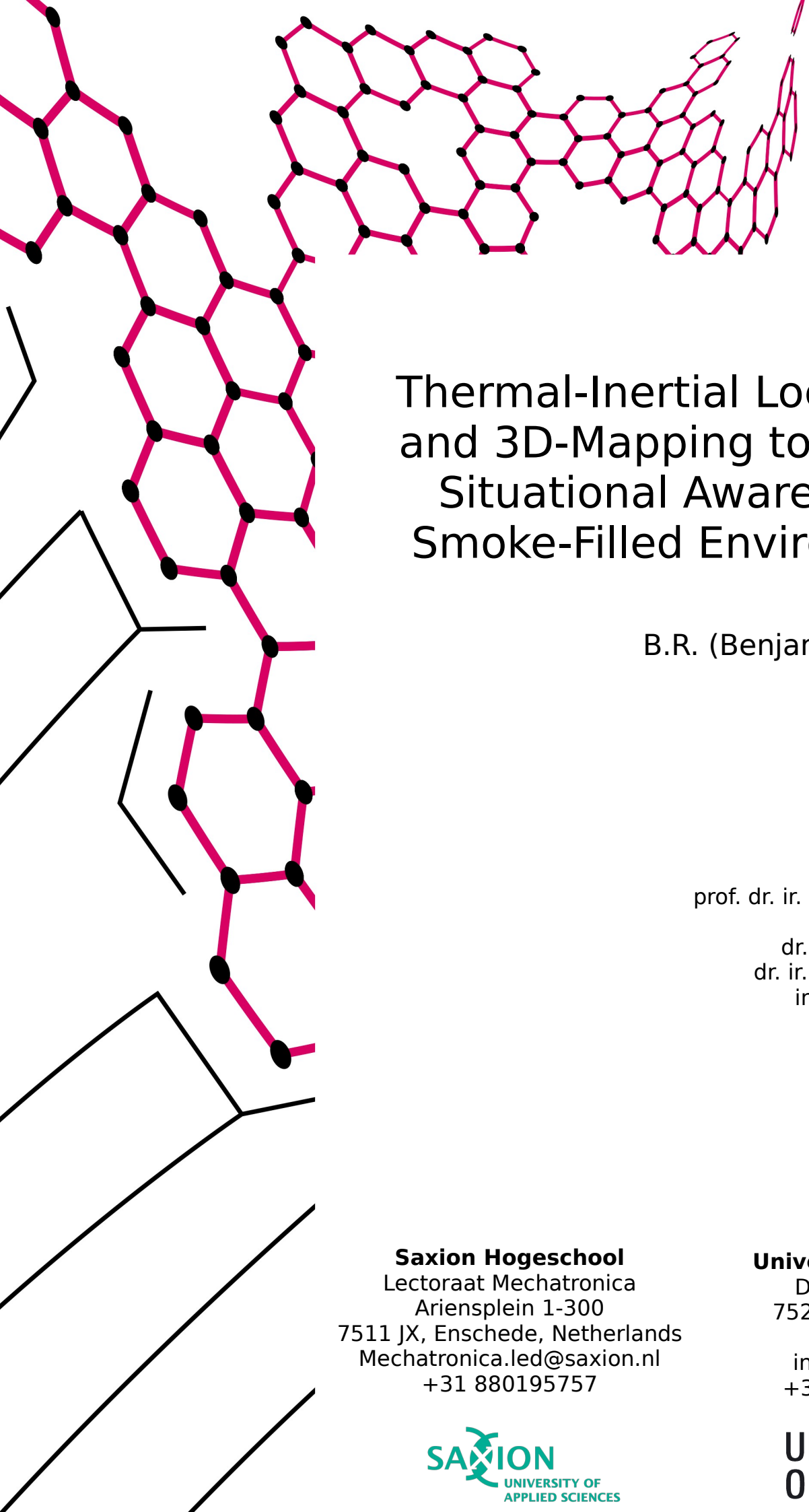


Thermal-Inertial Localization and 3D-Mapping to Increase Situational Awareness In Smoke-Filled Environments

B.R. (Benjamin) van Manen

Msc Thesis

Committee:
prof. dr. ir. M.G. Vosselman (UT)
dr. F.C. Nex (UT)
dr. ir. D. Dresscher (UT)
dr. ir. A.Y. Mersha (Saxion)
ing. V. Sluiter (Saxion)

May 2021

Saxion Hogeschool
Lectoraat Mechatronica
Ariensplein 1-300
7511 JX, Enschede, Netherlands
Mechatronica.led@saxion.nl
+31 880195757

University of Twente
Drienerlolaan 5
7522 NB, Enschede
Netherlands
info@utwente.nl
+31 53 489 9111

SAXION UNIVERSITY OF APPLIED SCIENCES

UNIVERSITY OF TWENTE.

Preface

This thesis is written as part of the author's graduation assignment for the Systems & Control master's degree at the University of Twente in Enschede, Netherlands. The assignment was conducted externally in the context of the Firebot project at the Mechatronics research group of Saxion University of Applied Sciences in Enschede, Netherlands.
Project Firebot is a consortium constituted of Saxion, four Dutch fire departments, the Insituut Fysieke Veiligheid (IFV), the Brandweeracademie and three companies from the Twente region with the mission to research the use of unmanned vehicles in order to increase the safety and effectiveness of firefighters.

Acknowledgments

The author would like to extend his gratitude to ing. V. Sluiter and dr. ir. A. Y. Mersha from Saxion as well as dr. F.C. Nex from the University of Twente for their guidance and continuous support during this thesis. The author would like to thank the staff of the entire Mechatronics lectorship and especially the members of Firebot at Saxion for their warm welcome and support along the project. The author would also like to thank dr. S. Khattak from ETH Zürich for his insights in the subject as well as the staff of the Twente Safety Campus for their assistance during the recordings of the datasets. Finally, the author would like to thank prof. dr. ir. M.G. Vosselman and dr. ir. D. Dresscher for their time examining the thesis.

Summary

Fires in large enclosed indoor spaces, such as industrial buildings and parking garages, lead to rapid smoke propagation through the environment. This does not only make it dangerous for firefighters to intervene, but also makes it difficult to formulate a plan of action due to the lack of situational awareness leading to severe damages and potentially even loss of life. In recent times, a number of fire departments have started deploying Unmanned Ground Vehicles (UGV) to gain situational awareness during fire incidents. Their usability proved to be suboptimal nonetheless due to a limited field of view. Hence, the need for increased situational awareness in the form of a 3D map arose.
Simultaneous Localization and Mapping (SLAM) using visual cameras or Light Detection and Ranging (LiDAR) have been the most popular and developed approaches to obtain a 3D map of the environment in the last decade. These sensors are however rendered useless in the presence of smoke. The properties of thermal cameras enable them to be used to see "through" smoke, making them in contrary a viable option to create a 3D map in a smoke-filled environment.
Obtaining a 3D map from thermal images is not as straightforward as providing the images to a state-of-the-art visual SLAM algorithm. Thermal cameras namely possess some additional challenges such as a lower resolution, high noise and lower contrast.
Since 2016, a handful of primitive thermal-based odometry systems have been published. A thermal-based SLAM algorithm that creates a global map of the environment solely using 3D information extracted from thermal images or assisted by an IMU has yet to be released. Additionally, none provide global optimization such as bundle adjustment by loop closure detection. The algorithms were designed to navigate in low or no-light conditions such as at night or in a mine and have thus not been tested in a fiery environment.
Due to the lower contrast, the literature showed that the main difficulty of applying visual SLAM techniques onto thermal images is obtaining meaningful relations between the images, mainly acquired in the form of matched images features.

During this thesis, research was conducted into the possibility of utilizing a stereo set of thermal cameras and potentially alongside an IMU to, in a consistent and computationally cost-effective manner, localize a UGV and create a 3D map in a smoke-filled environment during a fire incident.
As a result, the first thermal SLAM algorithm capable of creating a comprehensible 3D map using solely 3D information extracted from thermal images was developed. Additionally, the system is also the first thermal odometry system that is capable of obtaining a coherent trajectory by being the first to perform a global trajectory optimization, using bundle adjustment from loop closure detections. Finally, the proposed algorithm is the first thermal odometry and mapping algorithm to be tested in a smoke-filled environment with fire to validate the applicability of the proposed algorithm in an operational environment. To achieve this, a benchmark of various feature extraction and description methods, including a machine learning-based approach, was notably performed to determine the most efficient method to extract and match thermal image features under the targeted operating conditions. Furthermore, the possibility of using bags-of-visual-words to detect potential loop closures is assessed using the top candidate from the benchmark, SURF-BRIEF. To perform the different SLAM tasks such as the odometry computation, optimization and sensor fusion of the thermal and inertial data, the state-of-the-art ORB SLAM 3 algorithm was modified to accommodate SURF-BRIEF as well as to enhance its performance in the hard conditions of thermal images. The obtained 3D thermal maps provided enough detail and consistency for an individual to comprehend the general layout of an unknown building. In contrary to the stock ORB SLAM 3, the proposed thermal SLAM algorithm was capable of detecting the different loop closures present in the datasets to optimize the computed path. It was later discovered that the inertial data recorded at the Twente Safety Campus (TSC) was meaningless due to a faulty sensor. A complete thermal-inertial test could therefore not be performed in a smoke-filled environment. Finally, an attempt was made to obtain a quantitative measure of the system's accuracy using an Optitrack motion capture system. The combination of a very benign laboratory with small thermal gradient and lack of motion around all axes of the ground robot's IMU made it impossible for the system to correctly initialize.

With the developed SLAM algorithm, it was shown that it is possible to utilize a stereoscopic set of thermal cameras to localize a robot and create a 3D map in a smoke-filled environment in the presence of fire. It was observed that, in combination with the feature extractor SURF and descriptor BRIEF, the addition of fire facilitates the extraction and matching of unique image features. The hot smoke emitted by the fire helped to increase the differences of infrared radiation in the parking garage, even when the fire source was out of sight. Despite the malfunctioning of the IMU at the TSC, it was discovered that the inertial data is not essential to the operation of the system as it was possible to obtain a correct and up-to-scale trajectory and coherent 3D map using only the stereo thermal data. During the recordings at the TSC, the vehicle was in direct proximity of the fire. Further research into the robustness of Firebot SLAM in situations further away from the fire source, as well as with different types of smoke should be conducted. Furthermore, research into utilizing the 16-bit radiometric data for more reliability in benign environments in combination with research into IMU initialization techniques more suited for ground vehicles should be performed in order to successfully implement thermal-inertial SLAM in benign environments far from the fire source. Finally, the SLAM process was performed off-board during the development in this project. Once the system is ready to be utilized in real-time, the effect of the high temperatures on the hardware should be investigated.

Contents

List of Figures

List of Tables

List of Acronyms

AUC Area Under the Curve
CNN Convolutional Neural Network
DFA Dutch Firefighters Academy
DoG Difference of Gaussians
FFC Flat Field Correction
FFT Fast Fourier Transform
FP Fase Positives
FPR False Positive Rate
LiDAR Light Detection and Ranging
LWIR Long-wave Infrared
MAP Maximum A Posteriori
NCC Normalized Cross-Correlation
NUC Non-Uniformity Correction
PnP Perspective from n-Points
RGB Red Green Blue
ROC Receiver Operating Characteristic
ROS Robot Operating System
SLAM Simultaneous Localization and Mapping
SVD Singular Value Decomposition
TIO Thermal-Inertial Odometry
TP True Positives
TPR True Positive Rate
TSC Twente Safety Campus
UAV Unmanned Aerial Vehicle
UGV Unmanned Ground Vehicle
VO Visual Odometry

1 Introduction

Fires inside parking garages are often very problematic situations for firefighters.
A burning car can rapidly produce enough smoke such that it becomes difficult to find the source of the fire or to navigate towards an exit. For these reasons, it is sometimes too dangerous for firefighters to enter a parking garage to fight the fire, resulting in severe material damage [21].
According to a risk assessment in relation to fire safety equipment by the Dutch Firefighters Academy (DFA) [21], smoke ventilation systems are commonly chosen as the main fire safety equipment inside parking garages in the Netherlands. The reason behind this choice is that it simply requires an upgrade of the required regular ventilation system, making it the most cost-effective solution in contrast to a sprinkler system. In addition, fire localization sensors must also be installed.
According to the DFA's report, the Dutch NEN6098-norm contains criteria regarding the assessment of smoke ventilation systems inside parking garages. A smoke ventilation system must be designed to either guarantee clear sight on the fire for at least 27 minutes after detection or evacuate enough smoke within 40 to 60 minutes after detection such that firefighters can safely search for the fire source inside the structure. The assessment criteria inside NEN6098 are based on the assumptions that no more than three cars are on fire and that firefighters are able to start extinguishing the flames within 20 minutes after detection of the fire.
Practice has shown however that in most fires inside Dutch parking garages, more than four cars are affected by flames and firefighters are rarely able to quickly enter the structure [21]. Research from Universiteit Gent demonstrated that the ventilation capacity to disperse smoke needs to be immensely high. A ventilation factor (the capacity of a ventilation system to disperse airborne pollutants from a stationary source) of 10 is far from the required factor to achieve adequate smoke dispersion. The research added that, in general, existing parking garages did not satisfy this minimum ventilation factor [22].
Furthermore, research showed that fire risks inside Dutch parking garages still increased since the release of norm NEN6098. This is in part due to the expanded use of composite materials in vehicles [23]. With the recent popularization of electric vehicles, new concerns about potential related fire risks inside parking garages have also emerged. A combination of a literature review and experiments performed by TNO revealed that electric vehicles do not present an increased fire hazard. It was however highlighted that, due to the batteries, electric vehicles have shown to burn for a longer period of time and create very toxic smoke compared to their fossil-fueled counterparts [24].

