf{H} \in \mathbf{C}$ do
                  Grow $\mathbf{H}$ to a convex hull
                  if $\mathbf{H}$ meets requirements then
                      $peaks$.Add($\mathbf{H}$)
                  end if
              end for
          end if
      end for
      Return($peaks$)
  end procedure
```

Algorithm 4 Extract $\mathbf{f}_d$

```
Require: DEM $I_d$.
Require: Parameters of calibrated camera $\mathcal{N}_w$ and $FOV_H$.
  procedure ExtractDEMFeature($I_d$, $\mathbf{f}_{\vec{\alpha}}$, $\vec{\alpha}$, $\theta_{\vec{\alpha}}$, $\mathcal{N}_w$, $FOV_H$)
      $\mathbf{f}_d \leftarrow []$
      $\vec{\theta} \leftarrow$ DivideCameraFrustum($\mathcal{N}_w, FOV_H, \theta_{\vec{\alpha}}$)
      for $i \in (0, \mathcal{N}_w]$ do
          if $i \notin \mathbf{f}_{\vec{\alpha},x}$ then
              Continue
          end if
          $\theta_i \leftarrow \vec{\theta}[i]$
          $\mathbf{C}_i \leftarrow$ CrossSectionDEM($I_d, \vec{\alpha}, \theta_i$)
          $\mathbf{f}_d$.append($I_d$[argmax$_\theta$($\mathbf{C}_i$)])
      end for
      return $\mathbf{f}_d$
  end procedure
```

D. DEM Feature Extraction

ExtractDEMFeature in Fig. 2 utilizes the image feature $\mathbf{f}_{\vec{\alpha}}$ from **ExtractImageFeatures** to extract a set of $\mathcal{L}$ 3-dimensional points making up the DEM feature $\mathbf{f}_d$. As the position $\vec{\alpha}$ and heading $\theta_{\vec{\alpha}}$ are known, $\mathcal{L}$ 2-dimensional lines can be parameterized into the DEM by linear raycasts from the *optical center* of the camera aboard the maritime vessel to each of the $\mathcal{L}$ elements of $\mathbf{f}_{\vec{\alpha}}$ residing on the camera's normalized image plane. For $\mathbf{f}_d$ to match the feature $\mathbf{f}_{\vec{\alpha}}$ residing on the skyline contour of the visible terrain in $I_{\vec{\alpha}}$, the 3-dimensional point forming the highest angle to the camera's *optical axis* along each line has to be returned. These represent the topmost visible points projected back to the vessel for each element of $\mathbf{f}_{\vec{\alpha}}$. To this end, **ExtractDEMFeature** in Algorithm 4 illustrate how a cross-section of the DEM is extracted along each of the $\mathcal{L}$ parameterized lines. These cross-sections can be mapped to a 2-dimensional plane with the altitudes of extracted points along one axis and the Euclidean distances to the camera's optical center along the other. The angle that each point forms with the camera center can then easily be computed.

ExtractDEMFeature takes in a 1-dimensional vector $\vec{\theta}$ containing the angular deviations between the $\mathcal{L}$ lines, as illustrated in Fig. 2. In fact, $\mathcal{N}_w$ deviations are calculated by **SubdivideFrustum** in Algorithm 5, where $\mathcal{N}_w$ is the width of the image plane in pixels. **ExtractDEMFeature** only makes use of the deviations relating to the $\mathcal{L}$ nonzero elements in $\mathbf{f}_{\vec{\alpha}}$. The division of the entire camera frustum is a fairly simple step, calculated through a number of closed form, trigonometric expressions. Fig. 3 illustrates the subdivision of a camera frustum with normalized focal length f and known horizontal Field of View FOV_H into $\mathcal{N}_w - 1$ smaller, non-right triangles forming angles θ_m, $m \in [1, \mathcal{N}_w - 1]$ with the camera's optical center. Each of these triangles have a far side l, which is found by a simple division of the normalized image plane

$$l = \frac{2\tan(\frac{FOV_H}{2})}{\mathcal{N}_w - 1} \tag{6}$$

and two legs d_{m-1} and d_m, found by applying Pythagoras theorem to a right triangle formed with the normalized focal length f

$$d_m = \begin{cases} \sqrt{1 + (\tan(\frac{FOV_H}{2}) - lm)^2}, & \text{if } lm \leq \tan(\frac{FOV_H}{2}) \\ \sqrt{1 + (lm - \tan(\frac{FOV_H}{2}))^2}, & \text{otherwise.} \end{cases} \tag{7}$$

These are used iteratively to find the $\mathcal{N}_w - 1$ angles by a closed form expression based on the law of cosines:

$$\theta_m = \cos^{-1}(\frac{d_{m-1}^2 + d_m^2 - l^2}{2d_{m-1}d_m}). \tag{8}$$

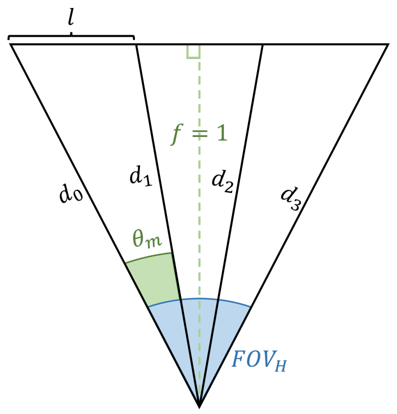


Fig. 3: Subdivision of camera frustum cross-section into $\mathcal{N}_w$ angles θ_m, with $m \in [1, \mathcal{N}_w - 1]$. The equidistant lengths between rays on the image plane are denoted l and all sides are denoted d. The camera focal length f is normalized.

