Usama Mujahid

Evaluation of feature extraction techniques on aquaculture place recognition problem

Graduate thesis in Marine Technology
Supervisor: Asgeir Johan Sørensen
Co-supervisor: Oscar Pizarro
June 2023

Usama Mujahid

Evaluation of feature extraction techniques on aquaculture place recognition problem

Graduate thesis in Marine Technology
Supervisor: Asgeir Johan Sørensen
Co-supervisor: Oscar Pizarro
June 2023

Norwegian University of Science and Technology
Faculty of Engineering
Department of Marine Technology

NTNU
Norwegian University of
Science and Technology

Abstract

Aquaculture industry is one of the most sustainable and environmental friendly way of satisfying the world's increasing food demand. However, fish escape still remains a critical challenge that is not only a financial loss but badly effects the biodiversity as well. These escapes mostly happens through a hole caused by a natural event or an accident hence it is essential to have a mechanism to detect it. SLAM(simultaneous localization and mapping) is the current state of the art method to resolve issues like this. The aim of this thesis is to focus on the Place recognition and loop closure part of the SLAM. BOW technique along with different feature extraction methods will be evaluated and compared in different realistic scenarios. Moreover, the performance of these techniques on non net cage marine data will be also discussed to establish a strong argument.

Acknowledgement

I would like to express my deepest gratitude to my supervisor, Asgier Johanson, and my co-supervisor Oscar Pizarro, for their invaluable guidance, support, and mentorship throughout the course of this thesis. Their expertise, patience, and encouragement have been instrumental in shaping the direction of my research and enhancing my academic growth.

Contents

Figures

Tables

Chapter 1

Introduction

1.0.1 Motivation and Background

According to estimates, global food consumption will be increased by 70% from 2016 to 2030 (*Offshore fish farming* 2020). The issue of how to satisfy such a demand in a world where climate change is a factor is thus very crucial. Even though just 2% of the food consumed by humans today comes from the ocean ibid., seafood is one of the most environmentally friendly options. The main reason behind is that plankton and algae—the trophic level of the ecosystem with the lowest food supply—are consumed by the majority of marine creatures. As a result, fish farms are a practical way to meet these needs. However, fish escapes is one of the biggest challenges of modern aquaculture that can have daring consequences on the biodiversity of the ecosystem. The chances of fish escapes increases much more in the exposed conditions due to turbulence and high sea currents. It has been documented that the two-third of the fish escape happening in the Norwegian aquaculture is caused by the tears in the net(Thorvaldsen, Holmen and Moe, 2015).
Net repairs are typically carried out by human divers, but this task involves some risk due to the potential entanglement of their heavy equipment with the net, which can occasionally have fatal consequences. Therefore, precise identification of these holes is of utmost importance. Accurate detection of the fault points provides two significant advantages.

- It reduces the diving associated risk as the diver already knew the location and extent of the hole and can plan accordingly
- The correct estimates and a 3d map of the net can be use-full in future net repairing, when this will be done by the robots

Currently the state of the art solution for localization and tracking where GPS does not work is the SLAM (Simultaneous localization and mapping).Visual Simultaneous Localization and Mapping (SLAM) systems rely heavily on place recognition. It speaks to a SLAM system's capacity to identify previously visited or well-known spots as the camera moves across an environment. With the aid of place recognition, the system can create loop closures, which are essential for lowering cumulative errors and enhancing the precision of the SLAM trajectory.

1.0.2 Scope and objective

Place recognition is a crucial step in the Simultaneous Localization and Mapping (SLAM). The focus of this study is centered around assessing the performance of widely utilized descriptors, namely AKAZE, BRISK, ORB, SURF, and SIFT, in the context of place recognition, particularly in the aquaculture industry. The comprehensive analysis encompasses feature extraction, feature matching, long-term and short-term tracking, as well as timing considerations. Furthermore, experiments are conducted using both fish net cage and underwater cave datasets to highlight the distinct characteristics of the aquaculture environment.

1.0.3 Literature review

Due to very low feature value and almost identical meshes on the fish net cage, it is considered as one of the hardest application on the SLAMs algorithm. But it is growing in attention for the past few years.

In the framework of visual simultaneous localization and mapping (vSLAM), the study "Evaluation of Several Feature Detectors/Extractors on Underwater Images towards vSLAM" (Hidalgo and Bräunl, 2020)focuses on analyzing different feature detectors and extractors for underwater images. The authors want to find the best feature extraction and detection techniques that can tackle the difficulties presented by underwater environments.

Using quantitative parameters including repeatability, matching accuracy, and computing economy, the study analyzes the effectiveness of several feature detectors and extractors on underwater photos. Popular methods including SIFT, SURF, ORB, BRISK, and AKAZE are among those that were tested.

The authors provide judgments regarding the efficiency of each strategy in underwater scenarios in light of the evaluation's findings. The properties of the seafloor, types of objects, lighting, color, and turbidity were used to categorize various datasets. The amount of features that were retrieved from the photos and then matched in the succeeding frames shows how these impacts affected the photographs. The findings revealed that the presence of turbidity and blurriness reduced the number of characteristics and matches. The survey offers a wealth of data and thorough insights that are important for vSLAM application decision-making. With the shortest computation time, the ORB detector/descriptor shone out in terms of detection and matching performance, making it a good choice for developing vSLAM.

The article (Tareen and Saleem, 2018) offers a thorough evaluation and comparison of various well-liked feature descriptors used in computer vision, including SIFT, SURF, KAZE, AKAZE, ORB, and BRISK. The authors' goal is to assess and comprehend the performance traits and constraints of these descriptors under diverse circumstances.

Each descriptor is examined in the study from a variety of angles, such as their robustness to picture alterations, computing effectiveness, scale invariance, and matching precision. The evaluation uses quantitative parameters including repeatability, matching score, and computing time and is based on a number of benchmark datasets.The analysis is used to highlight the advantages and disadvantages of each descriptor in the study. It talks about how they perform in various situations, like noise, scale changes, and viewpoint alterations. It is discovered that ORB is least scale invariant. Compared to other functions, ORB(1000), BRISK(1000), and AKAZE are more rotation invariant. In general, ORB and BRISK are more resistant to affine alterations than the others. In comparison to the others, SIFT, KAZE, AKAZE, and BRISK have greater picture rotation accuracy. Despite the fact that ORB and BRISK are the most effective algorithms for detecting a big number of features, the time required to match such a large number of features increases the overall image matching time. Contrarily, ORB(1000) and BRISK(1000) match images the quickest, but at the expense of precision. For all kinds of geometric transformations, SIFT and BRISK are determined to have the highest overall accuracy, and SIFT is declared to be the most accurate method.

The difficulties given by the distinctive features of underwater scenes, such as poor vision and limited color information, are the writers' main focus in (Li, Eustice and Johnson-Roberson, 2015). This research includes a thorough investigation into the extraction and evaluation of various high-level visual cues for underwater place recognition. These characteristics include the color histogram, the histogram of oriented gradients (HOG), and local binary patterns (LBPs). The authors compare these features based on their discriminative capacity and robustness to changes in lighting and visibility conditions after analyzing how well these features work on datasets of underwater images. The findings of the experiments show that high-level visual features, in particular HOG and LBP, perform well in identifying underwater locations. These features successfully capture the peculiar patterns and textures found in underwater environments, allowing for accurate location matching and identification. To detect visually salient regions and high level features, a salient region identification approach is proposed. Features are described and matched using SVM classifiers built on HOG features. Geometric constraints are used to disqualify SVM's false positive matches. The method's effectiveness is assessed using real data gathered during multi-year ship hull inspection missions, and it is contrasted with that of other industry-standard location recognition techniques. The suggested method performs significantly better than conventional point-based feature matching methods.

Another notable work done by a fellow NTNU student Straume Haugland, 2021. The simultaneous localization and mapping (SLAM) challenge for a remotely operated vehicle (ROV) operating inside a net cage for aquaculture is addressed in this paper. It suggested a six-degrees-of-freedom pose-graph SLAM technique with improved Doppler velocity log (DVL) visual loop closures. Through this work, a brand-new data association technique has been developed for using a mono camera to solve the SLAM loop closure problem inside a fish cage. The algorithm is based on how a ROV inspects a net while moving through it; in this process, the ROV is pointing in the direction of the net. The similarity of the depth and heading measurements at the time of the images' capture is therefore used to filter possible candidates for image loop closure. A global saliency map was modified to help with this filtering process and prevent matching low featured scenes in order to take into account the fish cage's sparsely featured environment. The best loop closure candidates were sorted using the cosine similarity of term frequency-inverse document frequency (TF-IDF) histograms of image visual words after the image candidates had been filtered based on these three criteria. SIFT based visual descriptor is used as a primary feature extraction model.

System loss is caused by fish cage dysfunctions from both an operational and financial standpoint. Fish can get out of the net because of its poor construction. It is necessary to conduct routine inspections in order to lower the fish mortality rate. The design of a small-sized autonomous vehicle-based inspection system for underwater fish cages was covered by the authors of Chalkiadakis et al., 2017 in this regard. While the vehicle navigates on its own throughout the inspection, the scheme gives facilities for net hole detection. The OpenCV triangulation approach, which is based on target identification in the camera image, is used to estimate depth. The vehicle is given instructions to go ahead or behind based on the depth data. The plan was successfully tested in a real-world setting. However, top-down movement control is necessary to increase the system's autonomy.

Due of the GPS system's inability to function underwater, ROV/AUV-based aquaculture inspection presents localization challenges. In contrast, the surface vehicle area is simple to set up and maintain with fewer localization and communication restrictions. The design and use of an omnidirectional surface vehicle (OSV) for fish cage inspection activities was explored in Tao et al., 2018. The vehicle was equipped with a depth-adjustable camera that records the net structure at various depths. A pre-trained deep-learning-based solution was also used to handle the problem of net damage identification. However, none of the variables that affect position estimate were taken into account. By introducing a mission planning method based on artificial intelligence, the authors of Lin, Tao and F. Zhang, 2020 provided an expansion of a prior study. To establish the guidelines for vehicle movement, a hierarchical task network was utilized. However, it is not tested in the real world.

The stability of fish nets and fish health are significantly impacted by standard biofouling cleaning techniques, which are also expensive. The leftover waste materials harm the fish by creating an unfavorable environment. An advanced solution in this area is provided by ROV/AUV-based biofouling detection and eradication. Static sensors are also employed to continuously track environmental variables. A thorough theoretical examination of robotic systems for biofouling prevention and inspection in fish farms was described by the authors of (Ohrem, Kelasidi and Bloecher, 2020). Different operational and technical requirements are suggested and explored. The study suggested an automated robotic system that includes ambient condition monitoring, net and biofouling inspection, growth inhibition, and fish monitoring inside the cages for the purposes of biofouling detection and cleanup. In order to provide detailed instructions for the deployment of the robotic system for the aquaculture inspection task, that work presented specifications and requirements for the creation of such a system.

For the automatic inspection of fish net pens in underwater fish farms, the authors of the research offer a vision-based positioning and control technique (Akram et al., 2022). The plan uses conventional computer vision techniques for target location detection and combines stereo and monocular image design approaches for input data acquisition. The vehicle may move along the net plane thanks to the integration of the vision algorithm with a control module. Through testing in a real-world setting and simulation, the system's performance is assessed.

According to the study, the monocular image-based method works better than the stereo image-based method. The latter is strongly influenced by computing costs, feature extraction, and hardware design decisions. However, because it has fewer criteria and less computational cost, the monocular image-based approach shows to be better suited for practical applications.

The health of farmed fish in fish farming depends on the quality of the water. As a result, evaluating water quality is a topic of great interest in the context of fish farming. The authors of (Betancourt, Coral and Colorado, 2020) described a fish cage inspection system that incorporates both net status and water quality monitoring. On the network, various sensors were installed to track temperature, dissolved oxygen (DO), oxidation reduction potential (ORP), potential hydrogen (pH), and ORP. A Hough Transform approach was utilized to build the net mesh for the purpose of detecting net damage, and based on the incomplete net pattern, the damaged part was found in the camera image. Despite being tested in an experimental setting, the vehicle was manually operated. In a similar manner, the authors in (Cario et al., 2017) used acoustic Internet of Things networks to deploy hardware and software solutions, such as SeaModem for communication, HydroLab for water quality monitoring, and energy harvesting systems using propellers in underwater fish farms.

The fish cage net is subjected to yet another routine check in (Livanos et al., 2018). In this paper, a distance control technique for real-time video streaming-based net status assessment is given. The target location in this technique is a real object that is attached to the net. The target is then found in the image using computer vision techniques, such as the canny edge detector, at a set distance and angle. The vehicle is then given instructions to travel ahead or backward in the direction of the net plan using the target information. The described technique, while straightforward and cheap to implement, necessitates the attachment of preset target items to the net surface. The controller is also susceptible to noise and environmental disturbances.

Long-baseline and ultra-short baseline positioning techniques used in traditional positioning techniques call for predeployed and localized infrastructure, which raises the cost and operational complexity. However, in a dynamic context, laser and optical systems are quick to set up and effective solutions. In this regard, the authors of (Bjerkeng et al., 2021) proposed an autonomous examination of the fish cage net based on laser-camera triangulation. The concept behind this strategy is to display two parallel laser lines onto the network diagram. The lines were taken out of the photos using image processing methods, and the triangulation method was used to determine their placements. This strategy outperformed the DVL method in terms of results.However, this work does not take into account the underlying control issue and merely proposed the position estimation for the net tracking problem. The laser triangulation technique must be applied in a closed-loop environment using a tracking controller specifically created for tracking applications.

Another interesting and simpler approach is presented by (Sandøy et al., 2020). This article introduces the polar map, a novel 2.5D map representation with a memory consumption of O (ML), where M and L depend on the angular and depth resolutions of the map. The map's update and evaluation processes have a computational effort of O(1), and updates are carried out using Kalman filters. Only environmental structures that could be properly represented in a cylindrical coordinate system, such as fish cages and square and cylindrical tanks, can use the representation.

Chapter 2

Theoretical concepts

In this section, we will go through some of the concepts that will be required in order to apply the intended technique.

2.0.1 SLAM

SLAM stands for Simultaneous localization and mapping. This technology continuously builds and improves the map of the environment through exploration and localize in current pose in that environment. Sensory input for the SLAM can come from any sensors e.g. acoustic, DVL (Doppler velocity log), camera, Laser etc. If we use multiple sensors then we have to perform sensor fusion. Using different types of sensors is actually better as it reduces the individual uncertainties related to each sensor measurement. One point to note here is that this technology is considered as a standard in the mapping industry and performs much better than Artificial intelligence algorithm for this particular problem. Figure 2.1 shows the general flow from sensor data acquisition till map creation.