Unmanned Ground Vehicles (UGV) are increasingly being adopted by fire departments across the world to gain situational awareness and to extinguish fires in environments that are hardly accessible or too dangerous for human firefighters.
A common design of firefighting UGVs is a tracked vehicle equipped with a fire extinguishing device such as a cannon or a nozzle and different perception sensors such as a visual and thermal camera, as seen in figure 1.1. The vehicles are always operated remotely. Examples of these robots are the Dutch Parosha Cheetah GOSAFER, Germany's LUF 60 and TAF 20, the American Thermite 3.0, the British Firemote and the South Korean Archibot [25].

Figure 1.1: Thermite 3.0 [1] (left) and TAF 20 [2] (right)

More unconventional robots have also been designed, such as the Anna Konda in figure 1.2, a three meter long snake-like robot capable of creeping through debris and extinguishing fires [25].

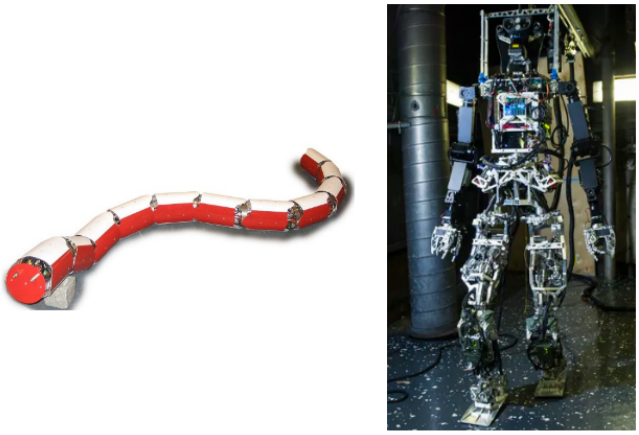


Figure 1.2: Anna Konda snake firefighting robot (left) [3] and SAFFIR semi-autonomous firefighting humanoid (right) [4]

Finally, a semi-autonomous firefighting humanoid, SAFFIR in figure 1.2, was developed by Virginia Tech to fight fires inside navy ships. The robot's movements are tele-operated but it is capable of autonomously detecting and extinguishing fire using fused data from a stereo pair of thermal cameras and a radar [26].

Following the firefighting UGV trend, in order to lower the number of casualties and increase the safety of its firefighters, several Dutch fire departments have acquired unmanned ground vehicles such as the ones displayed in figure 1.3. The vehicles are being used to gain information about the risks and to better find the fire source inside a parking garage. Currently, these UGVs are often fitted with both a regular and a thermal camera, through which the operator can safely assess the situation through a monitor.

Figure 1.3: Brandweer Amsterdam-Amstelland's Tecdron Scarab TX [5] (right) and Brandweer Haaglanden's LUF 60 [6] (left).

This reconnaissance limits however the situational awareness around the robot for the operator and does not provide a global picture of the situation at hand to create a better plan of action.
To remedy this problem, Dutch fire departments desire to implement a 3D-mapping system on their UGVs.
A commonly-used simultaneous localization and mapping (SLAM) system utilizing light detection and ranging (LiDAR) was previously tested in this project but was unsuccessful to obtain precise ranging measurements in smoke-filled environments. Laser rays would often reflect from smoke particles, causing the mapping software to misinterpret these as detected obstacles. Similar research showed

that LiDAR becomes obsolete in smoke-filled environments once the visibility decreases below 5 meters [27]. Furthermore, ultrasonic or radar ranging yielded accurate distance measurements however the sensor lacked the resolution needed to obtain a detailed map. Additionally, ultrasonic and radar ranging sensors are directional and are only capable of scanning a 2D plane. An array of sensors on each side of the UGV would therefore be required to obtain a complete 3D-map of the surroundings. Image-based 3D mapping has been well-researched and a number of state-of-the-art high accuracy mapping algorithms are available [14]. With the recent developments of higher resolution thermal cameras, this type of camera has become a viable alternative in order to obtain the desired 3D-map. Thermal cameras create images by detecting rays in the long-wave infrared (LWIR) spectrum (see figure 1.5) in contrast to detecting rays in the visible spectrum adopted by regular cameras. Contrary to LiDAR and regular visible-light cameras, thermal cameras are much less affected by the presence of smoke, as can be seen in figure 1.4.

Figure 1.4: Webcam (left) and LWIR thermal (right) view in a smoke situation

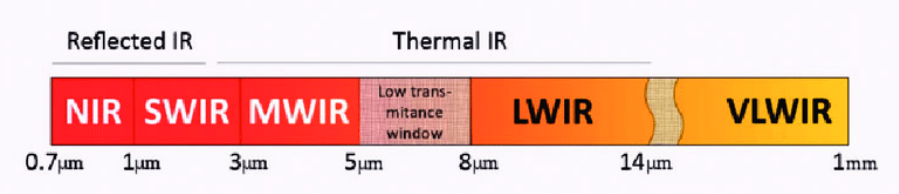


Figure 1.5: Infrared electromagnetic spectrum. N: near; SW: short wave; MW: mid wave; LW: long wave; VLW: very long wave [7].

Simultaneous localization and mapping using visible-light cameras has been a well-researched topic and has especially been perfected in the last decade. One of the keypoints determining the success of such an algorithm are robust image features which can be tracked across multiple frames. These features, e.g. corners or blobs, are commonly detected using the image's contrast.
The contrast of a LWIR thermal image is determined by the radiated energy in the environment, and more specifically the differences in infrared emissivity. In benign environments, this leads to images with a much lower contrast due to the lack of temperature differences, making the detection of robust features challenging. Additionally, thermal images accumulate non-uniform noise over time. As a result, LWIR cameras perform a flat-field correction (FFC) where a shutter with uniform temperature is placed in front of the array of infra-red sensors to calibrate them. Such an operation is performed multiple times per minute during which the camera's view is blocked for up to two seconds.
These additional challenges make thermal-based SLAM a new research subject. At the time of writing, no thermal-based SLAM algorithm capable of creating a full 3D map has been released. In 2015, T. Mouats et. al[10] developed a stereo thermal odometry algorithm. The algorithm utilizes FAST corner features together with the FREAK binary descriptor to compute the trajectory flown by a UAV. Additionally, the authors successfully managed to create a disparity map of an outdoor

scene using a stereo pair of thermal cameras. In 2019, S. Khattak et. al.[11], as part of the DARPA Subterranean Challenge, developed KTIO, a monocular thermal-inertial odometry algorithm utilizing the full 16-bit radiometric data from a thermal camera to detect gradient-based features. KTIO is able to robustly navigate a UAV through an underground mine, while state-of-the-art visual-inertial odometry algorithms with 8-bit thermal images failed to do so. Although some successful results in localization have been obtained in the last five years, a complete thermal SLAM method including a 3D map of the environment has yet to be achieved.

During this project, the feasibility of implementing stereo thermal-inertial SLAM in a smoke-filled environment to determine in a consistent and computationally cost-effective manner the location of a robot and create a 3D map during a fire incident will be assessed. To achieve this, the first complete 3D map constructed from thermal images will be attempted. The developed thermal-inertial SLAM algorithm will be the first system of its kind to be tested in a real scenario featuring fire and smoke. The accuracy of the algorithm will be measured using a motion capture system.
To complete the project, multiple datasets are recorded at the Twente Safety Campus inside a mock-up parking garage filled with smoke as well as fire sources. In the first part of the project, a benchmarking of multiple combinations of feature extractors and descriptors is performed to determine the most optimal way of obtaining robust features inside a smoke-filled environment. Along with the classical feature extractors and descriptors commonly used in visual computer vision algorithms, an interpretation of S. Khattak et. al.'s KTIO [11] gradient-based feature tracking in 16-bit radiometric thermal data and CNN-computed descriptors will also be evaluated. Next, the feasibility of bag-of-visual-words loop-closure detection, commonly used in feature-based visual SLAM algorithms will be assessed on the top performer from the benchmarking. Finally, the extractor-descriptor that is able to most-optimally detect robust features will be implemented inside ORB SLAM 3, a state-of-the-art visual-inertial SLAM framework in order to obtain a 3D thermal map of the environment. The entire SLAM process must be able to run on board the vehicle.

2 Related work

Visual odometry (VO), i.e. the estimation of a vehicle's motion from consecutive images, is a well-research topic, with many developments in the last ten years. The recent popularity of this odometry method can mainly be attributed to the development of smaller and cheaper high-resolution cameras, making it a very cost-effective and light-weight solution for localization.
A VO-algorithm can be further extended to be able to create a 3D-map of the environment. Such algorithms are called Simultaneous Localization and Mapping (SLAM) algorithms.
This section will first give an overview of the main concepts constituting a visual SLAM algorithm, including a short summary of the most popular feature detection and description algorithms. In the second part of this section, the current developments of thermal odometry will be presented along with two feature extraction and matching benchmarkings on thermal images.

2.1 Visual SLAM

2.1.1 The concept

According to D. Scaramuzza et. al. [28], the success of a VO algorithm depends on a number of factors. It is important that the scene is sufficiently and preferably uniformly illuminated as to obtain an image where the color gradients are correctly visible with minimum shadows. Furthermore, as VO estimates the vehicle's relative motion with respect to the environment, a relatively static scene is required to obtain accurate estimates. In order to prevent ambiguities, an environment rich in varying textures is preferred. Finally, successive images must have sufficient scene overlap for comparison.

A typical visual SLAM workflow can be found in figure 2.1. As the name suggests, the SLAM-algorithm is divided into two sections, namely the localization part responsible for computing the vehicle's position and the mapping segment which creates a 3D-reconstruction from the projected 2D features. Similar results in localization and mapping can be obtained using monocular vision and stereo vision. In monocular vision however, the trajectory can only be obtained up to a scale factor with respect to the real world [28]. Indeed, because the depth information cannot be obtained from a single camera, an external source for the scaling factor such as, for example, an IMU [29] or a trained convolutional neural network (CNN) [30] is required.

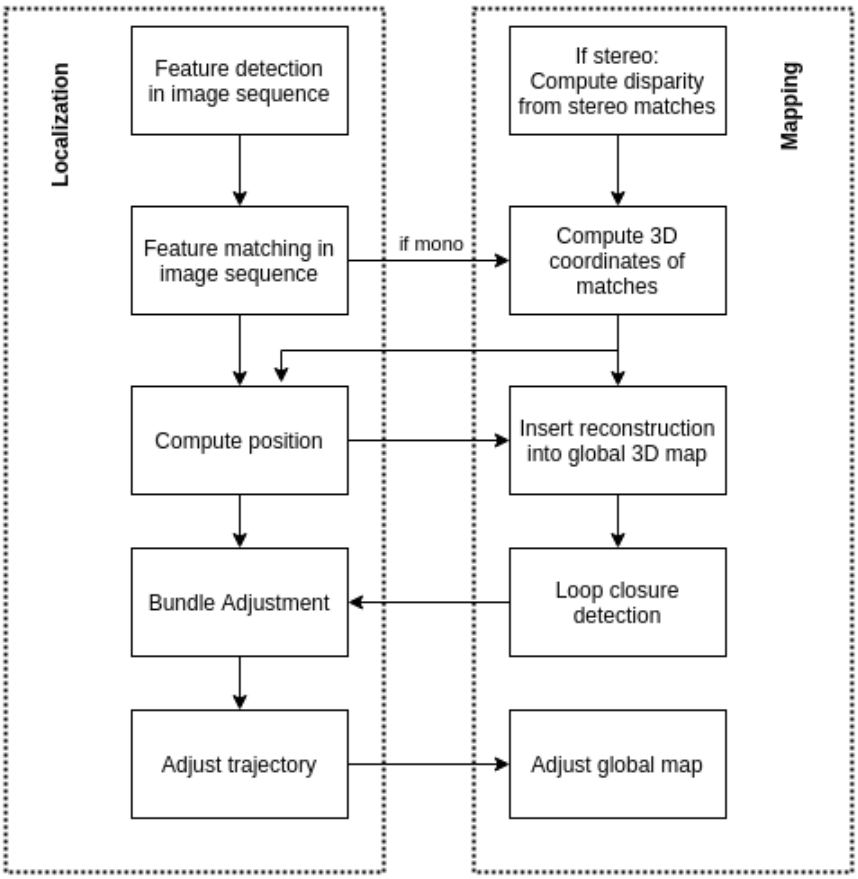


Figure 2.1: Visual SLAM-algorithm workflow

In stereo SLAM, both monocular with an IMU [31] and stereo localization using a sparse 3D-point cloud can be adopted for localization [32].
Both methods rely on point feature detectors and descriptors. Commonly-used feature detectors include the Harris corner detector, FAST, SIFT and SURF among others. In order to match identical features in an image pair, these keypoints first need to be described. To achieve this, algorithms such as BRIEF, BRISK, SIFT, SURF and ORB create an array that uniquely describe the surroundings of each feature [33].
After sufficient features are found, identical keypoints between two images are matched together following a specified metric such as the L1- or L2-norm, the Hamming distance, the sum of squared differences (SSD) or the normalized cross-correlation (NCC) by comparing their respective descriptors [34]. Incorrect matches can be removed using an outlier removal method such as Random Sample Consensus (RANSAC). For each iteration, RANSAC computes a hypothesis of the object projection between the two images using a random selection of matching features. The hypothesis will then be assessed against the entire set of matches. Finally, the hypothesis with the highest consensus is taken and its inliers remain [34].