Algorithm 5 Calculate $\mathcal{N}_w$ angular deviations

```
procedure SubdivideFrustum($\mathcal{N}_w$, $FOV_H$, $\theta_{\vec{\alpha}}$)
    $\vec{\theta} \leftarrow [\theta_{\vec{\alpha}} - \frac{FOV_H}{2}]$
    $l \leftarrow$ CalculateL()
    for $m \in [1, \mathcal{N}_w]$ do
        $l_{m-1} \leftarrow l \times (m-1)$
        $l_m \leftarrow l \times m$
        $d_{m-1} \leftarrow$ CalculateSide$(l_{m-1}, FOV_H)$
        $d_m \leftarrow$ CalculateSide$(l_m, FOV_H)$
        $\vec{\theta}$.append(CosineLaw$(d_{m-1}, d_m, l) + \vec{\theta}[m-1]$)
    end for
    return $\vec{\theta}$
end procedure
```

IV. Results

The framework was tested in a custom simulator environment, on a set of 3 predefined paths. Each path consisted of 19 *waypoints*, denoting the initial positions $\vec{\alpha}_t$, $t \in [0, 19]$. The target positions $\vec{p}_t$ were generated by adding Gaussian noise $\mathcal{N}(0, \sigma^2)$ along the ground-plane, forming 19 independent estimation problems per path. All waypoints are connected by a piecewise linear path, meaning the heading $\theta_{\vec{\alpha}_t}$ at any given waypoint is supplied by a simple arctan2 expression between $\vec{\alpha}_t$ and $\vec{\alpha}_{t-1}$. One should note that during deployment at $t = 0$, the position $\vec{\alpha}_0$ is considered known, rendering $\vec{p}_0 = \vec{\alpha}_0$. The 3 paths and the DEM utilized can be seen in their entirety in Fig. 4.

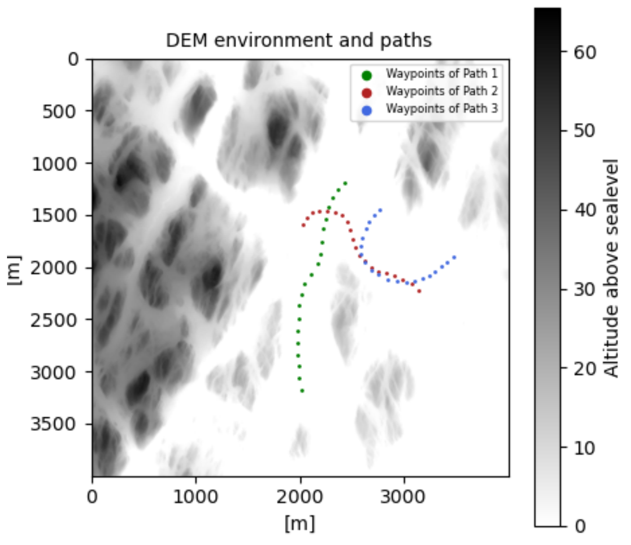
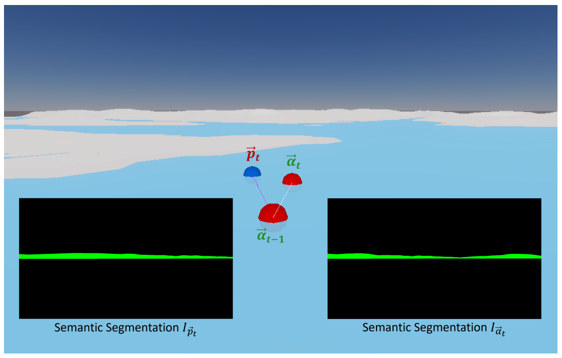


Fig. 4: The waypoints of all 3 paths used for data generation inside the external DEM, acquired from The Norwegian Mapping Authority [15].