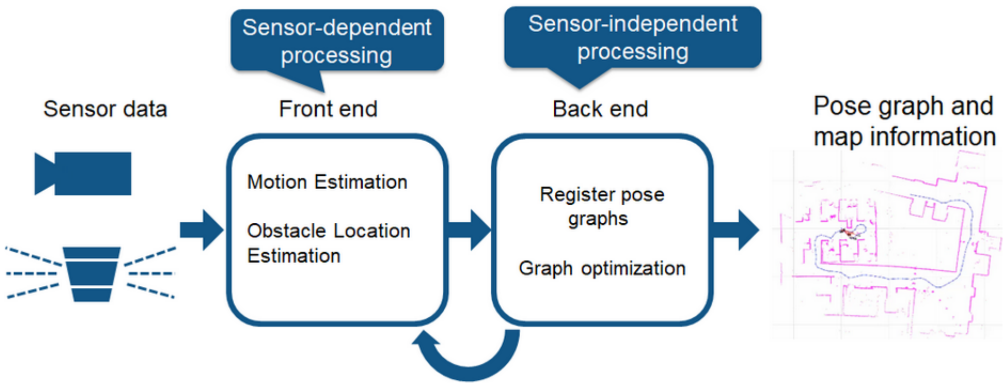

Sensor-dependent processing
Sensor-independent processing
Sensor data
Front end
Back end
Pose graph and map information
Motion Estimation
Obstacle Location Estimation
Register pose graphs
Graph optimization

Figure 2.1: SLAM process flow Figure taken from *What is SLAM* n.d.

Depending on the main sensor used, the SLAMs can be categorized in the following categories.

Visual SLAM

Visual SLAM is the type of SLAM where the sensory data is provided by the camera. It can be used as a fundamental technology for various types of applications and has been discussed in the field of computer vision, augmented reality, and robotics in the literature (Taketomi et al., 2017). Recently, SLAM using cameras only has been profoundly discussed because the sensor configuration is simple and the technical difficulties are higher than others. The basic structure of the Visual SLAM is as follows

- Initialization
- Tracking
- Mapping

The initialization step is about defining the 3D coordinate system and generating an initial map as an environment reconstruction. After this step, the tracking and mapping are continuously performed. During the tracking, the reconstructed map is tracked in the image to estimate the camera pose of the image with respect to the map. To perform this the correspondence between the two images is first obtained via a process called "Feature matching". Then the camera pose is estimated by solving the Prespective n Point problem **taketomi_uchiyama_ikeda_2017**.

In order to obtain a more stable vSlam we need to take care of two additional modules as well. It includes Relocalization and Global map optimization.
Relocalization is required when the tracking is failed due to fast camera motion or due to some disturbance. For that, it is necessary to compute the camera pose with respect to the map again.In general, the estimation error accumulates overtime depending on the camera position. To reduce errors, global map optimization is performed. This way, the map is refined by considering the consistency of whole map information.

Acoustic SLAM

The acoustic SLAM is technique that primarily relies on the acoustic sensor for the data. The topic of acoustic SLAM is gaining popularity in applications like home automation, teleconferencing, search-and-rescue robots, and Human-Robot Interaction (HRI) *What is SLAM* n.d.

Acoustic scene maps shows the Cartesian pose and trajectories of sound sources in the nearby environment. In order to obtain a scene map, instantaneous Directions-of-Arrival (DoAs) of sources are estimated using Sound Source Localization (SSL). Cartesian map feature positions are estimated over time from the DoAs by utilizing source tracking approaches.

LiDAR SLAM

Light detection and ranging (lidar) method normally uses the laser sensor. They are significantly more precise in comparison with Cameras and ToF (time of flight) sensors. They are generally used for applications with high-speed moving vehicles such as self-driving cars and drones. The output values from laser sensors is a point cloud data in either 2D or 3D space. The downside is that there are not commercially available for underwater applications as much as the other two types of sensors ibid.

2.0.2 Scale invariant feature transform (SIFT)

One needs features that can be traced from one image to the next in order to complete a loop in SLAM utilizing a camera. The Scale- Invariant Feature Transform is one of numerous well-established feature detectors (SIFT). There are four sequential phases in the SIFT method for extracting features:

- **Scale-space Extrema Detection:** SIFT analyzes the variation of the Gaussian (DoG) pyramid to find keypoints in a picture. The image is convolved with Gaussian filters at various scales, and the difference between neighboring scales is calculated to produce the DoG pyramid. In the DoG pyramid, keypoints are recognized as local extrema, designating areas with notable fluctuations in intensity.
- **Keypoint Localization:** Once possible interest sites have been identified, SIFT uses a method known as scale-space extremum localization to precisely pinpoint where each one is. To weed out subpar keypoints, it takes into account the contrast, stability, and curvature of the DoG responses at each keypoint.
- **Orientation Assignment:**In order to achieve invariance to picture rotation, SIFT computes a dominant orientation for each keypoint. It determines the keypoint's neighborhood's gradient magnitude and orientation, then chooses an orientation based on the distribution of gradient orientations. This step makes sure that the description consistently captures local image information in all orientations.
- **Calculation of the SIFT descriptor:** The SIFT descriptor is calculated by taking into account the gradient's strength and direction in the immediate area surrounding each keypoint. Each sub-region's gradient orientation histograms are generated when the region is partitioned into bins or subregions. The final descriptor vector, which captures the local picture structure and texture information, is created by concatenating these histograms.
- **Descriptor Description:** The local image region surrounding the keypoint is uniquely represented by the SIFT descriptor, which is a high-dimensional vector. Histogram equalization, a transformation, is used to normalize it and make it robust to changes in lighting. The descriptor is resistant to changes in lighting thanks to this normalization.

Due to the SIFT descriptor's robustness and invariance characteristics, it has been widely used in numerous computer vision applications. It offers a distinct depiction of the image's important elements, enabling accurate matching and identification at various scales and angles.

2.0.3 Speeded Up Robust Features(SURF)

One of the commonly used feature descriptor used in computer vision for picture matching and identification tasks is called SURF (Speeded Up Robust Features). It was created to be reliable and effective at locating and matching image keypoints(Bay, Tuytelaars and Van Gool, 2006). The following are the main steps in the SURF descriptor:

- **Scale-Space Extrema Detection:** SURF finds stable and recognizable interest spots at various scales to locate key points in an image. In order to do this, it is necessary to examine the difference of Gaussian (DoG) pyramid, which identifies keypoints at various sizes.
- **Keypoint Localization:** SURF uses a method known as the Hessian matrix to localize keypoints with sub-pixel precision after prospective interest points are discovered. This stage increases the keypoint localization's accuracy and guarantees noise and picture transformation resistance.
- To achieve invariance to image rotation, SURF computes a dominant orientation for each keypoint. It determines the keypoint's neighborhood's gradient magnitude and orientation, then chooses an orientation based on the distribution of gradient orientations.
- **Descriptor Calculation:** The SURF descriptor is calculated as a collection of multi-scale oriented patches located around each keypoint. These patches are represented as a set of gradient orientations or intensity values, which are combined to form a reliable and concise description. The Haar wavelet response, a wavelet-based approximation, is used by SURF to quickly and effectively compute the descriptor.
- **Descriptor Description:** The local picture region in and around the keypoint is uniquely represented by the SURF descriptor, which is a vector. It provides a reliable representation that is invariant to changes in size, rotation, and partial occlusions by encoding information about the intensity or gradient fluctuations within the region.

Due to its computational effectiveness and resistance to multiple image alterations, the SURF descriptor has grown in prominence. It is frequently employed in tasks like 3D reconstruction, image stitching, and object detection.

2.0.4 Oriented FAST and Rotated BRIEF (ORB)

For applications like object detection and tracking, the ORB (Oriented FAST and Rotated BRIEF) descriptor is a well-liked feature descriptor in computer vision. The BRIEF (Binary Robust Independent Elementary Features) descriptor and the FAST (Features from Accelerated Segment Test) keypoint detector are combined in this method. These are the primary steps in the ORB descriptor:

- **Keypoint Detection:** The FAST method is used by ORB to find keypoints in a picture. Based on changes in intensity, FAST finds corners and features. It looks at a neighborhood of pixels in a circle and finds keypoints when a large enough number of pixels have intensities that diverge significantly from the centre pixel.

- **Keypoint Orientation Assignment:** After keypoints are found, ORB gives each keypoint an orientation. In this stage, the neighborhood of the keypoint's intensity gradient is calculated, and the dominant orientation is chosen based on the gradient directions. Rotational invariance in the descriptor is made possible by the specified orientation.
- **Feature Description:** The local image patch surrounding each keypoint is represented by ORB using the BRIEF descriptor. By comparing pixel intensities at designated spots within the patch, BRIEF creates a binary string. The binary descriptor is created by these comparisons, which result in a pattern of 0s and 1s.
- **Rotational Alignment:** ORB aligns the descriptions with the allocated keypoint orientations to enhance rotational invariance. Regardless of the orientation of the keypoint, this alignment enables the descriptor to consistently capture picture information.
- **Descriptor Matching:** Because ORB descriptors are binary, they are commonly compared using the Hamming distance. By counting the number of bits that differ between two descriptors, the Hamming distance calculates how similar two descriptors are for matching.

Due to its binary nature, the ORB descriptor is renowned for being effective and efficient in handling real-time applications. Fast keypoint detection and binary descriptor calculation are combined, resulting in a computationally efficient system that performs well in matching and recognition tasks.

2.0.5 Binary Robust Invariant Scalable Keypoints(BRISK)

For applications like picture matching and identification, the BRISK (Binary Robust Invariant Scalable Keypoints) descriptor is a feature descriptor used in computer vision. It is renowned for being effective and resilient to scale and rotation variations. The following are the key steps in the BRISK descriptor:

- **Keypoint Detection:** To find keypoints in a picture, BRISK uses a variation of the FAST (Features from Accelerated Segment Test) algorithm. FAST uses intensity differences to locate corners and features, and BRISK expands on this idea to find scale-invariant keypoints.
- **Scale-space Construction:** BRISK builds a scale pyramid by blurring the input image with Gaussian noise at various scales. By creating a sequence of photos with varying degrees of blurring, this approach enables BRISK to identify keypoints at various scales.
- **Keypoint Orientation Assignment:** After keypoints are found, BRISK gives each keypoint a specific orientation. It determines the prevailing orientation based on the gradient orientations present in the keypoint's vicinity and computes the intensity centroid of that area. To achieve rotational invariance, perform the orientation assignment step.

- **Descriptor Computation:**BRISK calculates the description by randomly selecting pixel pairs from a circle centered on each keypoint. These pixel pairs' intensities are compared, and a binary pattern is created, with the bit values determined by the intensity comparisons. The BRISK descriptor is created from the resulting binary string.
- **Descriptor Description:** The local image data for the keypoint is represented by the binary string known as the BRISK descriptor. Both the relationships between pixel pairs and the specific qualities of the keypoint's surroundings are captured. The descriptor's binary form makes it computationally effective and appropriate for real-time applications.

The BRISK description strikes a compromise between effectiveness and resistance to rotations and scale changes. It is appropriate for applications that call for quick and accurate feature matching since it offers a binary representation of keypoints that can be matched using Hamming distance.

2.0.6 Accelerated-KAZE(AKAZE)

The AKAZE (Accelerated-KAZE) descriptor is a computer vision feature descriptor that is specifically made to handle difficult situations including scale changes, rotations, and picture noise. It delivers improved speed and efficiency and is an expansion of the KAZE (KAZE Features) description. The following are the primary steps in the AKAZE descriptor:

- **Scale-Space Construction:** AKAZE uses a number of nonlinear filtering processes to create a nonlinear scale space representation of the image. This method aids in identifying keypoints at various scales and maintaining the details of their immediate surroundings.
- **Feature Detection:** Scale-invariant keypoints in the image are found using nonlinear diffusion filtering using AKAZE's feature detection algorithm. By evaluating areas with notable intensity changes, it pinpoints keypoints. AKAZE can handle picture alterations and noise reliably thanks to this method.
- **Keypoint Orientation Assignment:** After keypoints are found, AKAZE gives each keypoint a specific orientation. This is accomplished by identifying the dominant direction of the gradient information surrounding the keypoint. Rotational invariance in the descriptor is made possible by the specified orientation.
- **Calculation of the Descriptor:** AKAZE calculates the descriptor by sampling the gradient and intensity data in the immediate vicinity of each keypoint. It builds multiscale representations of intensity and gradient values and creates a feature vector to reflect the specific properties of the keypoint.
- **Descriptor Description:** The local image data for the keypoint is represented by the floating-point vector known as the AKAZE descriptor. It provides a rich representation that is resistant to scale changes, rotations, and image noise since it captures both intensity and gradient information.
- **Keypoint Orientation Assignment:** After keypoints are found, AKAZE gives

each keypoint a specific orientation. This is accomplished by identifying the dominant direction of the gradient information surrounding the keypoint.

Compared to its predecessor, KAZE, the AKAZE descriptor performs better in terms of speed, robustness, and efficiency. It has been extensively employed, especially in settings with difficult conditions, in a variety of computer vision applications like picture matching, object detection, and 3D reconstruction.

2.0.7 Tracking

During any SLAM, tracking is one of the most crucial component. Tracking can generally be broken down to three levels short term, mid term and long term. All of these three tracking are being perform simultaneously
Short term tracking
Short term tracking is about the pose estimation. It requires high tracking rate and precision. One of the opportunities is that the next pose is going to be almost identical to the previous one and we can significantly restrict the correspondences.
Mid term tracking
It is about tracking after the significant motion of the ROV and still requires relatively high tracking rate. The opportunity is that we don‘t need to run in the key-frame rate.
Long term tracking
Long term tracking is about tracking after a significant amonunt of time and we a a lot of the viewpoints to compare with. Occlusions are one of challenges but we can run it at even slower rates.

2.0.8 Loop closure

The loop closure is one of the biggest challenges of the SLAM problem. It happens when you revisits the area and you want to update your whole map. The newer map will be more accurate as you have two readings of the same area, this will be back projected till the beginning. The inter-image metric of global saliency was used to assess uniqueness Straume Haugland, 2021. The global saliency is utilized to find distinctive images that can be used for extensive loop-closure. Utilizing inverse document frequency, it is possible to assess a keyframe’s uniqueness in terms of its properties (IDF). Less frequent features will be given more weight by the IDF algorithm, making them simpler to find. When the two images are matched the map could be updated by adjusting for the DVL(doppler velocity log) acoustic sensor value. This way the loop closure will produce more accurate results.

2.0.9 Bag of words(BOW)

A bag-of-words format becomes one of the most popular ways to represent image content (Y. Zhang, Jin and Zhou, 2010) and has been effectively used for object categorization. It was inspired by the success of text categorization. The first step in a conventional bag-of-words representation is to identify "interesting" local patches from a picture, either through dense sampling or an interest point detector. The key points are these small, localized areas that are represented by vectors in a high-dimensional space.

In this project an open source C++ bag of words (DBoW3) ('DBoW3 DBoW3' 2017) library is used which itself is based on OpenCV. It indexes and convert the image into the Bag of Word representation.

Following are some of the key steps taken in order to implement the bag of words algorithm

- **Tokenization:** The text corpus is broken down into discrete tokens, such as words. This procedure entails eliminating punctuation, changing the text's case to lowercase, and separating the content into individual words.
- **Vocabulary construction:**The process of building a vocabulary or dictionary involves gathering all distinct words or tokens from the corpus. An individual index or identification is given to each word.
- **Document-Term Matrix:** A matrix is created, where the rows are the documents and the columns are the vocabulary words. Zeros are initially placed in the matrix.
- **Word Frequency Count:** After analyzing each document in the corpus, the frequency of each word is recorded. The document-term matrix's counts, which reflect the frequency of words in each document, are then updated.
- **Vectorization:** Based on the word frequencies, each document is converted into a numerical vector representation. In the document-term matrix, the vector is commonly represented by a row, with each element denoting the frequency of a specific word in the text.
- **Feature scaling:** To normalize word frequencies and provide more weight to significant and discriminative terms, feature scaling techniques like term frequency-inverse document frequency (TF-IDF) may also be used.