The next step consists of computing the camera motion between the two poses. In monocular VO, the motion is obtained by calculating the essential matrix E containing the motion parameters (up to a scale factor) by solving the set of 2D-to-2D reprojection equations in eq. (1) where $\tilde{p}$ and $\tilde{p}'$ are the homogeneous image coordinates of two matching features. The equation is characterized by the epipolar constraint, which states that the match of a feature located on an epipolar line in the first image, is situated on the corresponding epipolar line in the second image [28].

$$\tilde{p}'^{T} E \tilde{p} = 0 \tag{1}$$

In stereoscopic VO, the 2D image features are first reprojected into 3D-coordinates. The pose trans-

formation matrix T (including the scale) is then obtained by minimizing the L2-norm as shown in eq. (2) where $\tilde{X}$ are the homogeneous feature's 3D-coordinates, k is the current iteration and i is the feature number [28].

$$\arg\min_{T_k} \sum_i ||\tilde{X}_k^i - T_k \tilde{X}_{k-1}^i|| \tag{2}$$

Finally, the relation between the starting location $\mathbf{X}_0$ and the current location $\mathbf{X}_n$ of the UGV can be computed by multiplying the product of all the pose transformation matrices with the starting point.

$$\mathbf{X}_n = \left(\prod_{k=1}^{n} T_k \right) \mathbf{X}_0 \tag{3}$$

A dense 3D-map is a collection of 3D-reconstructions created locally using stereo pairs of rectified images. Compared to a sparse reconstruction, the pixel-based 3D estimation offers a more detailed representation of the environment. During this project, a stereo pair of thermal cameras is available, allowing the creation of such a dense 3D-map.
In contrary to a sparse reconstruction, a dense reconstruction first consists of matching every single pixel in the overlapping area of the stereo pair. A window of size W is taken around a pixel in the left image. Its counterpart is then found in the right image using a cross-correlation analysis on the corresponding epipolar line. For both a sparse and dense reconstruction, the disparity d, i.e. the displacement of a point along the epipolar line in one image compared to the other, of each point of interest is computed using eq. (4) where u is the horizontal image coordinate and the subscripts L and R respectively correspond to the left and right image of the stereo pair [35].

$$d = u_L - u_R \tag{4}$$

Before reprojecting the points into 3D-coordinates, it is important to remove the erroneous disparity values. As can be observed in figure 2.2, the uncertainty of the computed disparity increases as the object is situated further away from the stereo cameras. Furthermore, wrong disparities can also occur due to falsely-matched points. Such values can be eliminated by analyzing the cross-correlation results by peak refined for example [35].

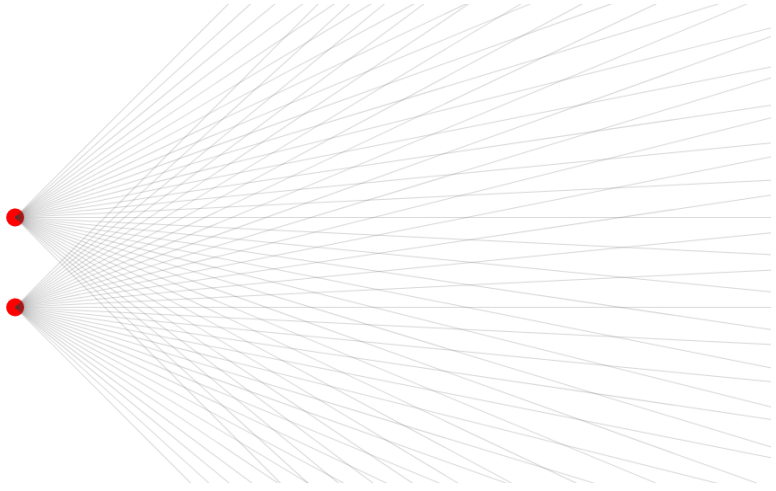


Figure 2.2: Disparity uncertainty in relation with object-to-camera distance [8]

The 3D-coordinates of each valid point can finally be computed using eqs (5)-(7) where f is the focal length in meters, b is the baseline (the distance between the two camera centers) in meters and (u, v), (u_0, v_0) are respectively the image coordinates of the point and the image center [35].

$$X = \frac{b\,(u - u_0)}{d} \tag{5}$$

$$Y = \frac{b\,(v - v_0)}{d} \tag{6}$$

$$Z = \frac{fb}{d} \tag{7}$$

Note that in the equations above, Y is pointing downwards and Z is positive in the depth direction. Using eq. (3) to compute the robot's absolute position will result in a slow accumulation of small errors, which will ultimately produce a distorted map. To prevent this problem, different optimization methods are adopted to increase the map coherence. Earlier solutions of SLAM modelled the probability distribution of the localization and 3D reprojection errors. It was first assumed that the errors followed a Gaussian distribution and thus an Extended Kalman Filter (EKF) was utilized. However, due to the non-linearity of the problem, this method resulted in large linearization errors on top of being computationally expensive. To remedy this, later solutions adopted an Unscented Kalman Filter (UKF) which is computationally less complex and assumed a non-Gaussian distribution of the localization error. Furthermore, particle filters (sequential Monte Carlo) were employed which estimates the probability distribution by rating a set of generated pose hypotheses [36]. Each (weighted) sample contains a possible pose of the vehicle with its suspected generated 3D-reconstruction inside the global map. The 3D-reconstructions of the samples are compared with the measured reconstruction to evaluate the most-likely pose [37]. Finally, recent SLAM-algorithms in general utilize bundle adjustment, or pose graph optimization, which imposes constrains between consecutive poses in the form of the probability distribution of their relative location. The map is then corrected once a loop closure, i.e. the vehicle visits a location for the second time, has been detected [38].

2.1.2 Popular Visual-Inertial SLAM algorithms

A number of Visual-Inertial SLAM algorithms have been developed and they can be divided into two categories, namely sparse and direct (or dense) SLAM.
The first category, sparse SLAM, transforms the image into a sparse set of keypoints before performing the odometry and mapping. This method is computationally faster, more invariant to illumination difference between a pair of images and allows matching of images with a wider baseline (i.e. larger distance between camera positions of the two images). On the other hand, the 3D map obtained is less detailed as it contains sparsely located points. Additionally, their performance degrades in locations of weak intensities and do not sample edges as most common keypoint detectors rely on corners and blobs [39]. Popular visual-inertial SLAM algorithms utilizing the sparse method include OKVIS [40], ROVIO [41] and ORB-SLAM-3 [14].
Direct, or dense, SLAM to the contrary tries to match as much pixels as possible between two images, making it computationally slow. To compute the odometry, most direct algorithms utilize a variant of the Lucas-Kanade tracking algorithm. Using the direct method compared with the sparse method, in general, offers virtually no gain in performance with respect to the odometry estimation. It can however be beneficial in low-textured environment where sparse methods cannot provide sufficient matches. The main advantage of direct SLAM relies in the increase in detail when creating 3D maps [39]. Some of the state-of-the-art direct visual-inertial SLAM methods are VI-DSO [42], VINS-Fusion [43] and LSD-SLAM [42][44].

2.2 Feature extraction and description

In this final section related to SLAM in the visual domain, commonly-used feature extraction and description algorithms suitable for real-time application will be explained.

2.2.1 Feature extraction

In visual SLAM, two types of feature extraction methods are used. These are corner detection and blob detection. While the first method is self-explanatory, blobs are defined as being image regions

with a similar property such as color intensity.
In Appendix B, five of the most common feature extraction algorithms are explained. The first three, FAST, ORB and Good Features To Track (GFTT) are used for corner detection whereas the final two, SIFT and SURF are blob detectors.

2.2.2 Feature description

After features have been detected, a descriptor is used to uniquely describe its proximity in the image in order to match identical points in different images.
In Appendix C, the functioning of the most common descriptors are explained. The first two descriptors, SIFT and SURF, are float-point descriptors, meaning that the feature description array is composed of float values. On the other hand, FREAK, BRISK, BRIEF and ORB, are binary descriptors meaning that the intensity of specifically-chosen pixels around a feature are compared using bit-wise operations. In this section, a novel method utilizing machine learning to describe the image region around two features to create the descriptors is discussed.

2.2.2.1 Siamese CNN feature description and matching

Recently, research into generating feature descriptors using deep-learning has emerged. The descriptor is formed by extracting areas of interest from an 32×32 image patch centered around the feature using sequential convolutional layers in a convolutional neural network (CNN). The 2D filters of the final convolutional layer are then compressed into a single 1-dimensional descriptor, as depicted in figure 2.3.

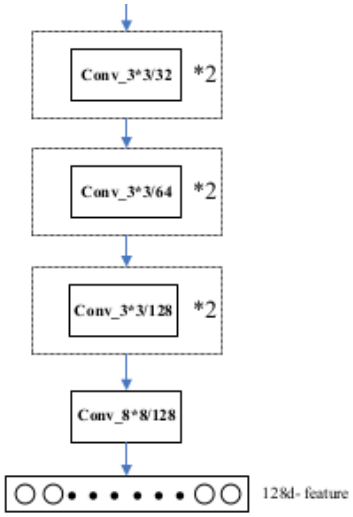


Figure 2.3: CNN-based descriptor architecture by Liu et. al. [9]

To train the deep-learning network, Y. Liu et. al. [9] utilizes a triplet siamese architecture, where the descriptor of a query as well as both of a matching (positive) and non-matching (negative) feature are computed in parallel. The network is then optimized to minimize the difference between the L2-norm of the matching features and the L2-norm of the non-matching features:

$$L = \arg\min \sum_{i}^{N} \max(0, ||x_i - x_i^p|| - ||x_i - x_i^n|| + t) \tag{8}$$

where L is the loss function, N is the total number of features present in the training dataset, x_i, x_i^p, x_i^n are respectively the descriptor of the query feature, a matching feature and a non-matching feature and t is a pre-defined threshold.

2.3 Thermal-Inertial SLAM

In recent years, research has been conducted into applying visual SLAM techniques using thermal cameras. Thermal imaging namely possesses a number of advantages compared to RGB-images as it is not dependent on the environment's illumination conditions making it more suitable in dusty conditions and at night. Thermal cameras are however more sensitive to noise and often have low image resolution [10]. Additionally, as LWIR images show the thermal information of the scene, thermal images in general have less texture information making finding features to track more challenging [11].

Although no research into thermal-inertial SLAM could be found that successfully created a global 3D-map of the environment, several research groups succeeded in utilizing thermal cameras and an IMU to compute the odometry of a vehicle.

Research by T. Mouats et. al. [10] implemented stereo thermal odometry onto an unmanned aerial vehicle (UAV) and created a disparity map from a pair of stereo thermal images. The authors first describe two LWIR-camera chessboard calibration setups, namely using a polished aluminium coated with matt black squares or a heated MDF board with laser-cut squares. Both setups were then used to compare different camera calibration software and found that using the aluminium chessboard in combination with the Automatic Multi-Camera Calibration (AMCC) Toolbox resulted in the smallest mean reprojection error.
In order to be able to find robust features in thermal images, the researchers compared a variety of feature detectors and descriptors. From the results, the Fast-Hessian detector was selected to extract image features and to describe them using the binary descriptor FREAK. It was noted that binary descriptors also offer faster matching speed compared to floating point descriptors. To increase the quality of the images, the top part (generally the sky) is masked out. Furthermore, an automatic gain control (AGC) algorithm which actively changes the dynamic range of each image was used to increase the contrast of the thermal images.
The movement of the UAV is estimated from a sparse 3D point cloud and corresponding 2D images features (3D-to-2D) using the perspective from n-points (PnP) algorithm. PnP minimizes the reprojection error of the set of 3D points from the previous iteration into the current image. This method provides greater accuracy as opposed to 3D-to-3D camera motion estimation, which minimizes the 3D-point cloud position estimation [28]. To solve the reprojection equations, T. Mouats et. al. compared a range of non-linear solvers including Gauss-Newton (GN), Levenberg-Marquadt (LM) and the Double Dogleg (DDL) algorithm. Testing on a full path showed that the thermal odometry using Double Dogleg presented similar results to Levenberg-Marquadt but at a much faster computational speed. On the other hand, thermal odometry using Gauss-Newton performed the worse out of the three solvers [10].
In figure 2.4, the average and final errors for each test sequence conducted by T. Mouats et. al. are shown. The average error is calculated as in the KITTI dataset [45] and experiments were conducted both during day- and nighttime at different environmental conditions. To acquire the thermal images, two FLIR Tau2 LWIR cameras with a resolution of 640x480 pixels at 30 frames per second were used. For each sequence, the best average error is highlighted in bold and the best final error in red. Overall, the Double Dogleg thermal odometry performed the best path estimation, as it obtained the best average error in 7 out of the 9 sequences as well as the best final error in 5 out of the 9 sequences.