The following section seeks to present the synthetic data generated in the simulator environment and a selection of intermediate and final results produced by the framework.

A. Simulator Data

A sample of the synthetic data generated at each waypoint can be seen in Fig. 5. In accordance with Fig. 2, two semantic segmentations $I_{\vec{\alpha}_t}$ and $I_{\vec{p}_t}$ are generated, and the positions $\vec{\alpha}_t$, $\vec{\alpha}_{t-1}$ and $\vec{p}_t$ are returned. $\vec{\alpha}_{t-1}$ is used to infer $\theta_{\vec{\alpha}_t}$, and $\vec{p}_t$ to verify the estimators performance. In generating $\vec{p}_t$, a variance of $\sigma^2 = 70$m was utilized.



$\vec{p}_t$
$\vec{\alpha}_t$
$\vec{\alpha}_{t-1}$
Semantic Segmentation $I_{\vec{p}_t}$
Semantic Segmentation $I_{\vec{\alpha}_t}$

Fig. 5: Synthetic data at each waypoint: Two semantic segmentations $I_{\vec{\alpha}_t}$ and $I_{\vec{p}_t}$, as well as positions $\vec{\alpha}_t$, $\vec{\alpha}_{t-1}$ and $\vec{p}_t$.

B. Feature extraction and matching

The synthetic data from the simulator was collected in its entirety prior to being processed by the proposed estimation framework in Fig. 2.

Fig. 6 illustrates the output of **ExtractImageFeatures** at two waypoints, where a matching has been performed correctly and another has resulted in a mismatch.

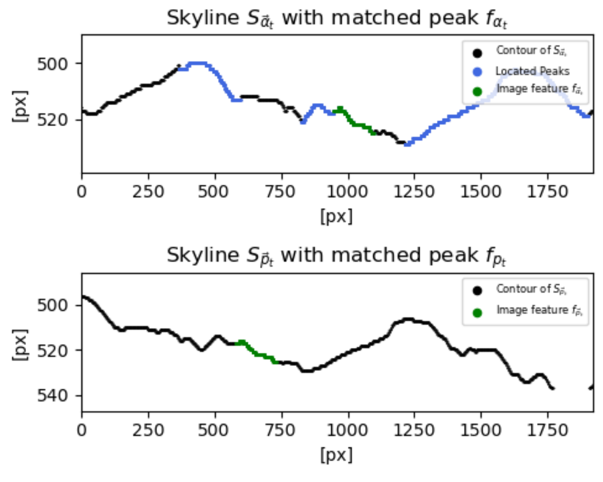
C. Pose estimations

The estimator performance was evaluated for each independent problem by the Euclidean distance in meters [m] between $\hat{p}_t$ and $\vec{p}_t$. Table I illustrates these for 11 out of 19 waypoints.

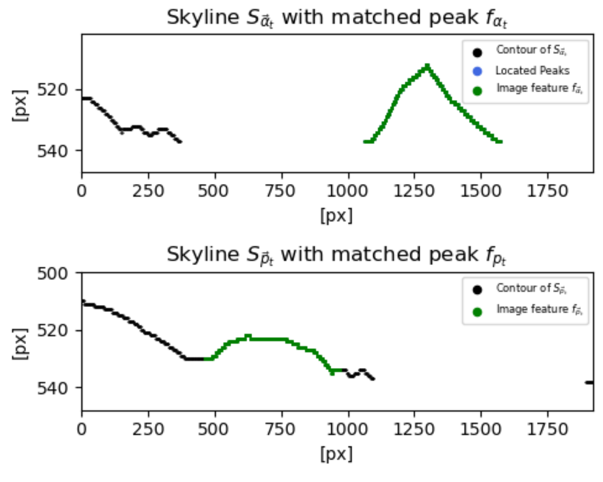
TABLE I: Euclidean Estimation Errors

Waypoint	Euclidean errors		
	Path 1	***Path 2***	***Path 3***
1	71.50m	4.24m	11.31m
2	59.64m	5.0m	32.80m
3	14.76m	73.82m	9.84m
4	30.87m	22.36m	26.92m
5	2.0m	15.03m	8.60m
6	59.80m	0.0m	10.77m
⋮	⋮	⋮	⋮
15	—*	21.21	4.47m
16	6.08m	30.67m	9.48m
17	21.02m	24.51m	54.70m
18	79.75m	26.24m	26.17m
19	5.0m	34.92m	19.10m
Average	**37.57m**	**16.30m**	**21.17m**

*Infeasible estimation.