By treating text documents as numerical feature vectors, the resulting Bag of Words representation enables the application of various machine learning techniques for tasks like classification, clustering, and information retrieval.

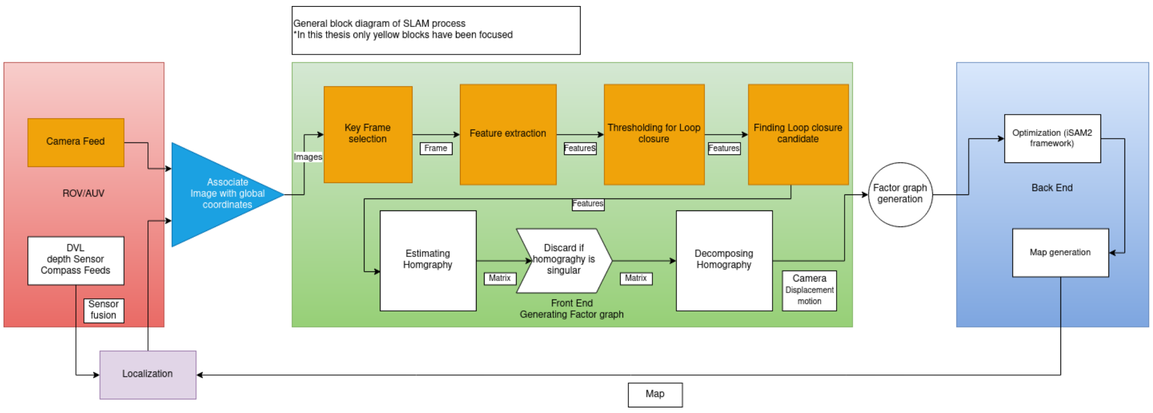
Chapter 3

Methodology

One of the main target of the project was to write a common firmware for all the descriptor. That was a bit challenging due to their own dependencies and library requirements. A joint framework is created in the end in which the user can select any of the descriptors and any of the tests.

3.0.1 Block diagram

The Figure 3.1 represents the complete block diagram of the SLAM but we are focusing only the place recognition part of it. Selecting a proper place recognition method is very crucial for better performance



General block diagram of SLAM process
*In this thesis only yellow blocks have been focused
Camera Feed
ROV/AUV
DVL
depth Sensor
Compass Feeds
Sensor fusion
Associate Image with global coordinates
Images
Key Frame selection
Frame
Feature extraction
Features
Thresholding for Loop closure
Features
Finding Loop closure candidate
Features
Estimating Homgraphy
Matrix
Discard if homogragphy is singular
Matrix
Decomposing Homography
Camera Displacement motion
Front End
Generating Factor graph
Factor graph generation
Optimization (iSAM2 framework)
Back End
Map generation
Localization
Map

Figure 3.1: Block diagram

3.0.2 Flow chart

The Figure 3.2 represents the flow diagram of the algorithm.

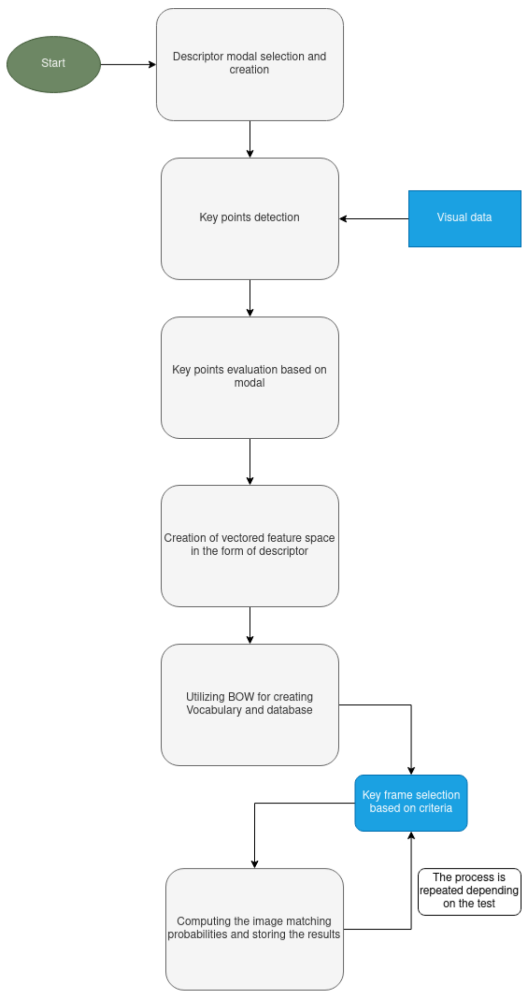

Start
Descriptor modal selection and creation
Key points detection
Visual data
Key points evaluation based on modal
Creation of vectored feature space in the form of descriptor
Utilizing BOW for creating Vocabulary and database
Key frame selection based on criteria
The process is repeated depending on the test
Computing the image matching probabilities and storing the results

Figure 3.2: Order of operation

More details of the steps are as under

- An extractor model is created based on the choice of descriptors
- Key points are extracted from the image and the regions of interest were obtained in pixel coordinates
- Key points are evaluated by descriptor specific criteria and then transformed into vector representation
- Bag of words (BOW) technique is used to create simplified vectored representation of feature space
- A vocabulary and database is created based on the extracted features
- A testing scenario is defined e.g. to test short term tracking, maximum feature extraction etc
- Feature space of images is compared, matching probabilities computed and results registered in .csv files
- Results are then visualized by a separate python program using Pandas and

matplotlib

- CMAKE and GCC compiler are used to build the project and get executable

3.0.3 Aquaculture net cage dataset

The Aquaculture data set is obtained from the opensource net inspection video data provided by the Deep Trekker Inc. . The data includes the shots from variety of distance. Some portions are very clear some are blurry. The impacts of this can be seen by the number of feature extracted plots at different parts of the video. The frames are extracted from the video using the sampling frequency of 0.1 Hertz. There were a total of 1000 continuous pictures, but the pictures are spatially discontinuous at one point during the video. The main drawback of this dataset is that it does not have enough frames to check for long term tracking i.e the area that is visited once is not visited again. So we cant find loop closures. That is one of the reason that we had to also add another underwater dataset.

3.0.4 Girona cave dataset

This dataset (Mallios et al., 2017) was collected in an underwater cave complex using an autonomous underwater vehicle. The purpose of the study was to map both the horizontal and vertical surfaces of the caves. To achieve this, the vehicle was equipped with various sensors, including two mechanically scanned imaging sonar sensors, a Doppler velocity log, two inertial measurement units, a depth sensor, and a vertically mounted camera. The camera was used to capture images of the ocean floor for ground truth validation at specific locations. The data collection was carried out in July 2013, with a human diver guiding the testbed to ensure safe navigation in the challenging underwater environment. The study provides the original robot operating system bag files for convenience, as well as a combined version of picture and text files that can be processed on different platforms for analysis.

During the scene there are 5 different cones that were visited and then again revisited while recording the scene. In total there are 10000 consecutive frames and they are continuous in both space and time.

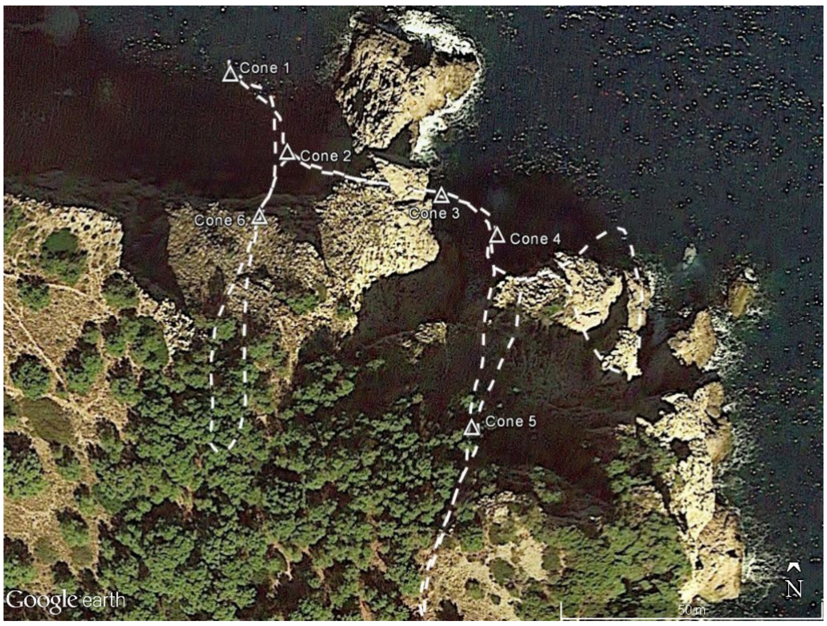


Figure 3.3: ROV trajectory for data collection taken from underwater vision and robotics lab website

Chapter 4

Results and Discussion

This portion we will be discussing in detail the performance of the descriptors under real world environment.

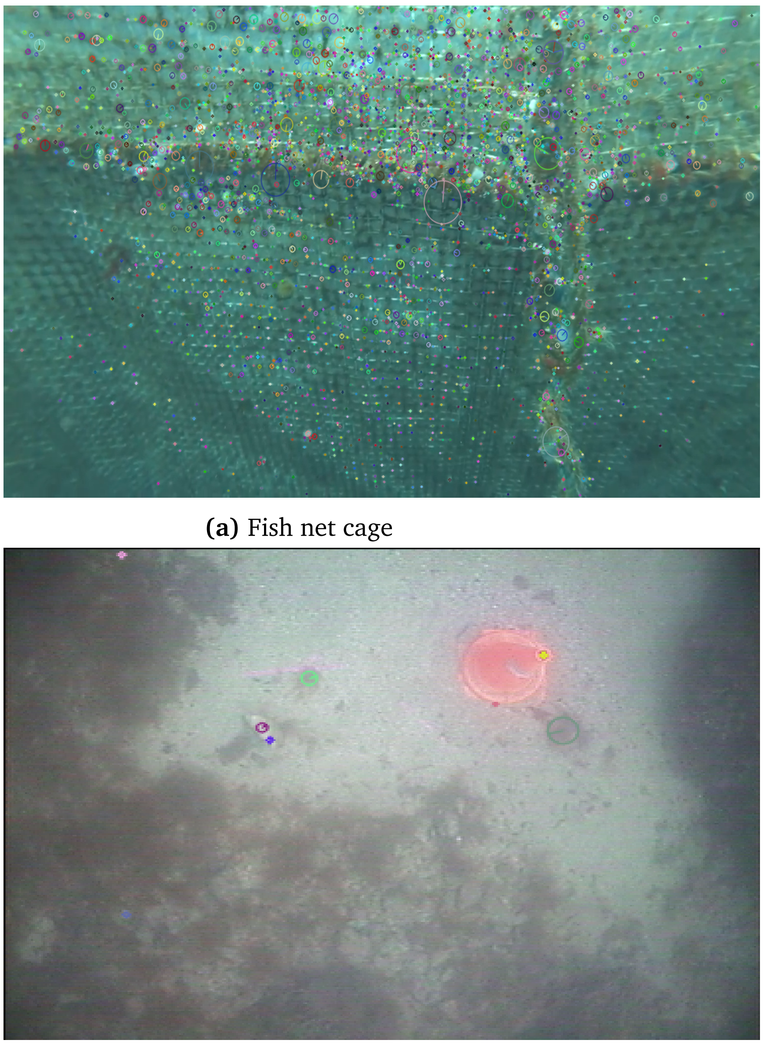
4.1 Feature extraction

4.1.1 Keypoints and regions of interest

Given that the five descriptors employed in this research each have their own feature extraction pipeline, it becomes crucial to evaluate their performance under identical conditions. This section will examine how various descriptors were capable of extracting features from the same set of images. A keypoint is defined by a circle in the image. The diameter represents the region of interest with the particular orientation based on the direction of the gradient.

SIFT

SIFT performed good as well in the aquaculture environment as shown in the Figure 4.1a. Even though, SIFT struggled in the cave environment to extract noticeable features. This can be attributed to its low blur rejection behaviour.



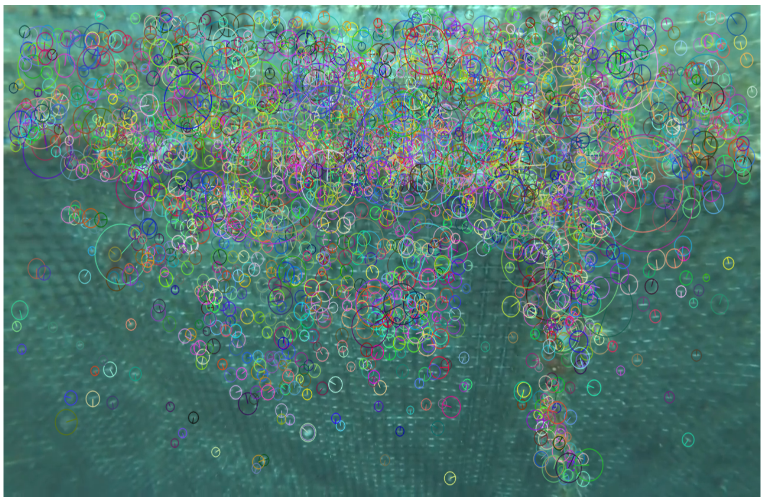
(a) Fish net cage

(b) Girona cave

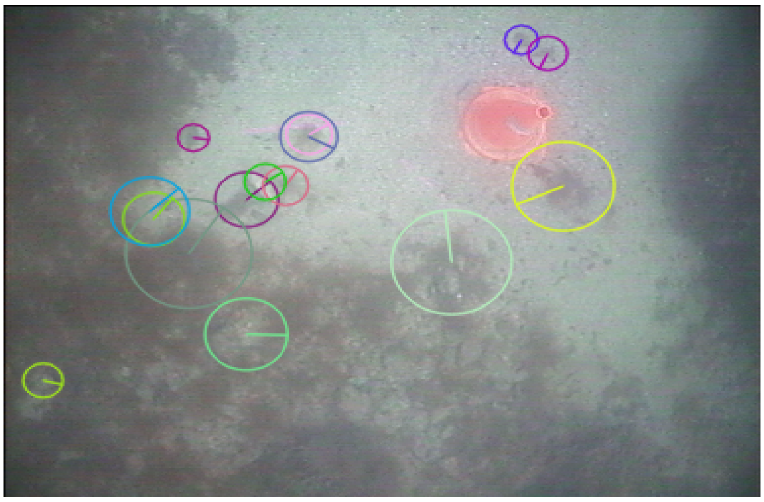
Figure 4.1: SIFT key points extraction

SURF

SURF was the only descriptor to perform really good in both the environments, see Figure 4.2. The keypoints extracted in the cave environment were quite big that could introduce the false positive while tracking.