#	Sequence	travelled distance	Time of day	Weather conditions
1	**Seq1**	687m	daytime (11am)	cloudy and cold
2	**Seq2**	497m	daytime (10am)	foggy and relatively warm
3	**Seq3**	701m	daytime (9am)	foggy and relatively warm
4	**Seq4**	460m	daytime (11am)	cloudy and cold
5	**Seq5**	882m	daytime (10am)	cloudy and warm
6	**Seq6**	489m	night-time (9pm)	clear sky and very cold
7	**Seq7**	399m	night-time (10pm)	cloudy and cold
8	**Seq8**	314m	night-time (10.30pm)	cloudy and cold
9	**Seq9**	655m	night-time (10pm)	cloudy and cold

Seq#	Travelled distance	GN		LM		DDL	
		Avg	Fin. % (m)	Avg	Fin. % (m)	Avg	Fin. % (m)
Seq1	687m	3.79	1.78 (12.2m)	3.53	2.24 (15.39m)	**3.07**	1.19 (8.18m)
Seq2	497m	3.06	0.15 (0.75m)	**2.30**	0.57 (2.83m)	3.04	1.38 (6.86m)
Seq3	701m	3.05	2.44 (17.10m)	2.85	1.04 (7.29m)	**1.14**	1.33 (9.32m)
Seq4	460m	2.11	0.50 (2.3m)	2.39	1.88 (8.65m)	**2.07**	0.61 (2.81m)
Seq5	882m	3.59	1.29 (11.38m)	3.87	1.52 (13.41m)	**2.17**	1.53 (13.49m)
Seq6	489m	6.28	1.46 (7.13m)	5.59	4.06 (19.85m)	**4.28**	1.33 (6.5m)
Seq7	399m	2.08	2.26 (9.02m)	2.00	1.99 (7.94m)	**1.36**	0.67 (2.67m)
Seq8	314m	1.46	1.38 (4.33m)	**1.44**	0.95 (2.98m)	1.51	0.72 (1.32m)
Seq9	655m	1.91	1.65 (10.81m)	1.91	2.13 (13.95m)	**1.41**	0.45 (2.95m)

Figure 2.4: Test sequences (left) and corresponding results of travelled distance estimation errors (right) by T. Mouats et. al. [10]

Finally, although T. Mouats et. al. did not create a global 3D-map, the researchers successfully created a 3D dense point cloud from a stereo pair of thermal cameras. As can be observed from figure 2.5, the disparity map from the thermal and visual cameras are very similar in detail. The thermal dense point cloud does seem to better visualize the tree in the distance and the cost of some additional noise. Overall, these results have proven that it is feasible to obtain a detailed 3D reconstruction from thermal images.
Recommendations of further research by the authors includes investigating the fusion of the thermal odometry with inertial measurements.

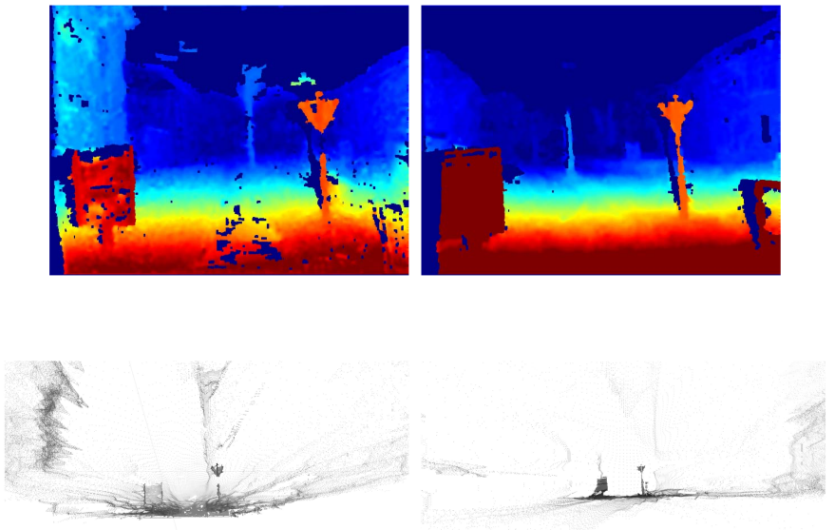


Figure 2.5: Disparity map (top) and 3D dense point cloud (bottom) using a stereo thermal cameras (left) and stereo visual cameras (right) by T. Mouats et. al. [10]

Another solution to thermal odometry was proposed by Borges et. al. [46], which instead of tracking features across frames, utilized semi-dense optical flow on sub-sampled images to estimate the camera motion located on a ground vehicle. Another noticeable difference is the use of only a single thermal camera. The scale factor is estimated using an IMU and road plane segmentation.
Thermal cameras accumulate non-uniform noise over time. To counter this, most cameras perform a Non-Uniformity Correction (NUC), or FFC, which closes the shutter during operation for up to two seconds to expose the sensors to a uniform temperature surface. Such a calibration is performed between one to three times a minute during which the image feed is turned off. The authors noted that this can have detrimental effects on the accuracy of the odometry. Experiments showed that this is especially problematic if the NUC is performed during a turn. To prevent this from occurring, the researchers proposed a NUC Trigger Manager, which based on the current vehicle rotation, future turns, the last applied NUC and the temperature change, determines if a NUC is allowed at the current time.
Experiments were conducted using a single Thermoteknix Miricle 307 K with a resolution of 640x480

at 14 frames per second. The vehicle was driven outside in an Australian industrial area both during the day and at night. The thermal odometry algorithm with triggered NUC obtained similar results to visual odometry at daytime. Furthermore, no performance difference from the thermal odometry was observed when operating at night.

More recently, research was done into directly utilizing the raw data from thermal images for thermal odometry. Research from S. Khattak et. al. [11] compares four different approaches to thermal-inertial monocular odometry, namely utilizing feature tracking in rescaled 8-bit or in the radiometric 14-bit data and feature matching.
In the first method, the Robust Visual Inertial Odometry (ROVIO) algorithm by M. Bloesch et. al. [41] destined for visual images is directly applied onto rescaled thermal images. The algorithm finds features using the FAST detector in certain frames and tracks those features using semi-direct image alignment by minimizing the sum of differences in intensities. This image alignment approach is considered direct as it is performed on a pixel level, however, because such a method can become computationally expensive, it is only performed on a patch of pixels around each detected feature. The tracked features are projected into the incoming frame. Tracking is then refined by minimizing the sum of differences inside a defined window around each projected feature. The states of the different features are being corrected using inertial data.
Applying visual odometry techniques with thermal cameras requires the images to be rescaled from 14 bits to 8 bits, which results in significant information loss. Additionally, histogram corrections using automatic gain control are often applied on the 8-bit thermal images to improve the contrast for feature detection. In environments where temperatures are changing or when hotter objects enter the frame, this can lead to feature loss. Experiments conducted with a thermal camera in a cold room heated by a radiator showed the normalized images slowly saturated [11].
To circumvent the different problems associated with the rescaling of thermal images, S. Khattak et. al. [11] converted the ROVIO algorithm into the Robust Thermal Inertial Odometry (ROTIO) algorithm, which directly uses the 14-bit radiometric data of the camera for feature tracking. In ROTIO, FAST feature detection is utilized to indicate optimal image regions for tracking, meaning that the thermal images are still being converted to 8 bits and histogram equalization is applied for better contrast. However, the tracking process itself using image patches is done in the raw radiometric data. Because feature detection is not performed on every frame but only when the number of features is below a certain threshold, the detection is only dependent on the contrast of the current frame. Utilizing the radiometric data for tracking makes the process independent of contrast changes between frames caused by the image data rescaling.
Furthermore, the authors designed the Keyframe-based Thermal Inertial Odometry (KTIO) which works similarly to ROTIO with the difference that KTIO operates fully on the 14-bit radiometric data. Instead of using the FAST corner detector, this algorithm finds features by utilizing the local gradient information. Using these gradient features, consecutive frames are again aligned by minimizing the sum of differences. This process is repeated in a pyramid scheme, from a coarse low-resolution image to the original image. The projection of the first set of images is initialized using the IMU measurements. Further iterations inside the pyramid are initialized using the results of the previous alignment. Once the features have been tracked in the new frame, their 3D-coordinates are estimated using triangulation from the OpenGV library. Finally, the motion of the camera between the sets of 3D landmarks is computed by 3D-to-3D structure correspondence. The estimates of the different landmarks' and the vehicle's states are being improved using an EKF [47].
Lastly, for comparison, S. Khattak et. al. added the Open Keyframe-Based Visual Inertial SLAM (OKVIS) by S. Leutenegger et. al. [40] in their research. OKVIS is a more typical visual odometry algorithm which was here directly applied to corrected thermal images. The algorithm uses a Harris corner detector to detect features which it subsequently tries to match in the following frame using the BRISK binary descriptor. The visual odometry is then refined locally using IMU data and a windowed bundle adjustment.
Experiments were conducted indoors in a room with a computer-controlled space heater. The hardware consisted of a tele-operated DJI Matrice 100 UAV equiped with a single FLIR Tau2, a VN-100 IMU

and an Intel NUC i7 for the processing. The ground truth was obtained using a VICON motion capture system. The odometry was performed on a trajectory that consisted of five loops at constant altitude around a rectangle of 4.0 by 2.5 meters. When comparing the performance of ROVIO and ROTIO, two identical algorithms apart from the type of image data they operate on, it was concluded that directly using the radiometric data instead of rescaled 8-bit data significantly improves the both the accuracy and the precision of the odometry. Looking at the sum of root mean square errors (SRMSE) over the translations in the three directions, ROVIO obtained a SRMSE of 2.11 meters whereas ROTIO only obtained a SRMSE of 0.81 meters. Furthermore, figure 2.6 shows both the translation estimates and errors of the three different types of thermal inertial odometry presented in their paper. It can be noticed that the traditional visual inertial odometry method, OKVIS, utilizing feature detection and matching across every frame rapidly deviates from the ground truth. Based on the results from the original paper [40], this was already expected by S. Khattak et. al. Moreover, both ROTIO and KTIO manage to accurately estimate the entire trajectory of the UAV. KTIO seems however to perform a more precise and robust odometry. It can be noticed that ROTIO's estimation starts to oscillate with time up to ± 1 meter along the horizontal axes whereas these are barely noticeable in KTIO's odometry.

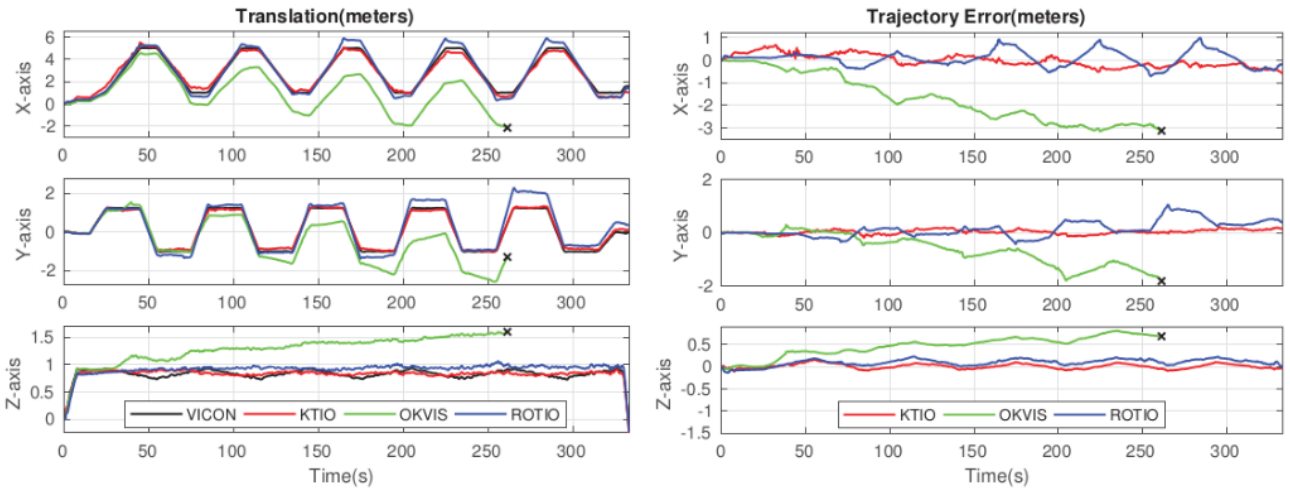


Figure 2.6: Translation estimations (left) and errors (right) of the three different types of thermal inertial odometry algorithms presented by S. Khattak et. al. against the ground truth from VICON [11]

In conclusion, S. Khattak et. al. state that the use of radiometric data in thermal inertial odometry algorithms offers significant performance improvements with respected to rescaled thermal images and should therefore be the standard selection of choice.