(a) Waypoint 2 of path 2.



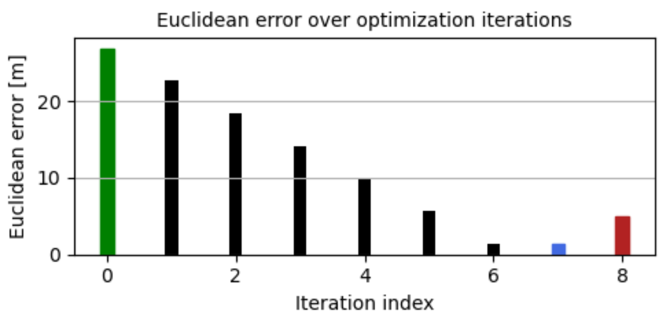
(b) Waypoint 18 of path 1.

Fig. 6: $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_{\vec{p}}$ resulting from **ExtractImageFeatures** applied at different waypoints. $\mathcal{K}$ blue peak are extracted from $\mathbf{S}_{\vec{\alpha}}$ and matched in $\mathbf{S}_{\vec{p}}$. The resulting features $\mathbf{f}_{\vec{\alpha}}$ and $\mathbf{f}_{\vec{p}}$ in green can be seen over $\mathbf{S}_{\vec{\alpha}}$ and $\mathbf{S}_{\vec{p}}$ respectively.

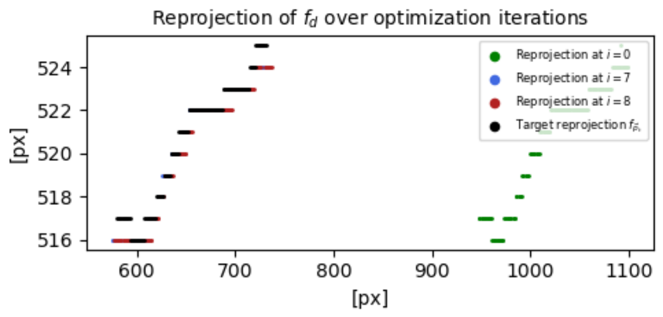
Fig. 7-9 illustrate 3 typical outcomes of the exhaustive grid search applied to (3): A successful estimation in Fig. 7, an estimation falling into a local minimum in Fig. 8 and a failed estimation in Fig. 9. Fig. 7a-9a illustrate the Euclidean error of the position-estimates from $\vec{\alpha}_t$ until $\hat{p}_t$, whilst Fig. 7b-9b highlight the distance between the reprojection of $\vec{f_d}$ to the target reprojection $\vec{f_{\vec{p}_t}}$. Red is used to highlight the final estimations $\hat{p}_t$, whilst other colors are used to highlight changes in the gradient of the Euclidean error of the position-estimate.

V. Discussion and Future Work

Although the results in Table I are far outperformed by state of the art GNSS receivers at sea, providing precision up to the range of a meter, it is important to note that the presented average errors are somewhat inflated by more

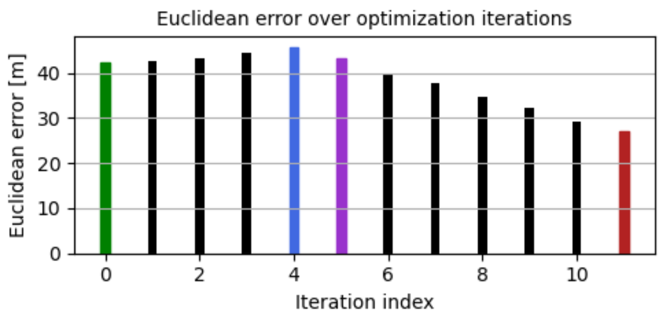


(a) Euclidean error of position estimate [m].