(a) Fish net cage

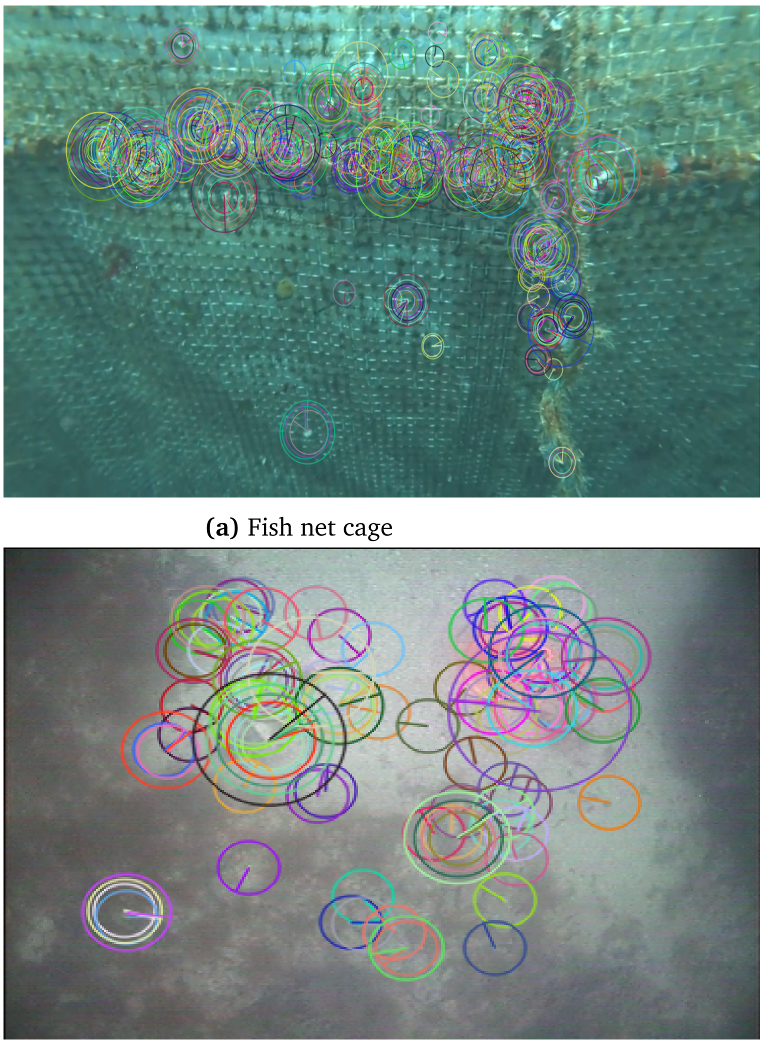


(b) Girona cave

Figure 4.2: SURF key points extraction

ORB

ORB performance turned out to be the opposite of SIFT. It extracted enough features in the underwater cave, see Figure 4.3 but at the same time it was not able to detect features outside of the big plus in the aquaculture environment.

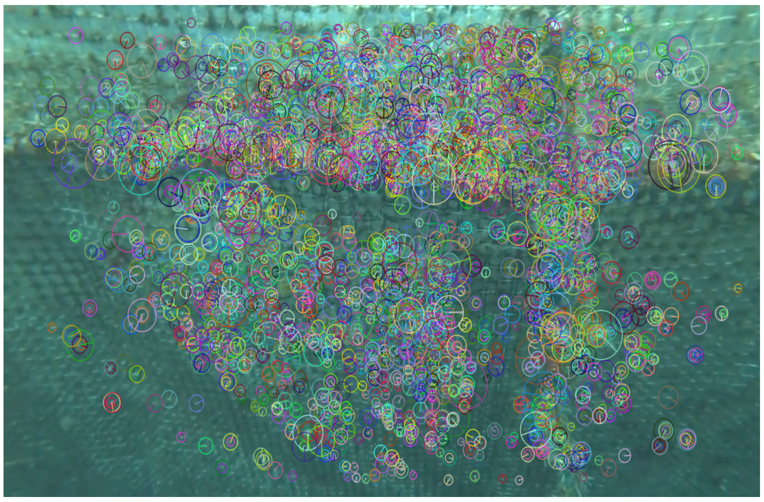

(a) Fish net cage

(b) Girona cave

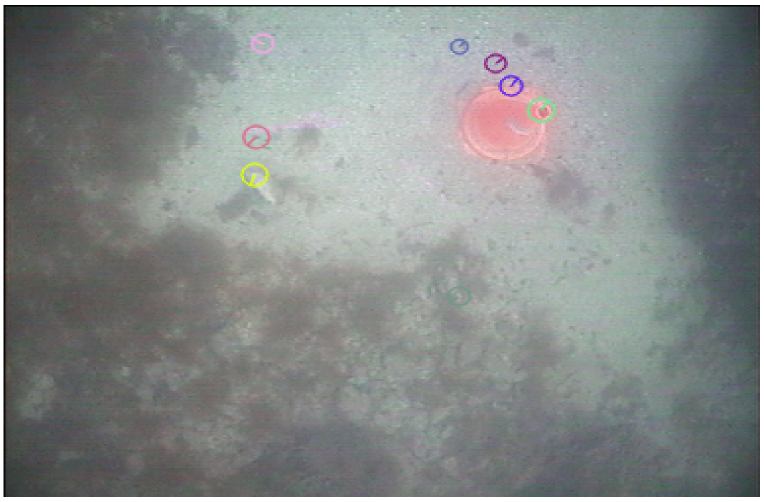
Figure 4.3: ORB key points extraction

BRISK

BRISK also performed well as far as the feature extraction in concerned. The area of keypoints is better than the AKAZE especially on the big plus sign due to higher gradients in that area. It did not performed that good in the underwater cave environment.



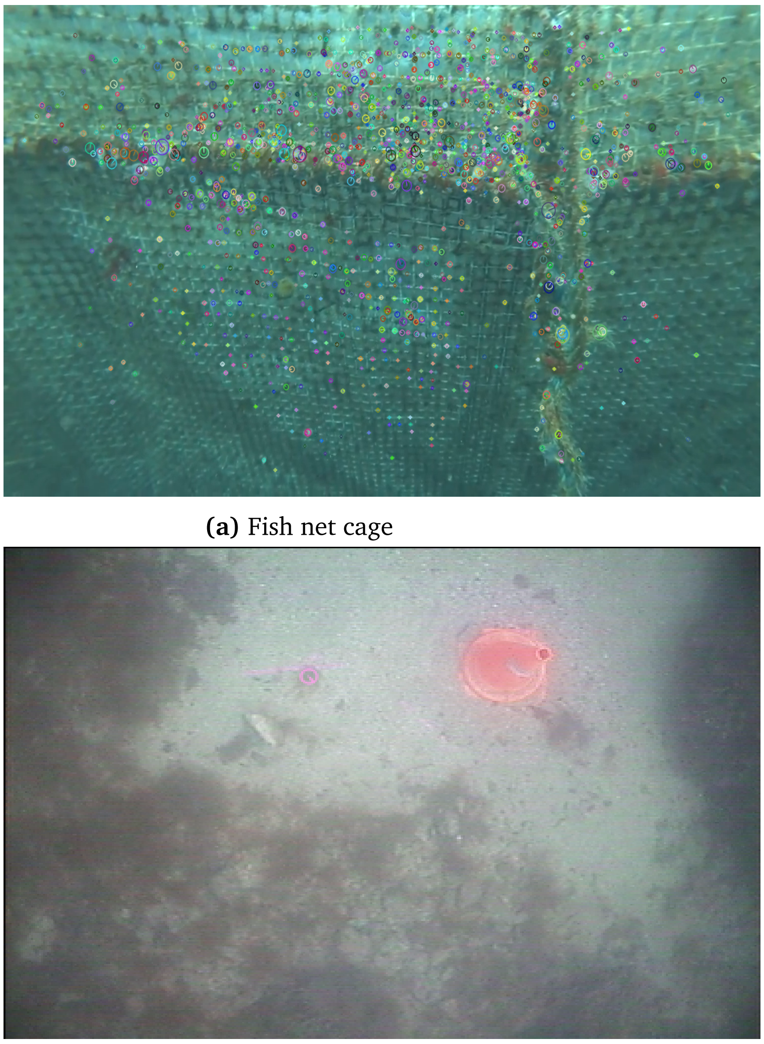
(a) Fish net cage



(b) Girona cave

Figure 4.4: BRISK key points extraction

Akaze


(a) Fish net cage

(b) Girona cave

Figure 4.5: AKAZE key points extraction

AKAZE successfully detected a considerable quantity of key points within the net cage setting. However, a majority of these points had a small diameter, indicating a limited region of interest. Consequently, tracking can become challenging in the presence of substantial movement.
AKAZE exhibited poor performance in the underwater cave environment, managing to extract only one key point, as depicted in Figure 4.5b. This outcome can be attributed to AKAZE's reliance on high-quality textures for effective feature extraction (Alcantarilla, Bartoli and Davison, 2011), which was lacking in this particular setting.

4.1.2 Feature extraction by whole episode

The purpose of this experiment was to have an overall quantitative picture of the amount of keypoints extracted by each descriptor. It is the continuation of previous experiment which focused more on the lower level understanding of the keypoints extraction.

Aquaculture fish net cage

As you can see in the Figure 4.6, the number of keypoints or features extracted varies over the course of the whole episode. This is because of the clarity of the scene varies through out. It can be explained by marking some alphabetical stages

- A: Beginning of the scene
- B: Drastic scene change (Discontinuity)
- C: More close and clear view
- D: Even more close and clear
- E: A bit far but still good
- F: End of scene

You can see that SIFT was able to extract most amount of features followed by BRISK, SURF, ORB and AKAZE. One thing to note here is that extracting higher number of features does not automatically gives the higher distinguishing power. And we will see in upcoming results that BRISK even extarcting considerable key-points were not able to perform well when it comes to tracking.

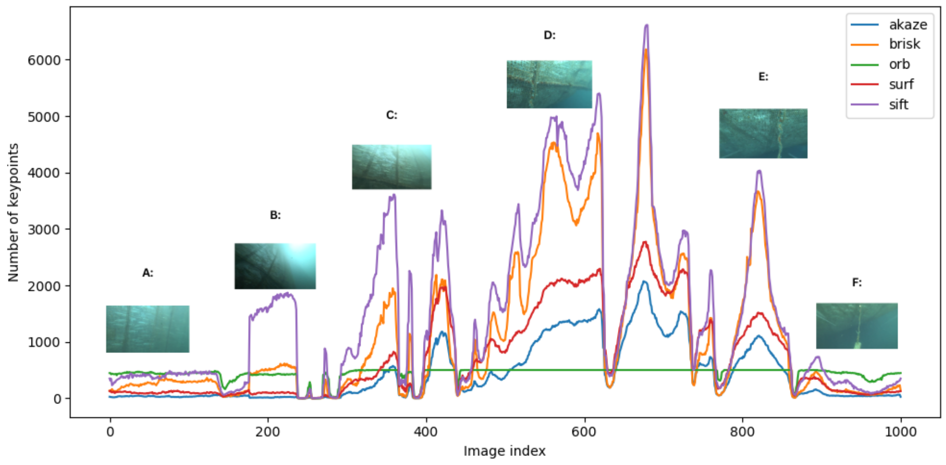


Figure 4.6: Number of feature extracted per frame

Girona underwater cave

We can see a similar trend in the under water cave environment as well and it can be explained the same way as above. One thing in particular to note here is that SIFT which was the leader in the feature extraction in net cage environment fell down to third behind BRISK and ORB.

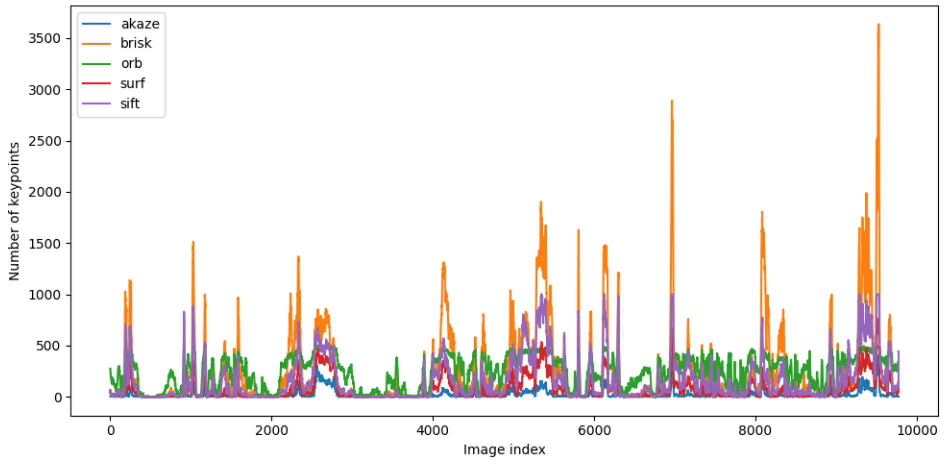


Figure 4.7: Number of feature extracted per frame

4.2 Short term tracking for Aquaculture fish net cage

As the camera passes through the scene, we have overlaps between consecutive frames. The short term tracking is test designed to track a keyframe with the upcoming frames untill they fall below the matching threshold criteria. We have performed the testing based on two thresholds one is 30 % and the other 10 %. It is intuitive that 10 % will be able to track longer before falling below threshold.

4.2.1 Key Frames tracking overall

As it can be seen in Figure 4.8, Only SURF and SIFT were able to track more than 10 frames. SIFT was leading the number of keypoints extracted but SURF was able to do well with fewer keypoints. It implied that the keypoints extracted by the SURF was of higher value. It is noted that the SURF performed better than SIFT in the blurry situation but SURF also did a false positive in the end i.e. even though there was no overlap between the images it continued to track it while SIFT was very consistent it did no false positive. In the plot x axis is the frame index from 0 to 1000 and y-axis is the amount of frames tracked before the keyframe is changed. The following histograms will also help us understand these results better.