To overcome the scarcity of robust features in thermal inertial odometry, research into using machine learning for feature detection from 16-bit radiometric data has also been conducted. M. Supatra et. al. [12] created a monocular thermal inertial odometry algorithm combining different deep neural networks for the estimation of a 6-DOF pose.

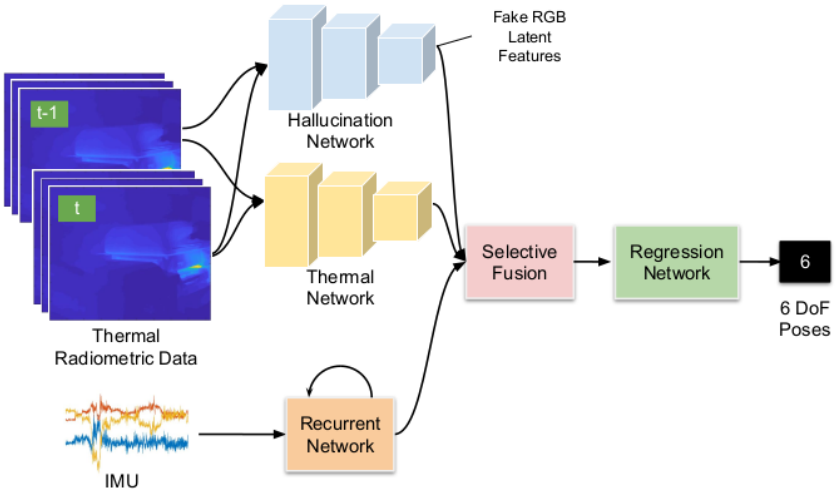


Figure 2.7: Architecture of DeepTIO [12]

To extract geometrically meaningful features from two consecutive frames, the authors adopted a combination of a Convolutional Neural Network (CNN) that detects and encodes features directly from the 16-bit radiometric data and a hallucination network. The latter is used to incorporate additional "fake" features that can be tracked across frames to compensate for the lack of robust features in thermal imaging. Both networks were initialized by the FlowNet-Simple structure, which uses a CNN to determine the optical flow from a monocular video stream [48]. The hallucination network was subsequently trained to imitate the feature detection performed by a visual encoder on RGB images. During training of the hallucination network, a NUC of the thermal camera causes a freeze of the thermal video stream leading to the network associating the same thermal image to multiple RGB images. As a result, the network would receive a large amount of outlier features as inputs which significantly deteriorates the quality of training when using the Mean Square Error (MSE) loss function. Therefore, it was chosen to train the hallucination network following the Huber loss function instead, which is more resistant to outliers.
The IMU data is encoded using a single Long Short-Term Memory (LSTM) layer with 256 nodes which takes the 6-DOF inertial data from a sequence of 20 frames.
To combat erroneous data, the algorithm selectively fuses the data from the three encoders, as thermal images may contain fixed-pattern noise and IMU data often suffers from white noise and biases. Additionally, the hallucination network may produce meaningless features. To implement selective fusion, a trained mask is applied for each encoder, which determines for each feature value if it is suitable to be used in the pose estimation.
Finally, the 6-DOF pose estimation is performed by passing the fused features into a series of 512-node LSTM layers followed by three parallel Fully-Connected (FC) networks. The translation and rotation vectors are estimated separately as it showed to produce better results. Moreover, determining the Euler angles instead of the quaternions or the rotation matrix resulted in a faster convergence during training.
Training of the entire DeepTIO network is performed in two stages. In the first stage, only the hallucination network is trained, which is subsequently disabled in the second stage when the rest of the network is fitted.
Experiments were both conducted using a handheld thermal camera setup as well as on a Turtlebot 2 UGV. The handheld experiments were operated inside different public indoor locations as well as in a smoke-filled firefighter training facility. The handheld thermal camera used is a FLIR E95 thermal camera at 60 fps with a resolution of 464x348 and is combined with a XSens MTI-1 Series IMU. The hallucination network was trained using RGB-D images from an Intel RealSense D435 depth camera at 30 fps with a resolution of 848x480. For the handheld tests, no ground truth could be obtained, but instead, the VINS-Mono visual inertial SLAM algorithm is used as a reference. The Turtlebot 2 experiments were conducted inside a laboratory and a corridor using a FLIR Boson 640 at 60 fps with

a resolution of 640x512 in combination with the same IMU. The ground truth was obtained using a VICON Motion Capture system inside the lab and a Velodyne HDL-32E LiDAR with GMapping inside the corridor. The training data for the pose estimation was obtained using a combination of inertial and wheel odometry measurements.

ATE (M) FOR EXPERIMENT WITH HAND-HELD DATA

	ROVIO	VINet	Vanilla DeepTIO (**ours**)	DeepTIO (**ours**)
Corridor 1	6.2496	**1.8825**	2.1975	1.9333
Corridor 2	**0.3343**	1.1036	0.7122	0.5267
Large Office*	failed	4.4359	3.3088	**3.2648**
Library 1*	failed	5.2647	2.5698	**2.0532**
Library 2*	failed	1.6812	1.5741	**0.5735**
Mean	3.2920	2.8736	1.7502	**1.6703**

*There is time misalignment among sensors for about 1-2 second.

ATE (M) FOR EXPERIMENT WITH MOBILE ROBOT DATA

	VINet	DeepTIO (**ours**)	Inertial +Wheel
CPS Lab 1	1.3619	**0.3824**	0.0931
CPS Lab 2	1.1440	**0.2785**	0.0805
CPS Lab 3 (poor)	1.2225	**0.2250**	0.0033
CPS Lab 4 (dark)	1.8379	**0.3807**	0.0036
Corridor 1	2.4497	**0.8988**	0.1279
Corridor 2	0.8294	**0.7433**	0.3595
Corridor 3	1.4629	**0.7307**	0.2297
Mean	1.4726	**0.5199**	0.1282

Figure 2.8: DeepTIO experiment results in terms of the Absolute Trajectory Error (ATE) from the handheld camera (left) and the Turtlebot 2 (right) [12]

In figure 2.8, the Absolute Trajectory Error (ATE) for both the handheld and the Turtlebot experiments are displayed. In the handheld experiments, DeepTIO is compared to an ordinary monocular visual inertial algorithm, ROVIO, and a deep-learning monocular visual inertial algorithm, VINet. Both were performed using visual cameras. A version of DeepTIO without a hallucination network, Vanilla DeepTIO, is also incorporated into the comparison. ROVIO performed well inside the second corridor but failed in corridor 1 when textures in the environment were lacking. Furthermore, the algorithm failed completely when a significant timing misalignment was present between the different sensor data inputs. VINet is more robust to textureless environments but, however, can become very inaccurate in the presence of a sensor timing misalignment. On the other hand, both DeepTIO versions were not affected by the timing misalignment, showing that the usage of selective sensor fusion increases robustness against time synchronization errors. Overall, the DeepIO performed best out of the handheld data experiments. Moreover, the utilization of a hallucination network to enhance the number of usable features proved to help increase the accuracy of the odometry. During the experiments however, all four algorithms suffered from an inaccurate scale estimation, a problem common to monocular visual odometry systems. The main cause of this issue, according to the authors, is the variable speed while walking around the test environment with a handheld setup.
During the Turtlebot experiments, the camera setup was exposed to different lighting conditions. Because of this, VINet often performed worse in darker environments whereas DeepTIO, operating with a thermal camera, was not affected by these changing conditions. In general, for every driven test, the odometry estimated by DeepTIO was more accurate than VINet. DeepTIO also showed to be able to generate a robust trajectory when performing difficult manoeuvres such as U-turns and when a dynamic object would enter the frame.
In conclusion, the authors discussed that the implementation of an thermal inertial odometry algorithm using deep learning was successful. Furthermore, the hallucination network could provide meaningful side information such as to improve the accuracy with respect to the vanilla DeepTIO. The researchers did note that the frame rate at which the algorithm performs is limited to the frame rate at which it is trained. Deviating from this lead to a deterioration of the accuracy.
Machine learning-based solutions can offer higher performance compared to more conventional solutions for a specific task in a known environment. For a machine learning SLAM algorithm to offer flexibility, it is required for it to be trained in a vast set of environments with different thermal cameras and IMU sensors, which would be a very time- and resource-consuming task. As flexibility is one of the firefighters' requirements, a completely machine learning-based thermal-inertial SLAM system does not seem to be a suitable solution.

2.4 Benchmarkings of feature detection and matching in thermal images

From the review of current state-of-the-art thermal inertial odometry algorithms, it is clear that a common problem between them is the detection of robust features. In this section, an overview of two feature extraction and description benchmarks on thermal images will be given.

In a collaboration between the Royal Institute of Technology (KTH) in Sweden and FLIR Systems AB, J. Johansson et. al. [13] evaluated the performance of different visual-based feature detectors and descriptors on thermal images. The dataset was taken inside both structural and textured environments. Before evaluation, the images were rescaled to 8-bit and their histograms were equalized. The different algorithms were tested on different image deformations such as viewpoint change, rotation, scale, noise and downsampling. In figure 2.9, the list of the different detectors and descriptors is shown.

Detectors		Descriptors	
Hessian-Affine	SURF	LIOP	BRISK*
Harris-Affine	BRISK*	SIFT	FREAK*
MSER	FAST*	SURF	ORB*
SIFT	ORB*	BRIEF*	

Figure 2.9: Detectors (left) and descriptors (right) included in the evaluation by J. Johansson et. al. (binary types are marked with an asterisk) [13]

When evaluating the effect of a viewpoint change on different combinations of detectors and descriptors, it was revealed that both the MSER or Hessian-Affine detector combined with the LIOP detector performed best in the floating-point algorithms. Although often faster, binary types provided similar results, with the ORB-BRISK (descriptor only) and ORB-FREAK combinations outperforming other binary options based on BRISK (detector) and FAST.
The overall performance by all detectors and descriptors was higher when prompted by a image rotation compared to a change in viewpoint. The highest recall was obtained again by the LIOP descriptor, which in this case was again combined with the Hessian Affine detector closely followed by the combination with the Harris Affine detector. As expected, descriptors that are sensitive to rotation, such as BRIEF, exhibit here a poor performance. For the binary types, the ORB detector along with either BRISK or FREAK performed the highest in the presence of rotations as well as the ORB detector and descriptor.
Examining the different algorithms when matching features between images with different scales, among floating point combinations, the best performance was obtained by MSER-LIOP and Hessian Affine-LIOP. In terms of binary types, in addition to ORB with FREAK or BRISK again, also the combination SURF with BRIEF showed a high invariance to scale changes.

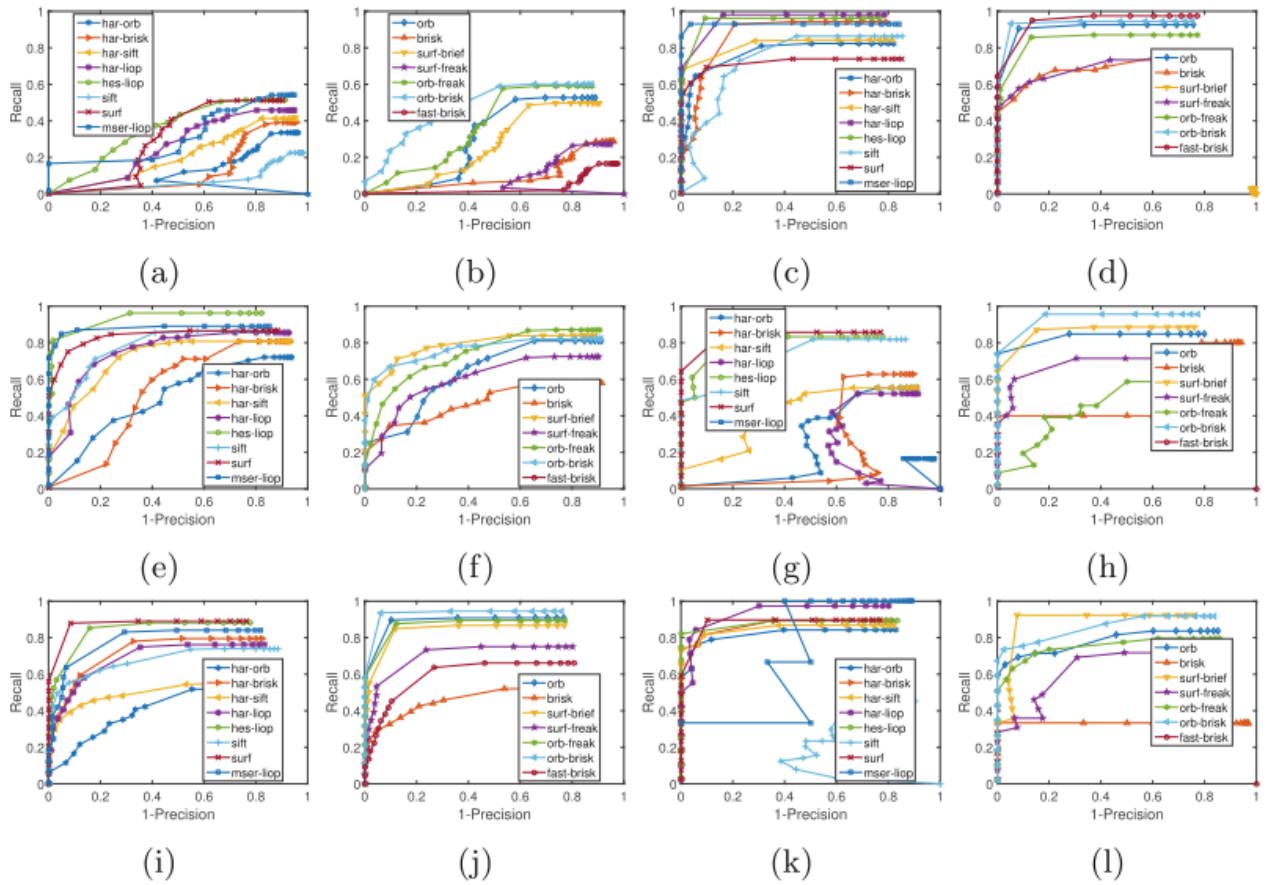


Figure 2.10: Performance against view-point (a) and (b), rotation (c) and (d), scale (e) and (f), blur (g) and (h), noise (i) and (j), downsampling (k) and (l) in structured scenes by J. Johansson et. al. [13].