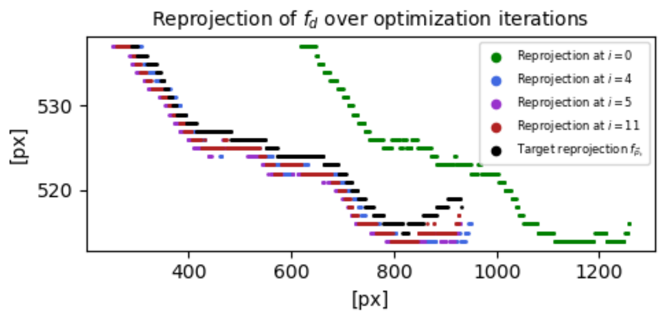


(b) Reprojection of $\mathbf{f}_d$ at given iterations of optimization.

Fig. 7: Successful estimation at waypoint 2 of path 2.

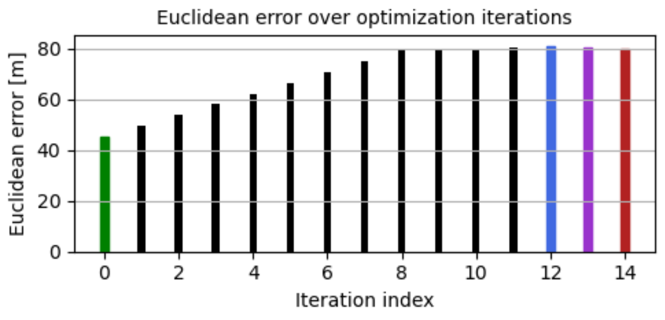


(a) Euclidean error of position estimate [m].

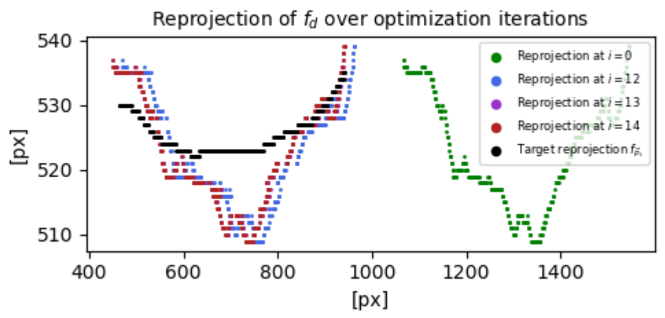


(b) Reprojection of $\mathbf{f}_d$ at given iterations of optimization

Fig. 8: Suboptimal estimation at waypoint 4 of path 3.



(a) Euclidean error of position estimate [m].



(b) Reprojection of $\mathbf{f}_d$ at given iterations of optimization

Fig. 9: Failed estimation at waypoint 18 of path 1.

catastrophic estimations and frequent deviations brought about by minor systemic flaws. In the majority of cases the presented framework converges towards the target position, but falls into local minima, as in Fig. 8, or proves overly sensitive close to the solution, as in Fig. 7. It does, however, bring about a noticeable decrease in the reprojection error (4).

Fig. 9 represents one of these more drastic failures of the system, recording a significant Euclidean error just below $80m$. Such errors occur fairly infrequently, but tend to have the erroneous matching of image features $\mathbf{f}_{\vec{\alpha}_t}$ and $\mathbf{f}_{\vec{p}_t}$ in common. Fig. 6b illustrates this for the above example, showcasing a set of "matches" clearly depicting different parts of the visible terrain. One can see how this renders the solving of (4) infeasible in Fig. 9b. The reprojection of $\mathbf{f}_{d_t}$ cannot be meaningfully aligned with $\mathbf{f}_{\vec{p}_t}$ as it stems from $\mathbf{f}_{\vec{\alpha}_t}$ in **ExtractDEMFeatures**, prompting a pseudo-random position estimate visible in Fig. 9a. Throughout the 3 recorded paths, such mismatches seem to occur in 3 specific scenarios:

- All visible features in $\mathbf{S}_{\vec{\alpha}_t}$ go out of frame between $I_{\vec{\alpha}_t}$ and $I_{\vec{p}_t}$, as is the case in Fig. 6b.
- Flat stretches of terrain and uniform peaks result in ambiguous image features, which are more easily misclassified.
- $\mathbf{f}_{\vec{\alpha}_t}$ of length $\mathcal{L}$ cannot be retrieved in $\mathbf{S}_{\vec{p}_t}$ due to significant changes in perspective.

This is where the modularity of the framework has the possibility to shine, being designed so as to adapt to different hardware and conditions. Changing the monocular camera aboard the maritime vessel to allow for panoramic imagery could wholly resolve the first problem and aid the second, whilst adopting an adaptive length $\mathcal{L}$ in Algorithm 1 could limit misclassifications due to perspective distortions. Alternatively, vast training sets could be generated synthetically and fed to a machine learning model, potentially improving the performance, accuracy and adaptability of the system as a whole.