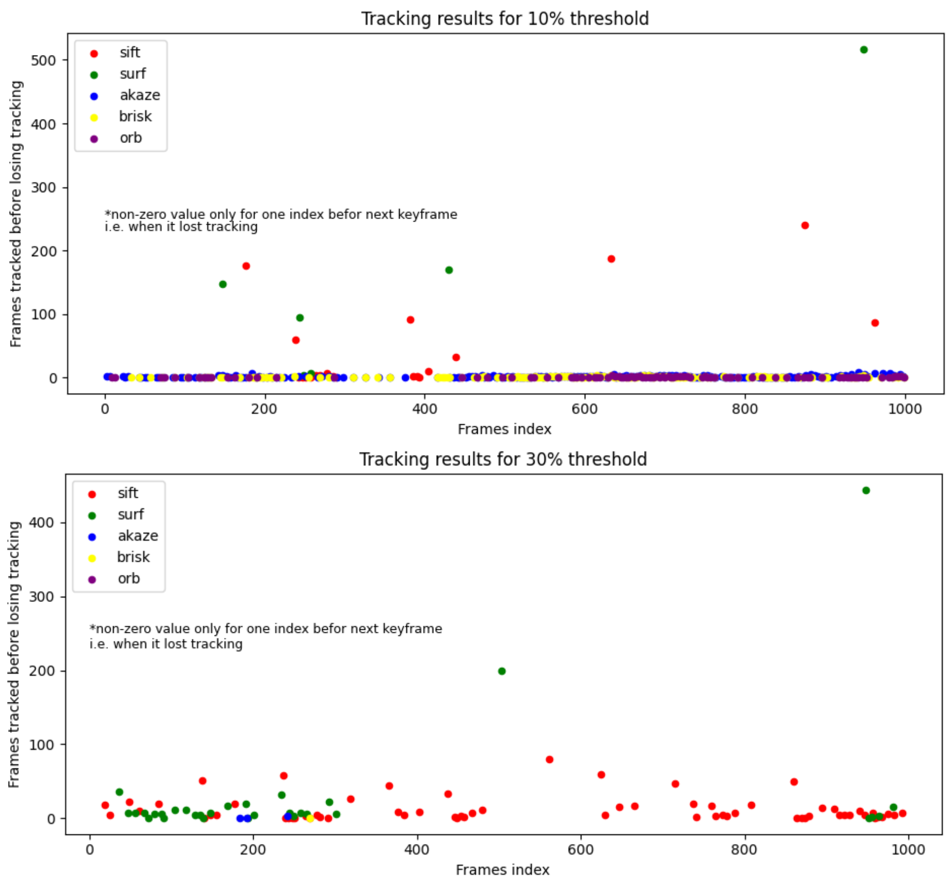
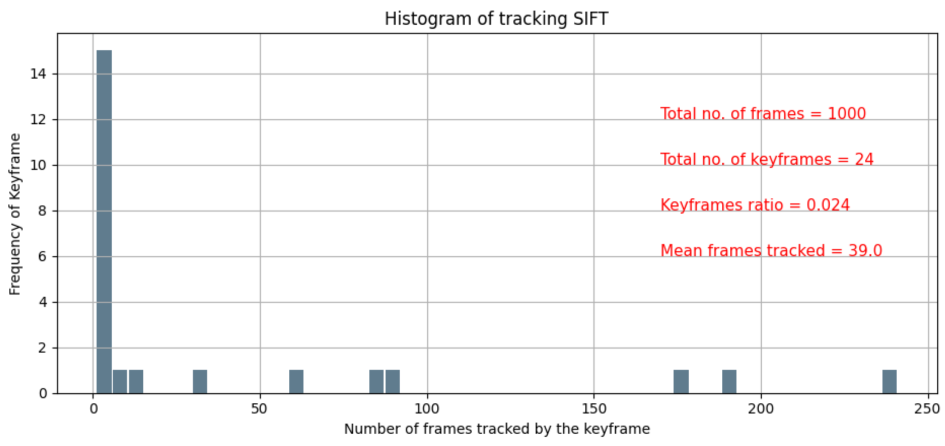


Figure 4.8: Tracking results of aquaculture dataset

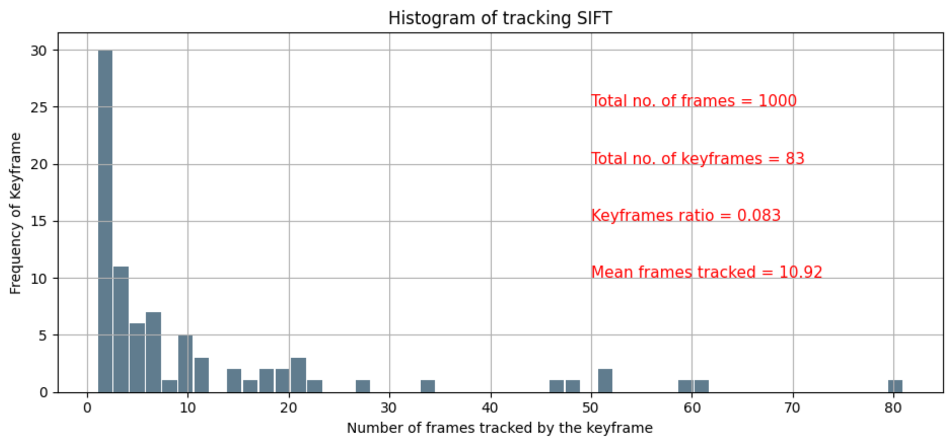
4.2.2 Histograms of tracking

Histogram provides us a nice representation of the important results. In the x-axis we have the number of frames tracked by any keyframe and on the y-axis we have the frequency of key frames. E.g. we read the first big bar that is at 1. It shows that there were 15 keyframes that were able to track only 1 next frame and then they lost tracking. SIFT with 10% threshold depicts the actual situation nicely, while SURF over does the thing.

SIFT



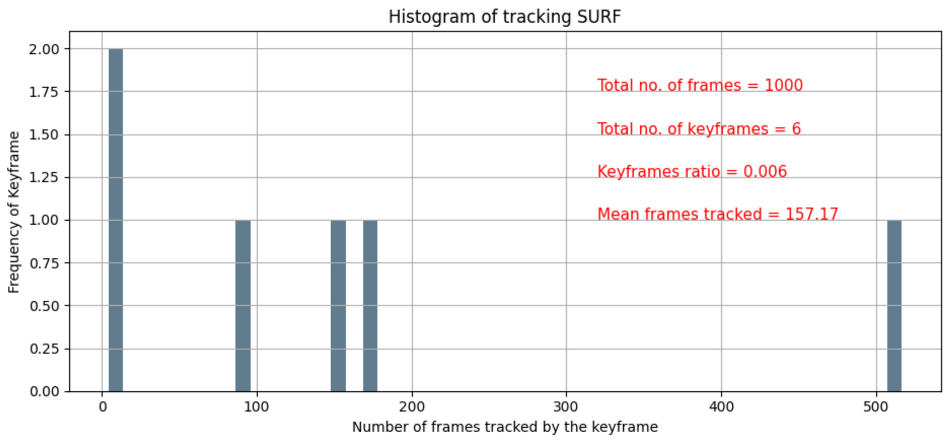
(a) 10 % threshold



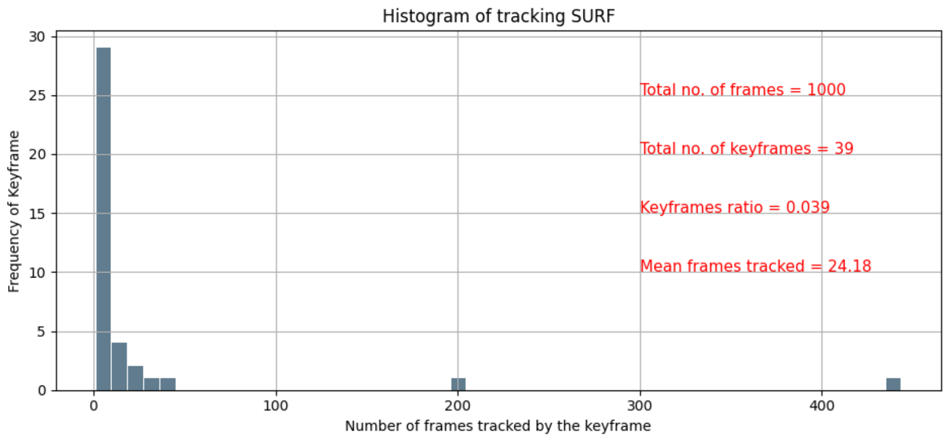
(b) 30 % threshold

Figure 4.9: SIFT fish net cage histogram

SURF



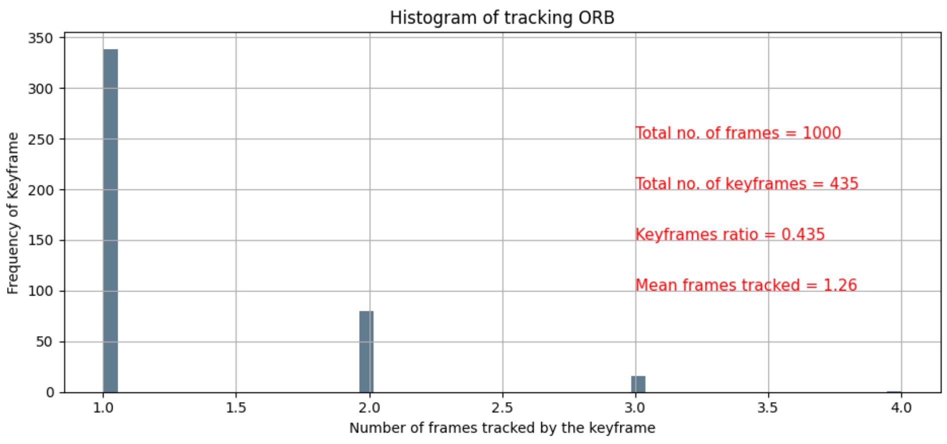
(a) 10 % threshold



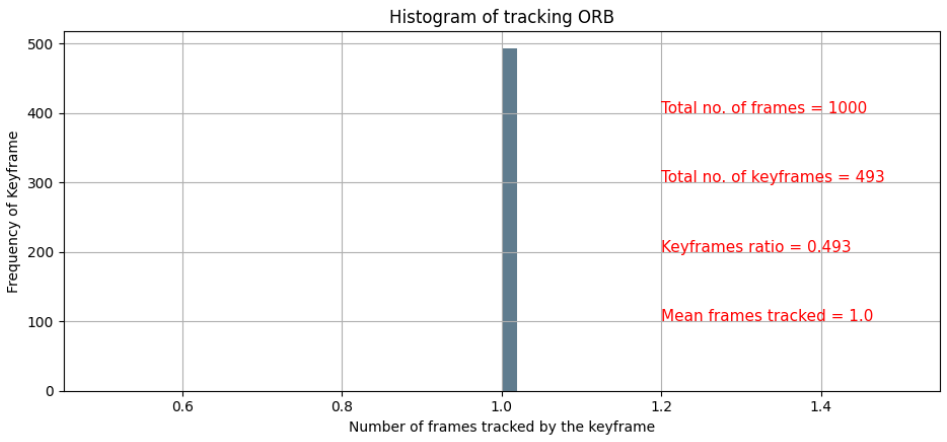
(b) 30 % threshold

Figure 4.10: SURF fish net cage histogram

ORB



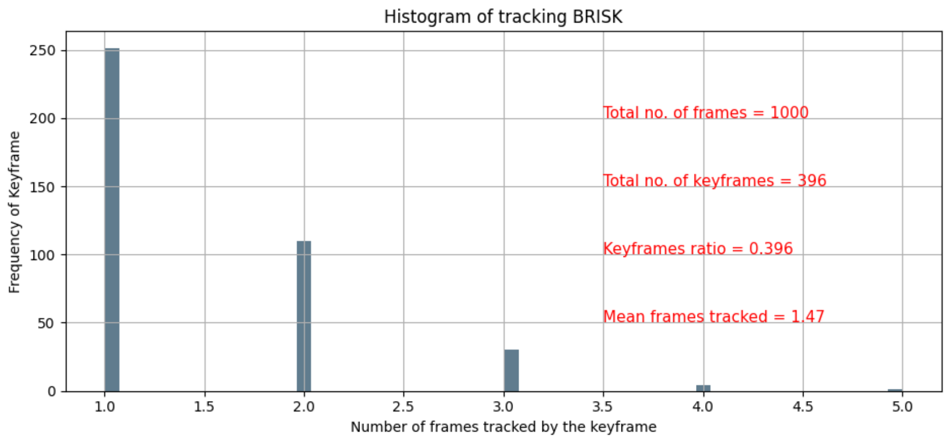
(a) 10 % threshold



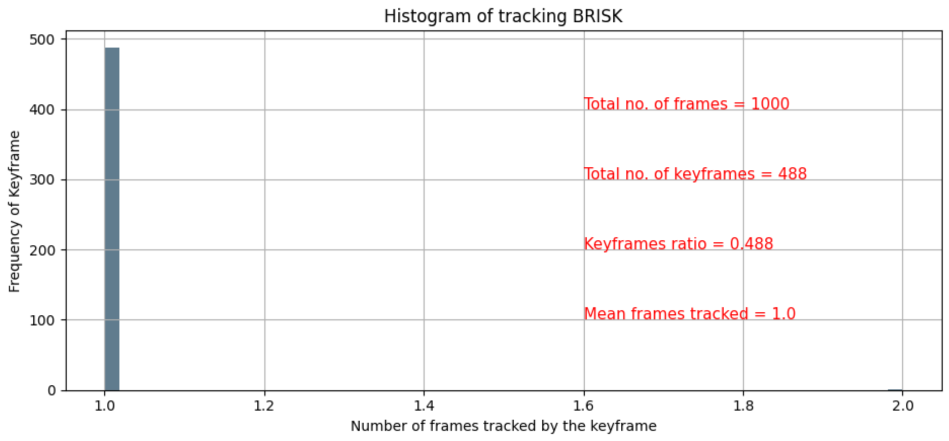
(b) 30 % threshold

Figure 4.11: ORB fish net cage histogram

BRISK



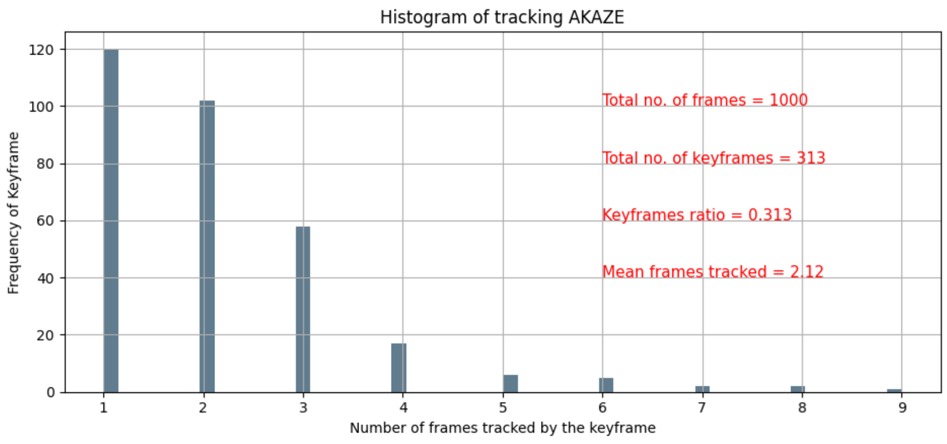
(a) 10 % threshold



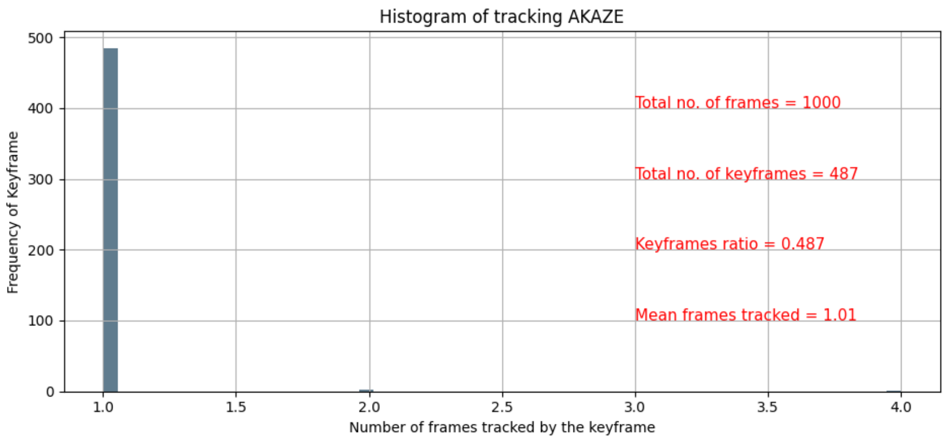
(b) 30 % threshold

Figure 4.12: BRISK fish net cage histogram

AKAZE



(a) 10 % threshold



(b) 30 % threshold

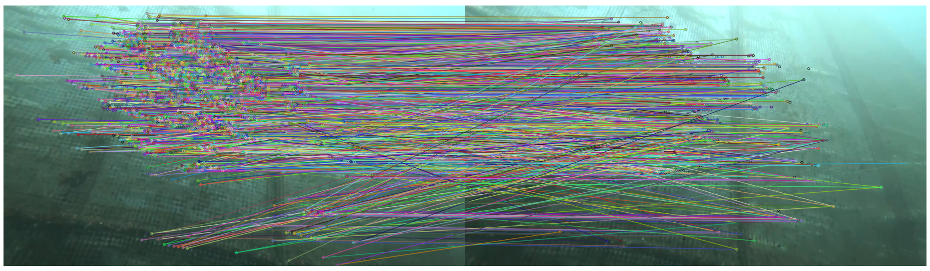
Figure 4.13: AKAZE fish net cage histogram