When images are subjected to Gaussian blur, SURF showed to have the highest stability followed by SIFT and the Hessian Affine-LIOP combination among the float point algorithms. The overall highest performance was demonstrated in the binary types by the combination SURF-BRIEF. Again ORB-BRISK and ORB belonged to the highest performers.
In the presence of Gaussian white noise, identical conclusions concerning the binary types can be drawn as with Gaussian blur. For float point algorithms, SURF here also demonstrated high performance, followed again by Hessian Affine-LIOP.
Finally, on down-sampled images, the best performers are SURF, Hessian Affine-LIOP, Harris Affine-LIOP and Harris Affine-BRISK. Between the binary types, SURF-FREAK, ORB-BRISK and ORB obtained the best recall.
As a conclusion, the combination that obtained the best results over the different image deformations is the Hessian Affine detector with the LIOP descriptor . In general, feature detectors using affine transformations obtained better performance on rescaled thermal images than other float point feature detectors. Among the binary types, ORB with BRISK offered the most competitive results.

A second benchmarking of feature extraction and description methods of thermal features was performed by T. Mouats et. al. [49]. The targeted application was thermal odometry estimation and therefore the processing time was also considered in the evaluation. The used datasets consisted of a number of video sequences recorded both indoor and outdoor using a FLIR Tau2 LWIR camera.
It was concluded that, in general, blob detectors such as SIFT and SURF provided lower repeatability than corner detectors like FAST and GFTT. This means that when an object is seen from multiple perspectives, the likelihood is smaller that features will be located at the same location on the object. On the other hand, blob features showed more distinctiveness than corner features as the comparison of their descriptor yielded higher matching scores, i.e. more correct matches.

SIFT feature extraction performed the worse in almost all aspects. Feature extractors seem to not be affected by motion blur, with the SURF blob extractor performing the best. It was observed that the descriptors were less influenced by non-uniformity noise (NUC) than expected. The effect seemed however to be stronger in indoor environments, where objects often have a uniform temperature giving less contrast to the image and thus amplifying the noise.
The authors concluded by proposing SURF feature extraction in combination with the FREAK binary descriptor as the best combination for thermal navigation applications. The proposed combination offers a good trade-off between good matchability as well as repeatability while also providing a satisfactory computation time for real-time navigation.

3 Problem Analysis

The main goal of this project will be to try to provide improved situational awareness to the firefighters in a smoke-filled environment during a fire incident.
For the SLAM algorithm to be of use for the firefighters, it must satisfy the following requirements:

1. Independent of the type of UGV.
2. No wheel encoders available.
3. Functional "out-of-the-box", no setup required.
4. The system is self-contained, all processing is performed on-board.
5. Real-time operation for obstacle avoidance.
6. Precise and consistent odometry, the operator should be able, for example, to reverse the UGV on an already driven path without any collisions.
7. A 3D map containing the location of the obstacles in the environment.

In table 1, a comparison of the different features presented in the state-of-the-art thermal odometry algorithms reviewed in chapter 2.3 is given. From the table, it can be seen that most algorithms are monocular and only estimate the odometry. In terms of 3D mapping using thermal images however, no previous results were found in the literature apart from the obtained stereo disparity map from T. Mouats et. al. [10]. None of the thermal odometry algorithms are equipped with global bundle adjustment in the form of loop closure detections. Additionally, all algorithms were tested in benign environments with constant conditions and no testing results could be found of thermal odometry algorithms in a fiery setting. The experiments closest to this project were performed with DeepTIO [12] inside a firefighter training facility using smoke from a wood fire. The temperature however remained constant and no fire source was visible inside the dataset.

	Thermal	**Stereo**	**Inertial**	**Radiometric Data**	**Tested in smoke**	**Tested near fire**	**Disparity map**	**3D mapping**	**Loop closure detection**
T. Mouats et. al. [10]	Yes	Yes	No	No	No	No	Yes	No	No
Borges et. al. [46]	Yes	No	Yes	No	No	No	No	No	No
ROVIO, ROTIO, KTIO [41]	Yes	No	Yes	Yes	Dust, synthetic smoke	No	No	No	No
DeepTIO [12]	Yes	No	Yes	Yes	Yes	No	No	No	No
ORB SLAM 3 [14]	No	Yes	Yes	No	No	No	Yes	Yes	Yes
Firebot SLAM	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes

Table 1: Comparison of the different state-of-the-art thermal odometry algorithms presented in chapter 2.3 as well as a state-of-the-art visual odometry algorithm and this project's final approach

From the literature review in Section 2, it becomes apparent that the main challenge of thermal-inertial SLAM will be the ability to find robust features inside thermal images. As can be seen in figure 3.1, thermal images have a much lower contrast than visual images as well as significantly more noise making it harder to detect image features.
Many algorithms using thermal images resolve this problem by modifying the contrast of the rescaled thermal images. Although this is a viable option in environments where the temperature is relatively constant, it may result in the loss of features when sudden temperature changes are introduced into the frame. For example as explained by S. Khattak et. al. [11], when a significantly hotter temperature, such as from flames, is present in an incoming frame, it will appear very bright and may saturate the entire frame. On the other hand, if equalization is done dynamically, it may change the contrast too

drastically such that features cannot be matched anymore. In building fires, the hot smoke will be situated towards the ceiling, whereas colder air is located near the floor, as seen in Appendix A. To obtain a better contrast, a mask should be implement that conceals these hotter regions.

Figure 3.1: Examples of 8-bit rescaled thermal images in a benign environment (top: hallway, bottom: laboratory)

Once robust thermal image features are obtained, the determination of the robot's odometry as well as a sparse 3D map can be computed using identical techniques as in visual SLAM.

The objective of this thesis will be to assess the feasibility and accuracy of thermal-inertial SLAM inside an environment with fire and smoke.
Therefore, this research will be lead by the following question: Is it possible to utilize a stereo set of thermal cameras potentially alongside an IMU to, in a consistent and computationally cost-effective manner, localize a UGV and create a 3D map in a smoke-filled environment during a fire incident?
Alongside, the following questions will guide the study:

- What is the most efficient method to extract and match robust features from thermal images?
- Can the robust features be detected in-real time?
- Can features from thermal images be used for loop closure detection?
- How accurate can the thermal-inertial odometry algorithm estimate the robot's position?
- Can the thermal-inertial SLAM algorithm provide a 3D map with sufficient detail as to increase situational awareness in a smoke-filled environment?

Robust features are seen as features that are validated by the RANSAC outlier removal. The robot drives at a relatively low speed and therefore a high refresh rate of the SLAM output is not needed. Rather, four location updates accompanied with at least a sparse point cloud per second would be enough to give the operator, along with a live video feed of the thermal cameras, additional situational awareness of the environment around the vehicle. To achieve this, the feature matching process should then be able to process 8 frames per second, leaving ample time for the SLAM estimation itself. In order to obtain an accurate odometry estimation, a minimum number of 50 3D-points is required although 200 points are recommended [50].

The main novelty of this thesis will be to determine the optimal method to create a 3D map of an environment inside a smoke-filled environment. During this project, the first complete 3D map

constructed from thermal images will be attempted. The developed thermal-inertial SLAM algorithm is the first to be able to perform global bundle adjustment by recognizing loop closures. The system is the first one of its kind to be tested in a real scenario featuring fire and smoke. The accuracy of the odometry will be measured using a motion capture system.

4 Method

The research of this thesis will be divided into two sections. The first part will consist of finding suitable feature extraction and description methods capable of providing robust features. As mentioned in section 3, this part is one of the current challenges when working with thermal images. The different feature extractors and descriptors will be compared during a benchmarking process on datasets containing smoke and fire recorded at a firefighters training facility.
After the benchmarking, the best candidate will be implemented inside an existing SLAM algorithm. An overall performance evaluation will then be performed to assess the possibility of thermal-inertial SLAM inside a fiery environment for increased situational awareness. This second part includes evaluating the estimated path driven by the robot, the detection of loop closures and finally the quality of the obtained 3D point cloud.
The remaining of this section will start with an overview of the materials available during the project, including the datasets and the hardware. This part is complemented with the calibration of the thermal cameras and the IMU. Before explaining the approach of the feature extraction and description, the implementation of the author's variant of a gradient-based feature tracking algorithm as well as CNN-based feature descriptors is described. The explained approach for the feature extraction and description benchmarking will contain the different criteria used to evaluate as well as the different candidates participating.

4.1 Hardware and dataset acquisition

The dataset used during this project was recorded inside a mock-up parking garage at the Twente Safety Campus training facility in Enschede, the Netherlands. As can be seen in figure 4.1, The parking garage contained a number of vehicles as well as a shipping container. Smoke was generated using burning firewood inside two oil barrels as well as a third wood fire inside one of the vehicles. Three datasets were captured at different levels of room temperatures and smoke levels. Each dataset respectively includes 1504, 1217 and 2508 stereo thermal pairs obtained at a rate of 20 frames per second. The image pairs are accompanied by IMU measurements recorded at 100 Hz. The acquisitions were performed in the morning in early December when outside temperatures just above freezing point.

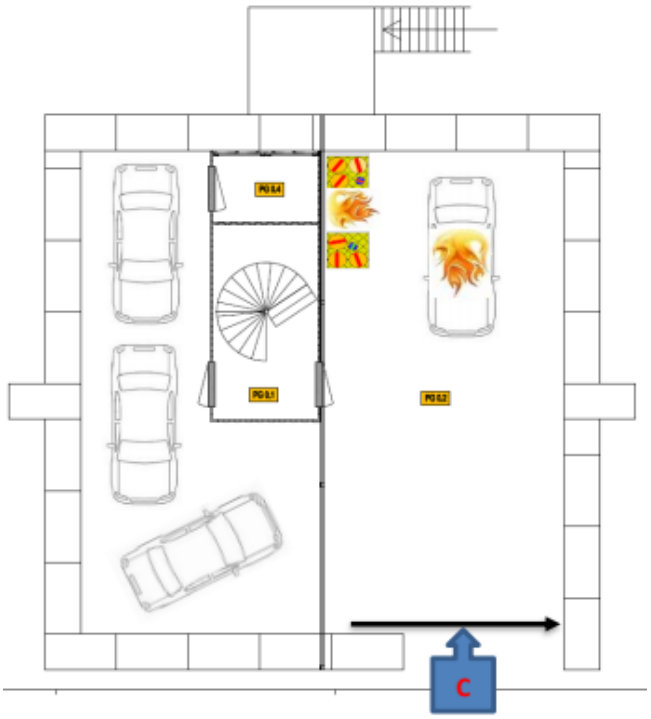


Figure 4.1: Layout of the mock-up parking garage at the Twente Safety Campus

4.1.1 Hardware setup

During this project, the following equipment will be available:

- 2x FLIR Boson 640, 30 fps, 640 × 512, 95° horizontal field of view.
- XSens Mti-620 IMU, 100 Hz.
- Dr. Robot Jaguar 4x4 remote-controlled vehicle.
- On-board Intel NUC7i7BNHX1 CPU, i7, 2 cores (4 threads), 3.5 GHz (deployment).
- Laptop Intel i7-8550U CPU, 4 cores (8 threads), 1.8 GHz (development).
- Intel Xeon Gold 6144 CPU, 4 cores (8 threads), 3.6 GHz (Machine learning workstation).
- NVIDIA Titan XP GPU, 12 Gb VRAM, 11.4 Gb/s (Machine learning workstation).