To achieve greater precision however, the nonlinear solver has to be considered. Fig. 8 is a prime example of poor estimations occurring even with a great set of features, as the estimator falls into local minima. Although improvements using global solvers should be investigated for completeness, the ambiguity in reprojections of $\vec{f_d}$ in various minima call for the integration of said framework into greater sensor fusions, using other sensors to more accurately distinguish between different minima. This would also work to remedy the many inherit shortcomings of cameras, such as the loss of precision over longer distances and ineptitude in suboptimal visual conditions.

VI. Conclusion

This paper demonstrates how existing information about the surroundings and intended positions of a maritime vessel can allow for fairly reliable, monocular pose estimation at sea. A variation of motion only bundle adjustment was adopted to solve the proposed problem, minimizing the reprojection error between a set of features found and matched by the modular framework. Although results at this stage render the current implementation infeasible for active deployment, the consistent convergence towards the actual position of the vessel, combined with the modularity of the framework, makes it an interesting prospect for future development.

References

[1] A.C. O'Connor et al., "Economic Benefits of the Global Positioning System (GPS)," RTI International, 2019

[2] T. Westbrook, "The Global Positioning System and Military Jamming: The geographies of electronic warfare," *Journal of Strategic Security*, vol. 12, 2019

[3] H. Helgesen, T. Fuglestad, K. Cisek, B. Vik, Ø. K. Kjerstad, T. A. Johansen, "Inertial Navigation aided by Ultra-Wideband Ranging for Ship Docking and Harbor Maneuvering," *IEEE Journal of Oceanic Engineering*, Vol. 48, pp. 27-42, 2023

[4] Ø. Volden, A. Stahl, T.I. Fossen, "Vision-based positioning system for auto-docking of unmanned surface vehicles (USVs)," *Int J Intell Robot Appl.*, Vol 6, pp. 86–103, 2022.

[5] M. Roedele, "GNSS-free maritime navigation using DEM data and monocular camera images", M.S thesis, IE Dept., Norwegian University of Science and Technology, Trondheim, Norway, 2023.

[6] C. D. Rodin, A. Stahl, T. A. Johansen, "Skyline Based Camera Attitude Estimation Using a Digital Surface Model," 15th Int.Workshop on Advanced Motion Control, Tokyo, 2018

[7] L. Wei, S. Lee, "3D peak based long range rover localization," presented at the 7th International Conference on Mechanical and Aerospace Engineering (ICMAE), pp. 600-604, 2016

[8] L. Matthies, S. Daftry, S. Tepsuporn, Y. Cheng, D. Atha, R. M. Swan, S. Ravichandar, M. Ono, "Lunar Rover Localization Using Craters as Landmarks," presented at the IEEE Aerospace Conference, 2022

[9] D. Dagdilelis, M. Blanke, R. H. Andersen, R. Galeazzi, "Cyber-resilience for marine navigation by information fusion and change detection," *Ocean Engineering*, Volume 266, Part 3, paper 112605, 2022

[10] H. Ma, E. Smart, A. Ahmed, D. Brown, ”Radar Image-Based Positioning for USV Under GPS Denial Environment,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, pp. 72-80, 2018

[11] J. Graeter, A. Wilczynski, M. Lauer, ”LIMO: Lidar-Monocular Visual Odometry,” arXiv 2105.14021, 2018

[12] R. W. Liu, Y. Guo, J. Nie, Q. Hu, Z. Xiong, H. Yu, M. Guizani, ”Intelligent Edge-Enabled Efficient Multi-Source Data Fusion for Autonomous Surface Vehicles in Maritime Internet of Things,” *IEEE Transactions on Green Communications and Networking*, vol. 6, pp. 1574-1587, 2022

[13] Ø. Volden, D. Cabecinhas, A. Pascoal, T. I. Fossen, ”Development and experimental evaluation of visual-acoustic navigation for safe maneuvering of unmanned surface vehicles in harbor and waterway areas,” *Ocean Engineering*, Volume 280, paper 114675, 2023

[14] T. V. Haavardsholm. (2021). A handbook in Visual SLAM [Handbook]. Available: https://github.com/tussedrotten/vslam-handbook

[15] Høydedata, The Norwegian Mapping Authority, 2023. [Online]. Available: https://hoydedata.no/LaserInnsyn2/