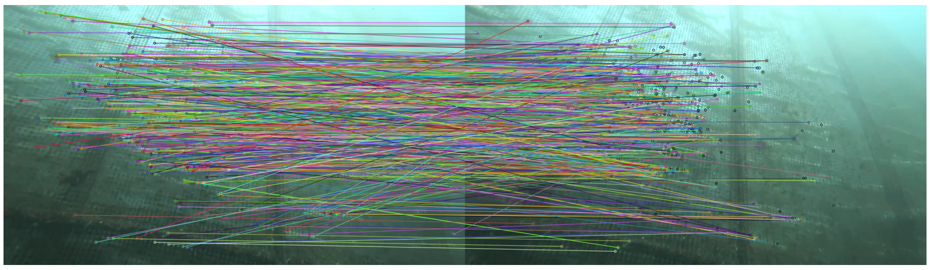
4.2.3 Comments

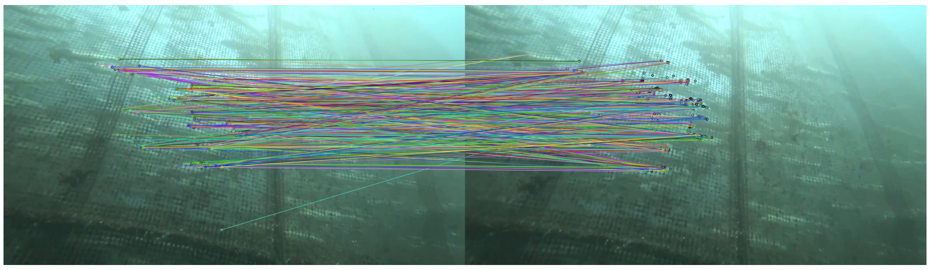
After looking at the histograms it is evident that SIFT is the closest to the ground truth, while surf did really well in the blurry condition but also produced a false positive result. AKAZE comes at the third position in short term tracking for the fish net cage but it‘s a long way off then the first two. ORB and BRISK performed very badly in these conditions.

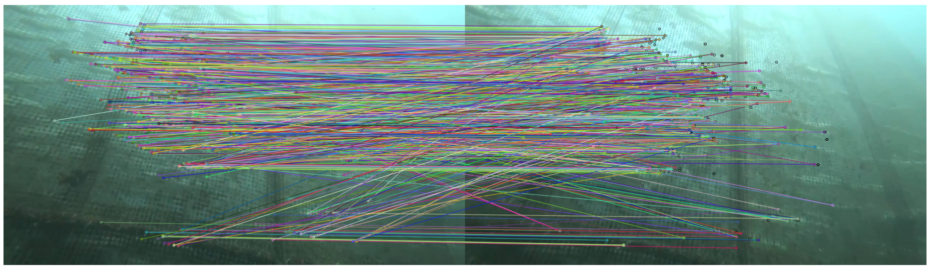
4.2.4 Feature matching

Figure 4.14 depicts how the feature from frame are mapped on the other after a small movement by the ROV. Since there is not a significant movement so most of the features should by mapped in the straight line. The black keypoints are those keypoints that are not found in the other frame. Ideally there number should be lower if the frames are close spatially.


(a) SIFT feature matching


(b) SURF feature matching


(c) ORB feature matching


(d) BRISK feature matching

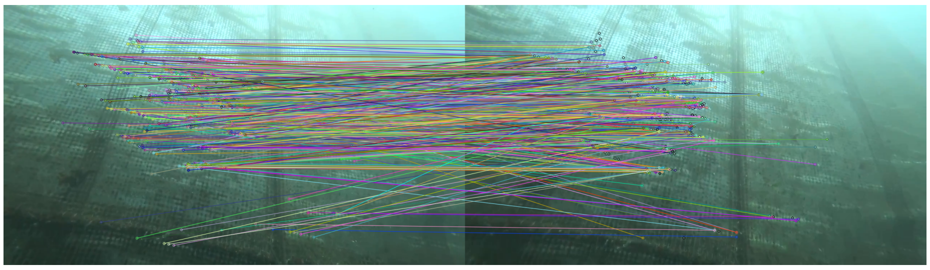

(e) AKAZE feature matching

Figure 4.14: Feature matching after small movement

4.3 Short term tracking for Girona underwater cave

The same tests are also being applied on the Girona underwater cave dataset in order to see how much results can change with another underwater environment.

4.3.1 Key Frames tracking overall

We can see a similar trend in short term tracking Figure 4.15 as was the case with fish net cage data sets. SURF and SIFT dominates the plots while AKAZE, ORB and BRISK falls behind. Surprisingly ORB falls behind even with having extracted a significant amount of features. SURF produces many false positives again like at one point it tracks up to 800 frames but in reality it should not be more then 60-70 frames. SIFT also produces some false positives but they can be easily controlled by increasing the threshold. Threshold is an important parameter and it should be optimized based on the datasets we are dealing with.

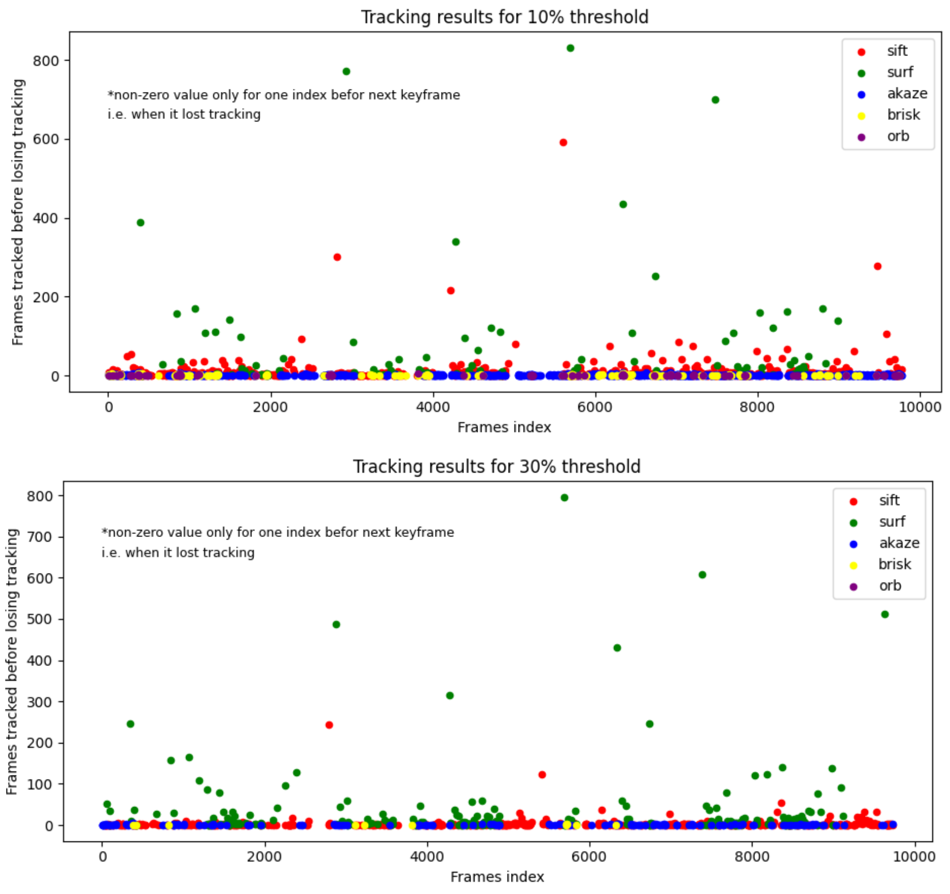
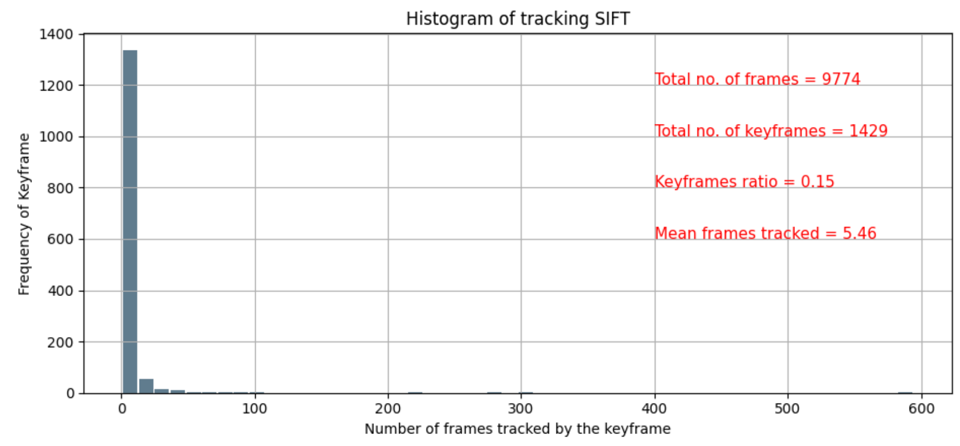


Figure 4.15: Tracking results of underwater cave dataset

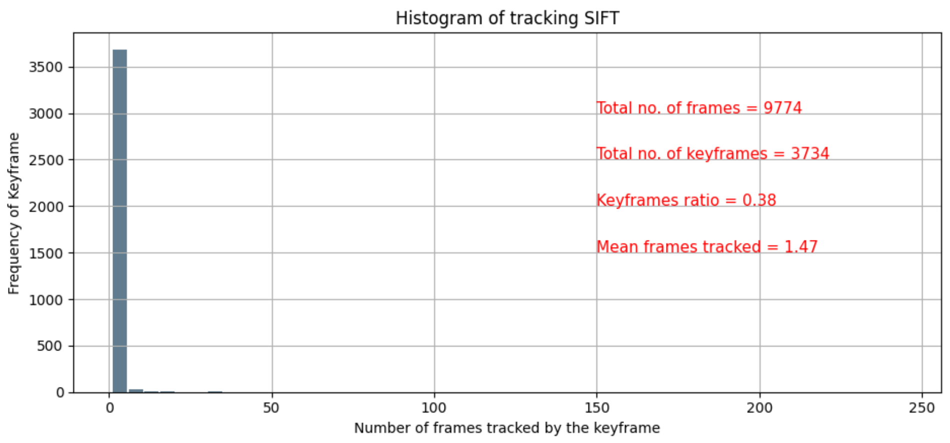
4.3.2 Histograms of tracking

We plotted the histograms again to have a better understanding of the performance. The results are consistent with the previous results. The way to read the graph has been explained in the previous histograms of tracking section for fish net cage.