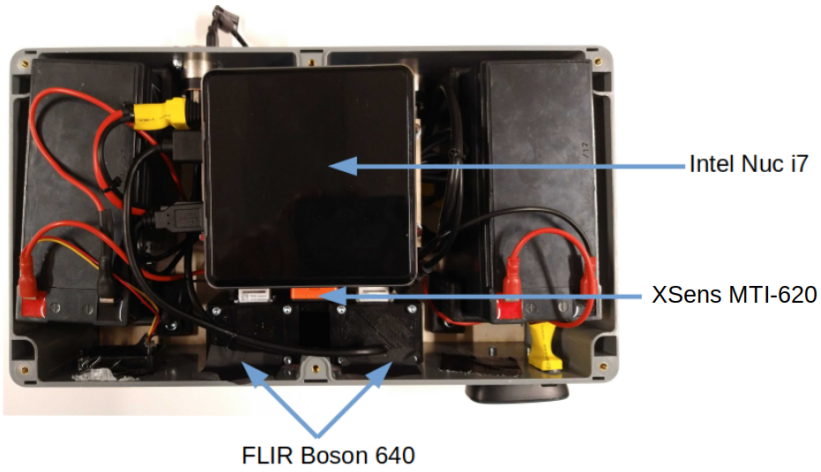


Figure 4.2: Labeled top view of the data acquisition setup

The recording setup can be seen in figures 4.2 and 4.3. The box containing the hardware needed for the acquisition was mounted on top of the vehicle.
The thermal cameras are installed with a baseline of 80 millimeters. The thermal images are retrieved and synchronized at a mean rate of 20 frames per second. The synchronization of stereo pairs is performed by retrieving the latest image of each camera. As the actual capture time is not available, the retrieval timestamp is then assigned to the stereo pair. The raw output of the thermal cameras is 14-bit radiometric data which has been bit-shifted to obtain a 16-bit image. The current FLIR Boson cameras do not have any radiometric cores, meaning that they do not have the ability to measure their own temperature and thus cannot compute the absolute temperature of the environment. Although the absolute temperature can give useful insight into the range of interest for conversion into 8-bit images, the data that is of most interest are the raw measurements of long-infrared radiation emission. The latter will be used in raw 16-bit form with the gradient-based feature tracking and converted to 8-bit using the method in section 4.1.2 to be utilized by the remaining feature extraction and description methods.
The IMU data was measured using a XSens Mti-620 mounted on the camera mount between the two cameras, at a rate of 100 Hz.
The SLAM algorithm is required to operate in real-time on-board the vehicle once the system will be deployed. During this project, in the development phase, the system will be tested on a similar CPU mounted inside a laptop. The CPU used for development has twice the number of threads but runs at a clock speed that is half of the Intel NUC. In the case of feature extraction and matching, performance during deployment should therefore improve as no multi-threading is utilized during this task. Whether a negative difference in performance will be observed during deployment will depend on the selected visual SLAM algorithm used for optimization and the number of threads it will require. Testing and benchmarking of the CNN-based descriptors will be executed on a desktop workstation possessing a GPU. Again no multi-threading is here utilized and the number of threads is therefore irrelevant. The clock speed of the CPU is very similar to the one of the deployment CPU.

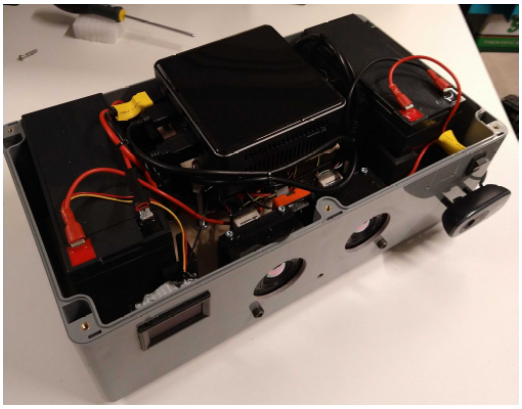


Figure 4.3: Side view of the data acquisition setup

4.1.2 Linear Automatic Gain Control

Color intensity display in the visual spectrum is limited to 8-bit in most modern computer screens. As a result, almost all state-of-the-art computer vision algorithms are designed to only work with 8-bit images. As mentioned previously, applying a linear Automatic Gain Control (AGC) to convert a 16-bit images to 8-bit leads to information loss and possible saturation due to temperature change. In practice, only a small portion of the 16-bit spectrum is non-zero in benign environments. To restrict the information loss, only the dynamic range of interest will therefore be converted to 8-bit. Additionally, the brightness of the images will be automatically corrected to limit changes between consecutive frames due to shifts in the dynamic range and accumulation of non-uniform noise.

The dynamic range of the 16-bit radiometric data is determined using its cumulative histogram of 65535 bins. The dynamic range is defined as $(p_l,\ p_u]$ where p_l is the defined lower bound of the cumulative percentage of 16-bit values and p_u the defined upper boundary of the cumulative percentage of pixels to be included. The 8-bit value I_8 at the image location (u, v) is then linearly transformed from the 16-bit intensity I_{16} as follows:

$$I_8(u,v) = 255 \cdot \frac{I_{16}(u,v) - I_{p_l}}{I_{p_u} - I_{p_l}} \tag{9}$$

where I_{p_l} and I_{p_u} respectively correspond to the 16-bit intensity associated with p_l ad p_u.
As significantly hotter objects compared to the environment are introduced into the frame, the dynamic conversion method above leads to a darkened scene. To avoid such a phenomenon, the upper intensity value I_{p_u} is limited to a maximum value I_{max} which is determined from practical observations, demonstrated in section 5.1. Correctly tuning the ceiling threshold I_{max} for the dynamic range is of high importance. If it is set too high, the range to be converted might be too large in areas where intense heat is present, causing the cold environment to be darkened making feature detection more challenging. On the other hand, setting the threshold too low may result in the suppression of useful data.

The brightness of the obtained 8-bit thermal image is adjusted to a constant pre-defined value. As a first step, the 8-bit image is transformed into the HSV color space, where the V-channel is the image brightness between 0.0 and 1.0. Then, the mean image brightness $\bar{b}$ is obtained by computing the mean value:

$$\bar{b} = \frac{1}{n_r \cdot n_c} \sum_u \sum_v V(u,v) \tag{10}$$

where n_r and n_c are respectively the number of rows and columns in the image.
The gain g is multiplied with the 8-bit gray-scale intensities to adjust the brightness is finally equal

to the ratio of the desired brightness b_d over the mean image brightness $\bar{b}$:

$$g = \frac{b_d}{\bar{b}} \tag{11}$$

4.1.3 Camera calibration and image rectification

The calibration dataset was obtained using a heated plywood board containing an acircular calibration pattern cut using a laser cutter. The board is fabricated out of birch plywood and possesses an acircular pattern of 4×11 with a step of 43 mm in both the horizontal and vertical direction. To prevent false detections due to background noise, the calibration board is clamped onto an additional sheet of plywood after being heated. To achieve an equal heat distribution across the board, it is heated with the help of an infrared panel.

Figure 4.4: Heating the calibration pattern (left) and calibration layout after heating (right)

Because of the camera's wide field of view, it is important to classify the lens before the calibration in order to select the correct reprojection and distortion model. Based on the focal length and the sensor size, it was concluded that the lens with a horizontal field of view of 95° is an ultra wide-angle lens. Its focal length is however not short enough such that a fisheye lens model can be applied. Therefore, the classical pinhole reprojection model is used. The full reasoning can be found in Appendix D.

The camera calibration was performed using the camera calibration module included in the *image_pipeline* ROS package [51]. The calibration process itself inside the package is performed by the OpenCV library, but the *image_pipeline* includes some additional metrics used to evaluate the detection of the calibration pattern, offering more certainty for reliable results.
The ROS camera calibration tool starts by extracting the calibration grid across the stereo dataset. For each successfully extracted grid, the tool tries to prevent motion blur by rejecting large shits in image position of the grid between consecutive frames. Furthermore, it also rejects images where the calibration board is warped by computing the board's skewness. Finally, once a good selection of calibration boards has been extracted, the camera calibration module computes the camera intrinsic parameters, the camera matrices and lens distortions, and the extrinsic parameters, the transformation between the cameras in the stereo pair.

The focal length computed by the calibration is expressed in units of pixels. In case of sensor pixels with square dimensions, the following holds:

$$\frac{f}{f_x} = \frac{W}{w} = \frac{f}{f_y} = \frac{H}{h} \tag{12}$$

where f is the focal length in millimeters, f_x, f_y are the focal lengths in pixels in image x- and y-direction respectively, W, H are the width and height of the image sensor in millimeters and w, h are

the image width and height in pixels.
As $W = w \cdot p$ and $H = h \cdot p$ where p is the pixel pitch, i.e. size, equation (12) can be rewritten as:

$$\frac{f}{f_x} = \frac{f}{f_y} = p \tag{13}$$

With the FLIR Boson 640 cameras having a focal length $f = 4.9$ mm and a pixel pitch $p = 0.012$ mm [52], the expected focal lengths from the camera calibration are therefore:

$$f_x = f_y = \frac{f}{p} = 408 \tag{14}$$

In theory, the principle point $(c_x,\ c_y)$ lies in the center of the image. Although in practice it often slightly deviates, the principle should always be near the enter of the image. The ideal intrinsic camera matrix K from the camera calibration is then:

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 408 & 0 & 320 \\ 0 & 408 & 256 \\ 0 & 0 & 1 \end{bmatrix} \tag{15}$$

Note that in reality, the obtained intrinsic parameters may differ from the expected values due to calibration errors and manufacturing deviations.
The results from the camera calibration will be quantified by computing the RMS-value e_{RMS} of the reprojection error between two undistorted images, computed using the residual of eq. (1):

$$e_{RMS} = \sqrt{\frac{1}{N_c} \sum_{i=1}^{N_c} (p_i' \cdot F \cdot p_i)^2} \tag{16}$$

where p, p' are respectively an undistorted image point in homogeneous coordinates in the left and right image, N_c the total number of circles detected across the images and F is the fundamental matrix. The fundamental matrix F describes the epipolar geometry, i.e. the geometry between a 3D-point and its projected point in two images, in image coordinates and is related to the essential matrix E by eq. (17)[53]:

$$F = \left(K_r^{-1}\right)^T \cdot E \cdot K_l^{-1} \tag{17}$$

where the subscript l and r respectively denote the left and right camera.
Finally the essential matrix E, describing the epipolar geometry in world coordinates, can be obtained as in eq. (18)[53], by multiply the rotation matrix with the skew-symmetric representation of the translation matrix between the two cameras:

$$E = R \cdot t_\times \tag{18}$$

where R and $t_\times$ are respectively the rotation matrix and the skew-symmetric representation of the translation from the left to the right camera obtained from the camera calibration.
As a rule of thumb, a RMS-value under 1.0 is generally taken to qualify as the minimum value required for a camera calibration to be acceptable, meaning that the reprojection error is on average below a pixel.

After the camera calibrations are obtained, the images are undistorted, meaning that the lens distortion is removed. Finally, the images have to be rectified, which entitles that the planes of both left and right images are rotated such that they are parallel and share a common focal length. As a result, the corresponding epipolar lines are horizontal meaning that two matching image points lay on the same image row.
In practice however, this case might slightly deviate. To verify this, the calibration dataset is rectified

and the RMS-value of the difference in y-coordinates of the detected circles between the stereo pair will be computed as follows:

$$\Delta y_{RMS} = \sqrt{\frac{1}{N_c}\sum_{i=1}^{N_c}(y_l - y_r)^2} \tag{19}$$

where y_l and y_r are the y-coordinate of a circle center in the calibration board in the left and right image respectively. Note that the circles in the image are only included if the entire calibration grid has been detected.
Additionally, the rectification can be visually inspected by drawing horizontal lines at fixed intervals onto the rectified stereo pair. If the pair is correctly rectified, each line should intersect the same points in both images.

4.1.4 IMU calibration

Along with providing 6-DOF inertial measurements, the XSens Mti-620 IMU also provides on-board calibration parameters such as biases, scale factors and axes misalignment that are directly applied to the raw data. The XSens Mti-620 possesses a thermometer, which is used to compensate the different calibration parameters for temperature changes while operating.
What is left to determine are the noise density and random walk of both the accelerometer and the gyroscope. Both noise parameters can be extracted from the manufacturer's provided Allan curves which describe the standard deviation σ observed in the measurement according to the time cluster size τ. The UGV is a rigid body with two degrees of freedom, namely a translation in the body-fixed forward direction, the z-direction and a rotation around its azimuth, the z-axis. Hence, the random walk on the of these two measured motions will suffice.

The velocity/angle random walk K_v, or noise density is the random walk of the integrated measured acceleration or angular rate. K_v can be obtained by drawing the tangent line of the Allen curve where the slope is equal to -0.5 in the log-log space. K_v is then equal to the intersection of the tangent line and the vertical line $\tau = 1s$ [54]. Doing so in figure 4.5 results in a velocity random walk coefficient of $0.000438\ m \cdot s^{-1.5}$ and an angular random walk of $1.51 \cdot 10^{-6}\ rad \cdot s^{-0.5}$ in figure 4.6.