SIFT



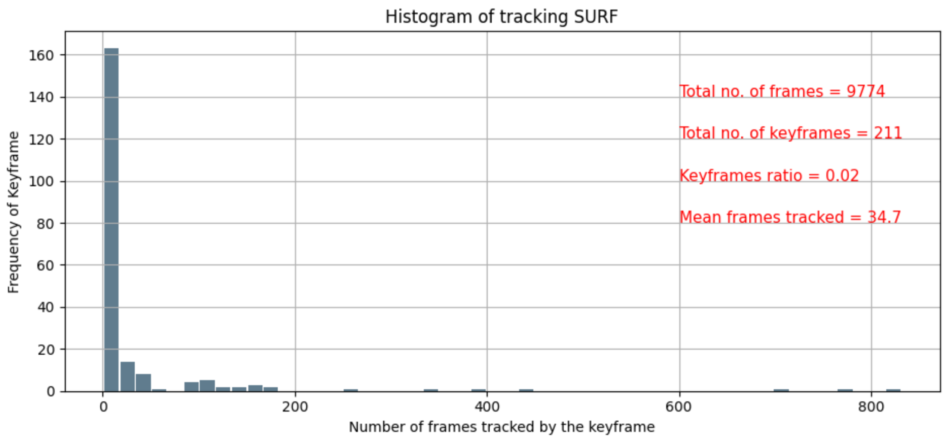
(a) 10 % threshold



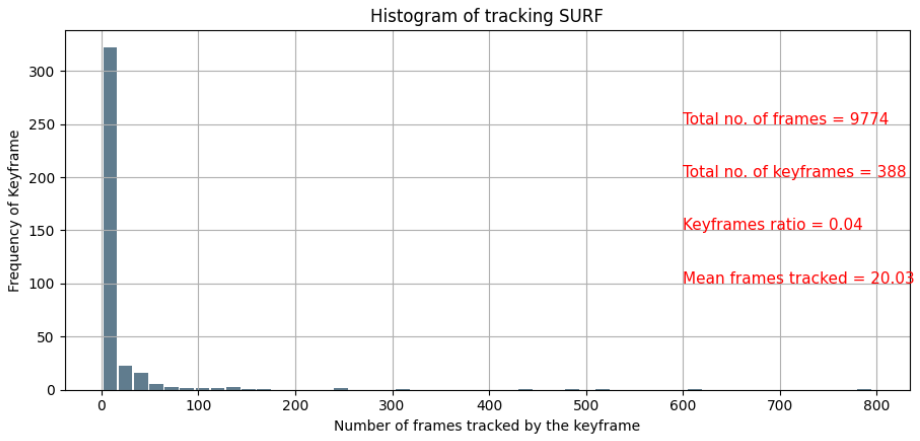
(b) 30 % threshold

Figure 4.16: SIFT underwater cave histogram

SURF



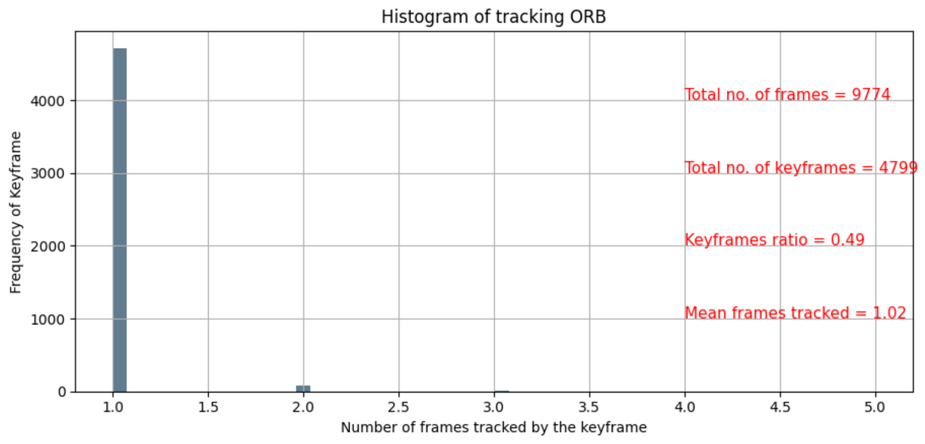
(a) 10 % threshold



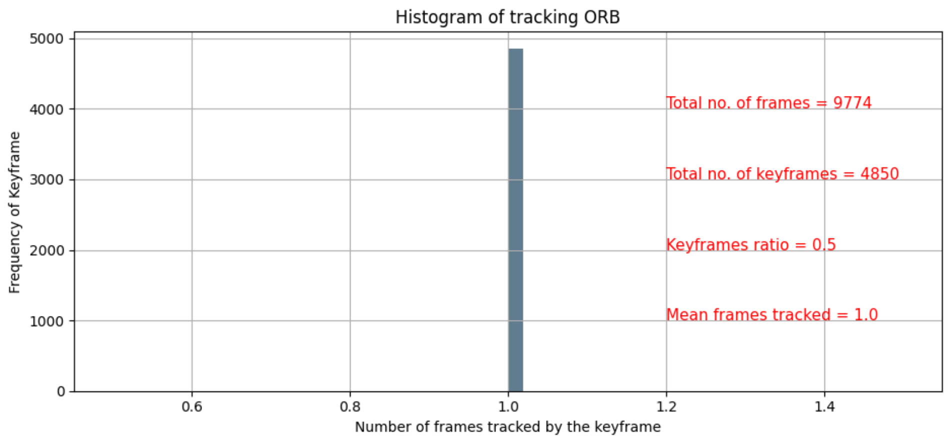
(b) 30 % threshold

Figure 4.17: SURF underwater cave histogram

ORB



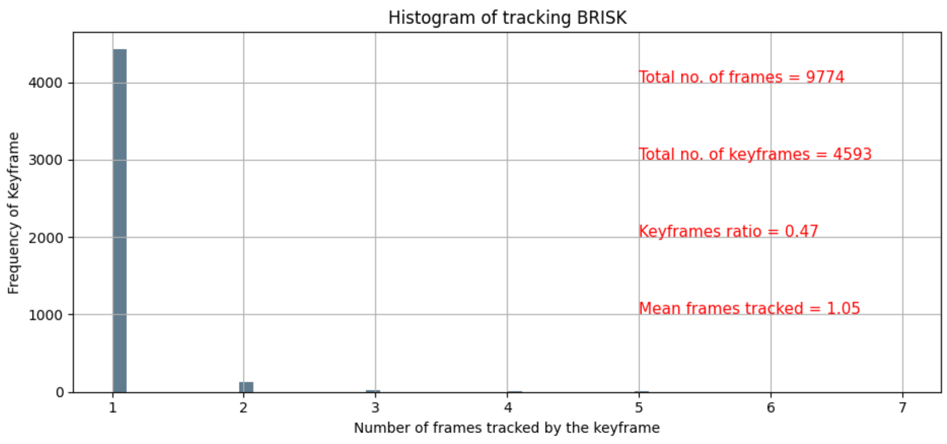
(a) 10 % threshold



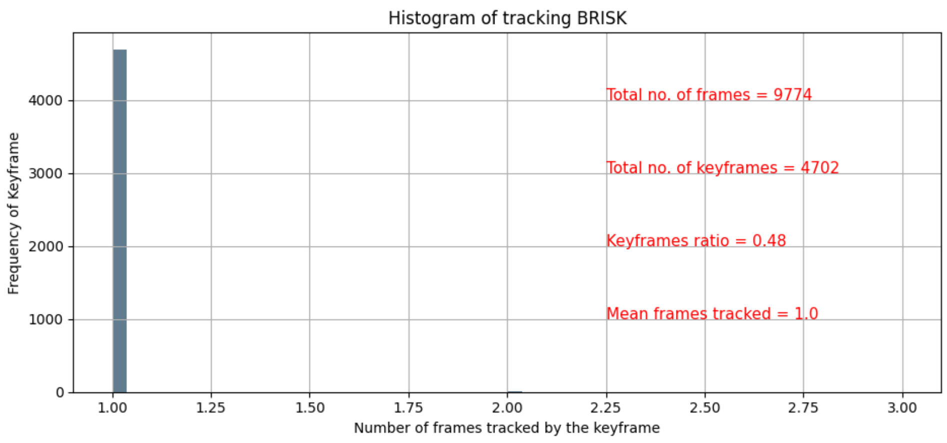
(b) 30 % threshold

Figure 4.18: ORB underwater cave histogram

BRISK



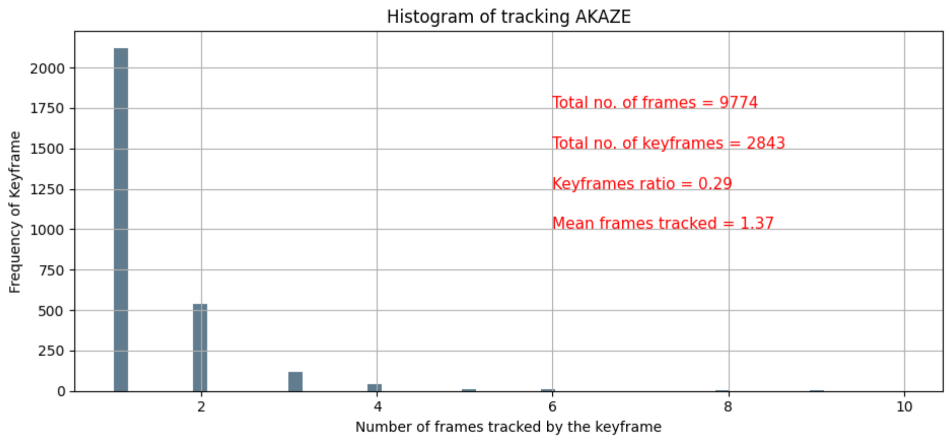
(a) 10 % threshold



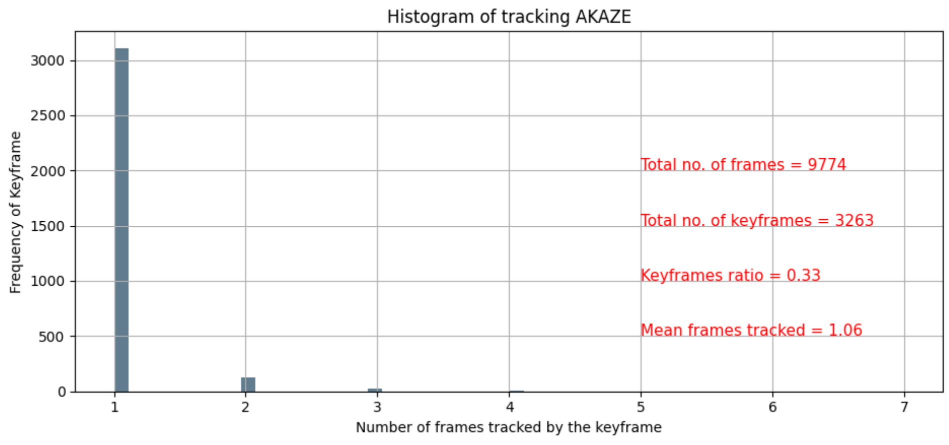
(b) 30 % threshold

Figure 4.19: BRISK underwater cave histogram

AKAZE



(a) 10 % threshold



(b) 30 % threshold

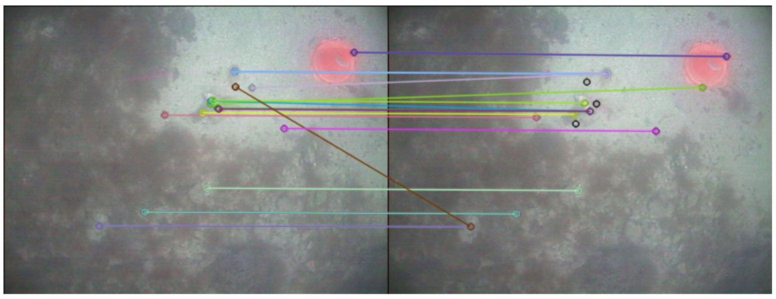
Figure 4.20: AKAZE underwater cave histogram

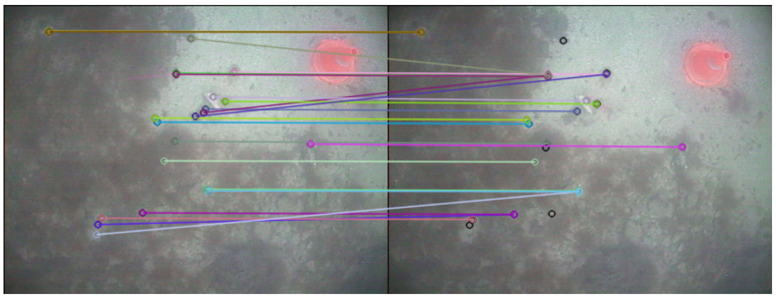
4.3.3 Comments

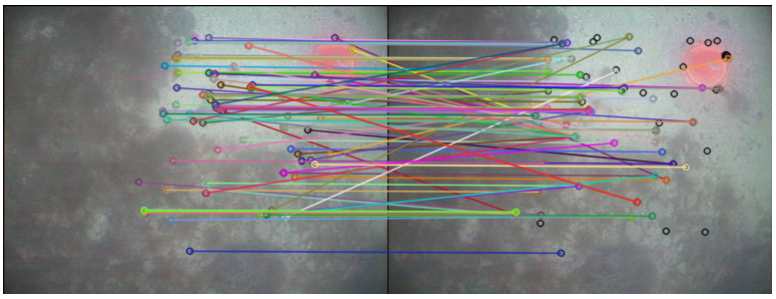
Histograms made it quite clear that despite of the change of datasets SIFT and SURF still performed better than the ORB, AKAZE, and BRISK.

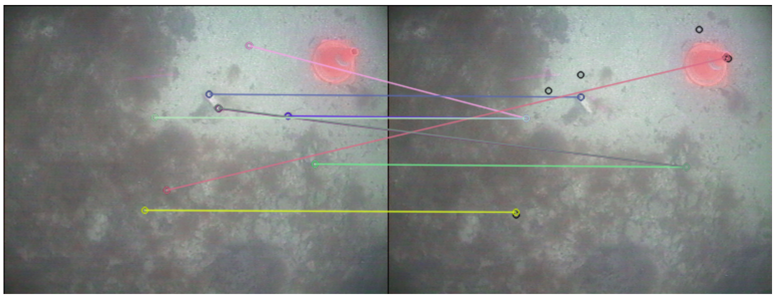
4.3.4 Feature matching

The underwater cave environment does not have many significant gradients and the frames are of lower quality as compared to the previous dataset. That results in less features extraction. With less features extracted we can visualize the the feature matching in a better way. The scenario remains the same with the small If see Figure 4.21a for the SIFT, you can see most of the features have been accurately mapped and there are only three unmapped features. Same is almost true in the case of SURF, but SURF could not detect any feature on the cone that‘s a downside. In Figure 4.21c for the ORB, you will see more unmapped features and more incorrectly mapped features (that results in the low score). BRISK also did wrong matching mostly. AKAZE could only find one feature in both the frames but it mapped accurately, that‘s why we say AKAZE performing better then ORB and BRISK. But it‘s low feature extraction overall brings it down.


(a) SIFT feature matching


(b) SURF feature matching


(c) ORB feature matching


(d) BRISK feature matching

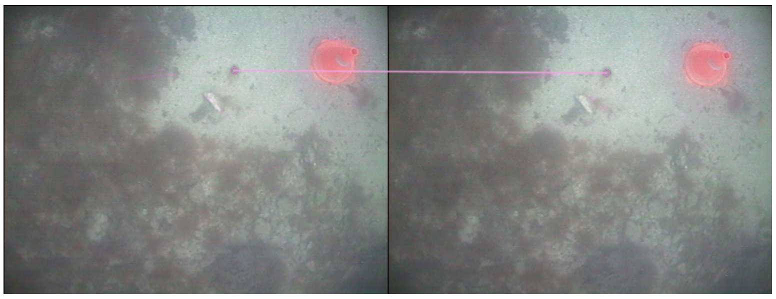

(e) AKAZE feature matching

Figure 4.21: Feature matching after small movement

4.4 Long term tracking for Girona underwater cave

The underwater cave dataset has five different cones located in different position to train the SLAM algorithm. We are not running a full SLAM but we selected one keyframe with cone 1 in the beginning and compared it with all the frames to see if we get a spike in scores when the area is revisited. One thing to note is when it is revisited its a different orientation and placement. So some features may not be tracked.

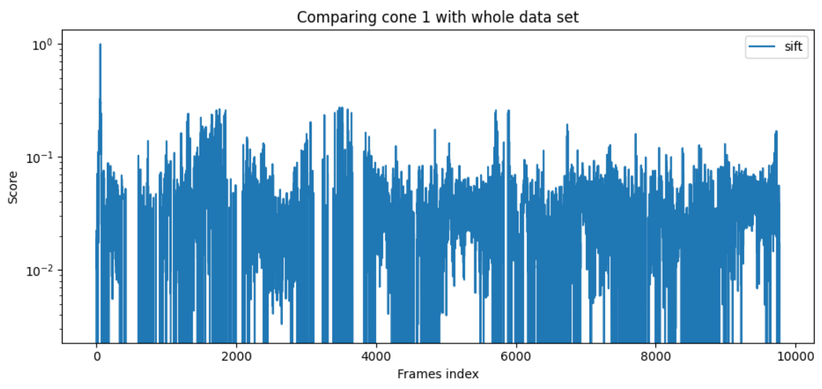


Figure 4.22: SIFT cone 1 loop closure search

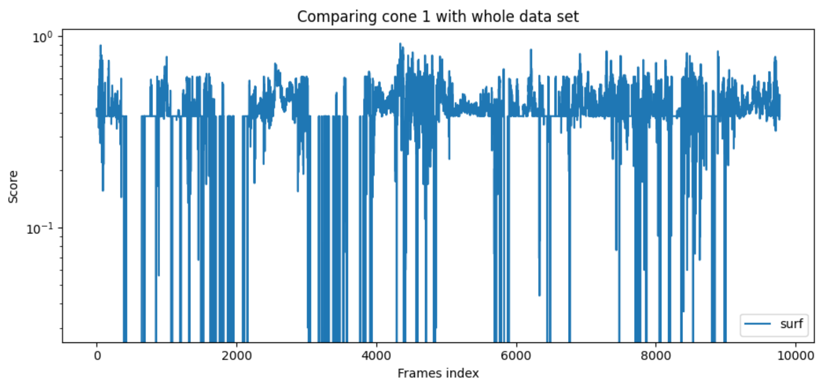


Figure 4.23: SURF cone 1 loop closure search

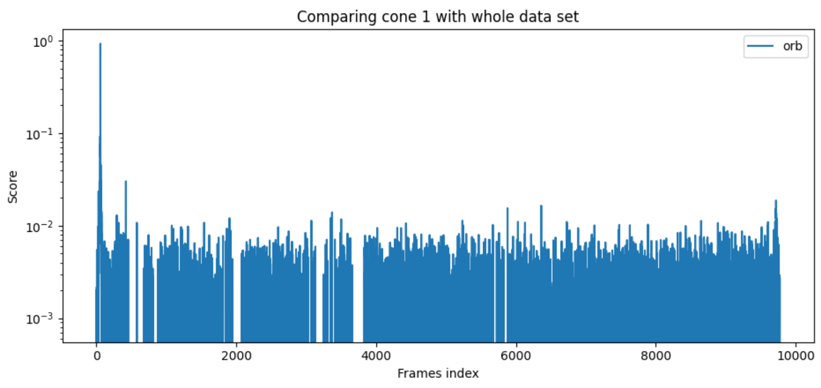


Figure 4.24: ORB cone 1 loop closure search

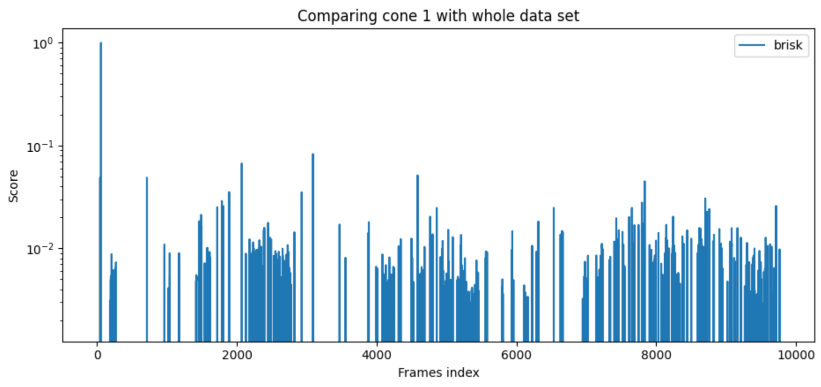


Figure 4.25: BRISK cone 1 loop closure search

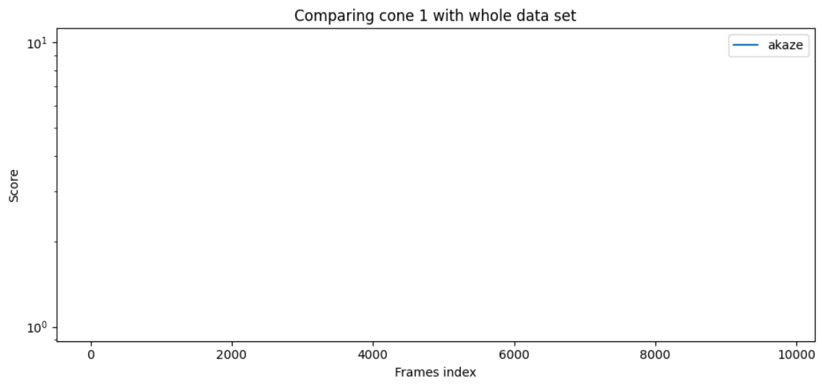
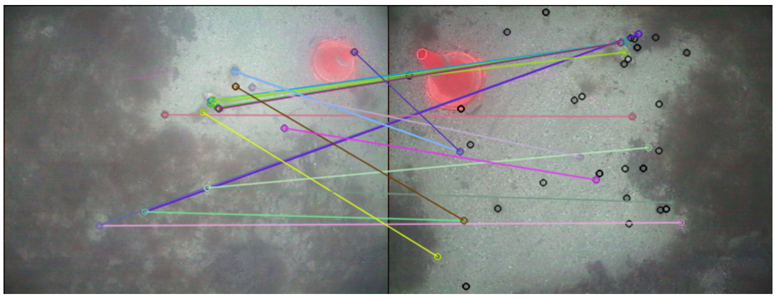


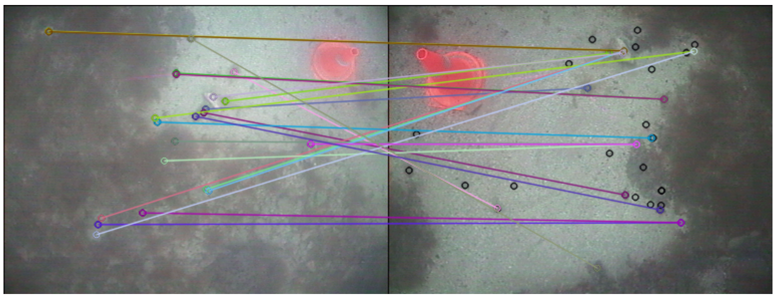
Figure 4.26: AKAZE cone 1 loop closure search

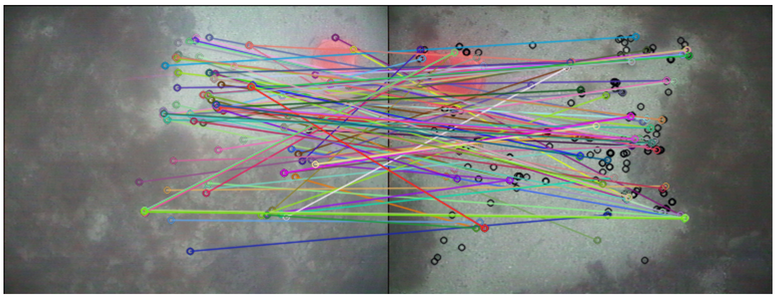
We can see the long term tracking results of all the descriptors from Figure 4.23 to Figure 4.26. We can see many ups and down throughout the episode. This is due to the fact that the terrain looks quite similar in many places and their are different but exactly same looking 5 cones as well that are revisited too. The good thing is that all the descriptor except the AKAZE (which could not detect any single feature to begin with) were able to detect a peak in the end when the area of cone 1 is visited again. SO if we can apply spatial constraint which are very usual (Straume Haugland, 2021), we can easily find a local maxima and find the loop closure candidate.

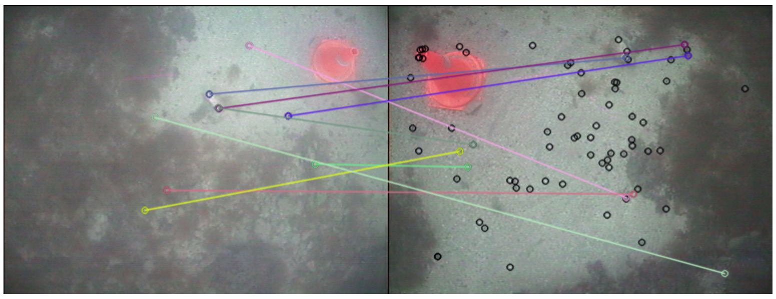
4.4.1 Feature matching

We performed this test to visualize the feature matching between the first and the revisited frame. Because of different pose of camera and the scale of image, we can find many many keypoints being missed or wrongly mapped. But we say some regions being correctly mapped as well e.g the white stone in the top right of Figure 4.27a is being correctly mapped by SIFT, SURF, and BRISK. ORB also got some features right. AKAZE is an exception, it only had one feature again and successfully mapped it even with a reverse orientation.


(a) SIFT feature matching


(b) SURF feature matching


(c) ORB feature matching


(d) BRISK feature matching

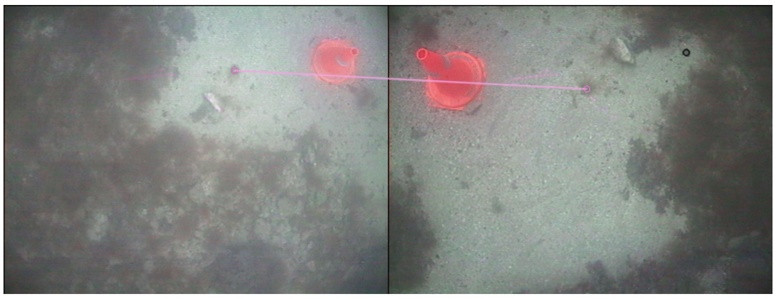

(e) AKAZE feature matching

Figure 4.27: Feature matching after re-visiting the area

4.5 Timing evaluation

In the last experiment we measured the time of some important operations for both datasets. This is an important aspect to consider in opting for a particular descriptor if you want t perform real-time operations.

	SIFT	**SURF**	**ORB**	**BRISK**	**AKAZE**
Mean detection time	227192	95652.80	17636.20	33834	174324
Mean transformation time	5882.45	4142.81	917.35	4137.39	1427.65
Mean comparison time	27.19	0.07	39.31	200.37	69.43

Table 4.1: Timing condition of the descriptors(Aquaculture)

	SIFT	**SURF**	**ORB**	**BRISK**	**AKAZE**
Mean detection time	34770.50	9928.07	3285.28	6999.51	23041.40
Mean transformation time	705.01	355.78	522.29	863.27	71.4695
Mean comparison time	0.31	0.0001	8.59	0.31	0.0001

Table 4.2: Timing condition of the descriptors(Cave)

In the Table 4.1 we can see that the ORB is the fastest when it comes to feature detection, SIFT is the slowest that is also because SIFT usually extracts more feature then the rest of the group. In image transformation from the pixel to feature vector space ORB comes on top and SIFT is the slowest.
ALL in ALL we can deduce the following results (all the times are in microseconds):
Mean feature detection time:
SIFT > AKAZE > SURF > BRISK > ORB
Mean image transformation time:
SIFT > BRISK > SURF > AKAZE > ORB
Mean image comparison time:
BRISK > AKAZE > ORB > SIFT > SURF

SO ORB overall comes out to be the fastest and SIFT the slowest. The results remain the same in Table 4.2 but overall all descriptors became quick because of less features in the environment.

Chapter 5

Conclusion

Based on our experimental findings, we can draw several interesting conclusions. All five descriptors utilized in this case study have established themselves as reliable options for SLAM applications. Notably, SIFT excelled in extracting a high number of features from the aquaculture net cage dataset, while BRISK performed exceptionally well in detecting numerous features in the underwater cave environment. In terms of short-term tracking, both SIFT and SURF outperformed ORB, BRISK, and AKAZE in both environments.

For long-term tracking, all descriptors except AKAZE were capable of identifying local maxima, making it possible to extract loop closure candidates by applying coordinate-based constraints.

Although SIFT demonstrated excellent tracking performance, it was time-consuming, making ORB the more efficient choice in this regard. Moreover, both SIFT and SURF displayed superiority in matching features, presenting a dilemma between these two descriptors.

Further analysis of the results reveals that SIFT exhibited greater consistency compared to SURF, which is more prone to false positives. However, SIFT is more sensitive to disturbances and noise. In conclusion, we recommend using SIFT in most cases. However, if the data contains significant noise or computational constraints are present, SURF can also be a viable alternative.

Given the increasing popularity of underwater SLAM in the aquaculture industry, this work aims to facilitate the selection of the most suitable place recognition tool for Aquaculture SLAM applications.

Bibliography

Akram, Waseem et al. (2022). 'A visual servoing scheme for autonomous aquaculture net pens inspection using ROV'. In: *Sensors* 22.9, p. 3525.

Alcantarilla, Pablo F, Adrien Bartoli and Andrew J Davison (2011). 'Fast explicit diffusion for accelerated features in nonlinear scale spaces'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7, pp. 1281–1298.

Bay, Herbert, Tinne Tuytelaars and Luc Van Gool (2006). 'Surf: Speeded up robust features'. In: *Lecture notes in computer science* 3951, pp. 404–417.

Betancourt, J, W Coral and J Colorado (2020). 'An integrated ROV solution for underwater net-cage inspection in fish farms using computer vision'. In: *SN Applied Sciences* 2.12, pp. 1–15.

Bjerkeng, Magnus et al. (2021). 'ROV navigation in a fish cage with laser-camera triangulation'. In: *Journal of Marine Science and Engineering* 9.1, p. 79.

Cario, Gianni et al. (2017). 'Long lasting underwater wireless sensors network for water quality monitoring in fish farms'. In: *OCEANS 2017-Aberdeen*. IEEE, pp. 1–6.

Chalkiadakis, Vaggelis et al. (2017). 'Designing a small-sized autonomous underwater vehicle architecture for regular periodic fish-cage net inspection'. In: *2017 IEEE International Conference on Imaging Systems and Techniques (IST)*. IEEE, pp. 1–6.

'DBoW3 DBoW3' (2017). In: URL: `https://github.com/rmsalinas/DBow3` (visited on 17/02/2017).

Hidalgo, Franco and Thomas Bräunl (2020). 'Evaluation of several feature detectors/extractors on underwater images towards vSLAM'. In: *sensors* 20.15, p. 4343.

Li, Jie, Ryan M Eustice and Matthew Johnson-Roberson (2015). 'High-level visual features for underwater place recognition'. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3652–3659.

Lin, Tony X, Qiuyang Tao and Fumin Zhang (2020). 'Planning for Fish Net Inspection with an Autonomous OSV'. In: *2020 International Conference on System Science and Engineering (ICSSE)*. IEEE, pp. 1–5.

Livanos, George et al. (2018). 'Intelligent navigation and control of a prototype autonomous underwater vehicle for automated inspection of aquaculture net pen cages'. In: *2018 IEEE International Conference on Imaging Systems and Techniques (IST)*. IEEE, pp. 1–6.

Mallios, Angelos et al. (2017). ‘Underwater caves sonar data set’. In: *The International Journal of Robotics Research* 36.12, pp. 1247–1251.

Offshore fish farming (July 2020). URL: `https://www.salmar.no/en/offshore-fish-farming-a-new-era/`.

Ohrem, Sveinung Johan, Eleni Kelasidi and Nina Bloecher (2020). ‘Analysis of a novel autonomous underwater robot for biofouling prevention and inspection in fish farms’. In: *2020 28th Mediterranean Conference on Control and Automation (MED)*. IEEE, pp. 1002–1008.

Sandøy, Stian S et al. (2020). ‘Polar Map: A Digital Representation of Closed Structures for Underwater Robotic Inspection’. In: *Aquacultural Engineering* 89, p. 102039.

Straume Haugland, Kyrre (2021). ‘Underwater Pose Graph SLAM with DVL-Enhanced Visual Loop Closure for Future Aquaculture’. MA thesis. NTNU.

Tao, Qiuyang et al. (2018). ‘Omnidirectional surface vehicle for fish cage inspection’. In: *OCEANS 2018 MTS/IEEE Charleston*. IEEE, pp. 1–6.

Tareen, Shaharyar Ahmed Khan and Zahra Saleem (2018). ‘A comparative analysis of sift, surf, kaze, akaze, orb, and brisk’. In: *2018 International conference on computing, mathematics and engineering technologies (iCoMET)*. IEEE, pp. 1–10.

Thorvaldsen, Trine, Ingunn M Holmen and Helene K Moe (2015). ‘The escape of fish from Norwegian fish farms: Causes, risks and the influence of organisational aspects’. In: *Marine Policy* 55, pp. 33–38.

What is SLAM (n.d.). URL: `https://www.mathworks.com/discovery/slam.html`.

Zhang, Yin, Rong Jin and Zhi-Hua Zhou (2010). ‘Understanding bag-of-words model: a statistical framework’. In: *International journal of machine learning and cybernetics* 1, pp. 43–52.

Appendix A

Additional Material

This is the github repository of the codes `https://github.com/UsamaMujahid/master_thesis`