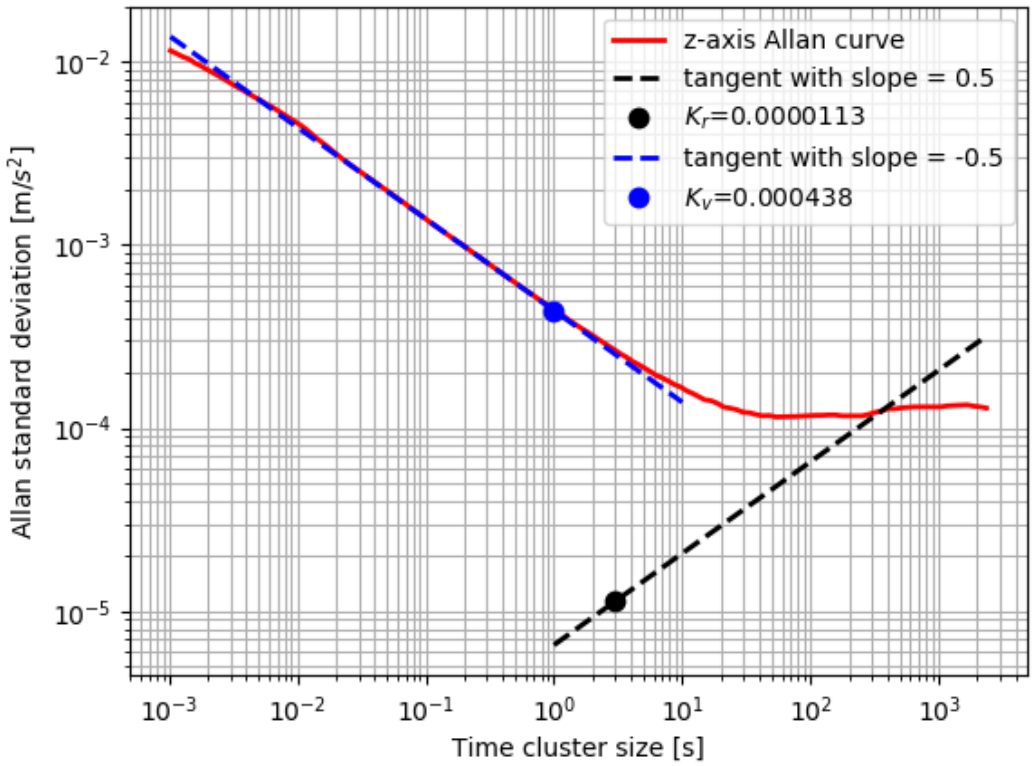


Figure 4.5: Allan curve of the z-axis in the XSens Mti 600-series accelerometer (courtesy of XSens) and determination of the velocity random walk K_v

The rate random walk is obtained similarly to the velocity or angle random walk above. Taking the tangent of the Allan curve where the slope is now equal to 0.5 in the log-log space, K_r is equal to the intersection between the tangent and the vertical line $\tau = 3s$ [54]. Performing this in figure 4.5 yields an acceleration random rate of $1.13 \cdot 10^{-5}\ m \cdot s^{-2.5}$. Similarly, from figure 4.6, it can then be observed that the gyroscope's rate random walk is equal to 0.3186 $rad \cdot s^{-1.5}$.

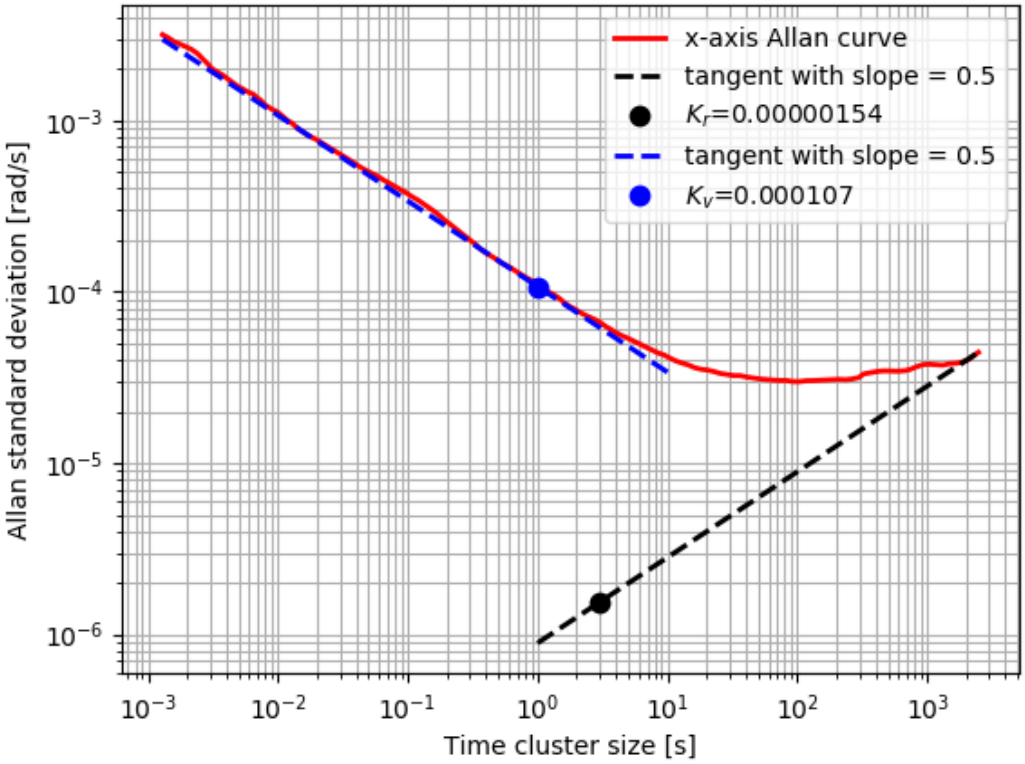


Figure 4.6: Allan curve of the x-axis in the XSens MTi 600-series gyroscope (courtesy of XSens) and determination of the rate random walk K_r

In table 2, a summary of the noise density and rate random noise parameters obtained above is provided.

Sensor	**Noise density**	**Rate random walk**
Accelerometer forward z-axis	$4.38 \cdot 10^{-4}\ m \cdot s^{-1.5}$	$1.13 \cdot 10^{-5}\ m \cdot s^{-2.5}$
Gyroscope heading x-axis	$1.07 \cdot 10^{-4}\ rad \cdot s^{-0.5}$	$1.54 \cdot 10^{-6}\ rad \cdot s^{-1.5}$

Table 2: Noise density and rate random walk of the XSens 600-series

The XSens IMU compensate for accelerometer and gyroscope errors such as bias, scale factor and misalignment based on the temperature measured with the on-board thermometer. However, some rest errors might still remain and an IMU calibration to estimate these three error parameters will therefore be performed.
The actual body-fixed acceleration $\vec{a}_b$ in m/s^2 and angular rate $\vec{\omega}_b$ in rad/s are respectively related to the measured acceleration $\vec{a}_m$ and angular rate $\vec{\omega}_m$ as follows:

$$\vec{a}_m = (I_{3\times3} + S_a + M_a) \cdot \vec{a}_b + \vec{b_a} + \vec{n}_a \tag{20}$$

$$\vec{\omega}_m = (I_{3\times3} + S_g + M_g) \cdot \vec{\omega}_b + \vec{b_g} + \vec{n}_g \tag{21}$$

where $I_{3\times3}$ is the identity matrix, S is the scale factor error matrix, M is the axis misalignment matrix, b is the bias and $\vec{n}$ the Gaussian white measurement noise.

To obtain these parameters, measurements are taken at standstill with the axis in question pointing both in positive and negative direction of gravity. The accelerometer and gyroscope bias of the i^{th} axis can then respectively be computed using eq. (22) and eq. (23):

$$b_{a_i} = \frac{\bar{a}_{p_i} + \bar{a}_{n_i}}{2} \tag{22}$$

$$b_{g_i} = \frac{\bar{\omega}_{p_i} + \bar{\omega}_{n_i}}{2} \tag{23}$$

where $i = 1, 2, 3$, $\bar{\cdot}$ signifies the mean value and the subscripts p and n respectively the positive and negative direction of gravity.

The i^{th} diagonal component of the scale factor matrices S_a and S_g can be obtained in a similar manner:

$$S_a(i,i) = \frac{\bar{a}_{p_i} - \bar{a}_{n_i} - 2g_l}{2g_l} \tag{24}$$

$$S_g(i,i) = \frac{\bar{\omega}_{p_i} - \bar{\omega}_{n_i} - 2\omega_e \sin(\psi_l)}{2\omega_e \sin(\psi_l)} \tag{25}$$

where g_l is the local gravity in m/s^2, ω_e is the rotational speed of the Earth in rad/s and ψ_l is the local latitude. The non-diagonal components of S are equal to zero.
Finally, the misalignment error of the i^{th} axis with respect to the j^{th} axis is computed as:

$$M_a(i,j) = \frac{\bar{a}_{p_{ij}} - \bar{a}_{n_{ij}}}{2g_l} \;\forall\; i \neq j \tag{26}$$

$$M_g(i,j) = \frac{\bar{\omega}_{p_{ij}} - \bar{\omega}_{n_{ij}}}{2\omega_e \sin(\psi_l)} \;\forall\; i \neq j \tag{27}$$

where $i, j = 1, 2, 3$ and the subscript ij signifies the mean value measured along the j^{th} axis as the i^{th} axis in in positive or negative direction of gravity. In the case of $i = j$, $M(i,j) = 0$.

4.1.5 Camera-IMU calibration

The calibration of the camera-IMU setup consists of determining the relative transformation from the origin of the IMU to the principle point in the image of the left camera. This can be achieved by solving the following equation:

$${}^{c_l}\vec{T} = \begin{bmatrix} {}^{c}R_b & t_\times \cdot {}^{c}R_b \\ 0_{3\times 3} & {}^{c}R_b \end{bmatrix} {}^{b}\vec{T} \tag{28}$$

where ${}^{c}R_b$, $t_\times$ are respectively the unknown rotation and skew-symmetric translation matrix from the IMU origin to the principle point of the left camera and ${}^{c_l}\vec{T}$, ${}^{b}\vec{T}$ are respectively the twist vector of the system expressed in the coordinate frame of the left thermal camera and in the body-fixed frame of the IMU. The twist vector $\vec{T}$ is defined as the combination of the linear velocity vector $\vec{v}$ and the angular velocity vector $\vec{\omega}$:

$$\vec{T} = [v_x \; v_y \; v_z \; \omega_x \; \omega_y \; \omega_z]^T \tag{29}$$

Eq. (28) will be solved in two parts, starting by estimating the rotation matrix R using the Procrustes rotation problem. The translation between the IMU and camera will be retrieved from a 3D-CAD

model of the thermal-inertial setup. In the case of this project, the accuracy of this translation does not need to be precise to the millimeter. The reason behind this is that the effect of the translation is only noticeable through the acentric rotation in the measured acceleration of the IMU with respect to the camera. This effect is equal to the upper right part of eq. (28). As the arm, i.e. the translation vector, between the two sensors is relatively small, the effect will also be small. Therefore, obtaining an accurate estimation to the millimeter of the translation is very difficult due to the different additional components included in the measured acceleration such as gravity and noise [55]. The estimated rotation between the two sensors is on the other hand more of importance as it contributes to the gravity compensation in the measured acceleration when estimating the orientation between the world (camera) reference frame and gravity's reference frame.

The rotation ${}^{c}R_{b}$ can be obtained by solving the lower right part of eq. (28), i.e. the angular rate transformation. The different steps taken to achieve this are summarized in figure 4.7.

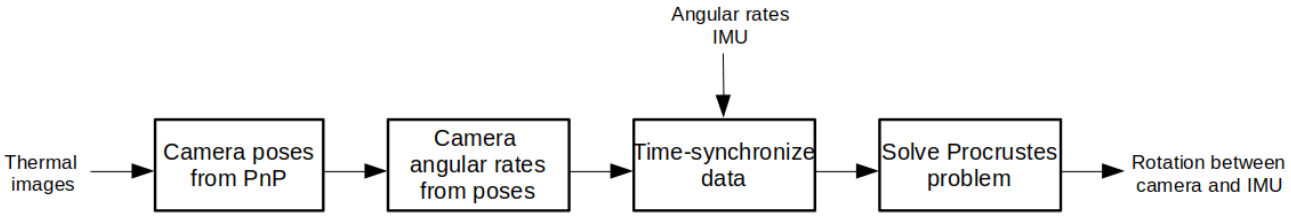


Figure 4.7: Flow diagram of the IMU-camera rotation estimation

The angular rates of the left thermal camera is obtained by first computing the pose R_c of the camera using Perspective from n-Points (PnP) on the heated calibration board. The skew-symmetric matrix Ω of the camera's angular rates can then be computes between two poses at moments t_{k-1} and t_k as follows:

$$\Omega_k = \frac{1}{2(t_k - t_{k-1})} \cdot \frac{\theta_k}{sin(\theta_k)} \cdot (A_k - A_k^T) \tag{30}$$

where A_k and θ_k are equal to:

$$A_k = R_k R_{k-1}^T \tag{31}$$

$$\theta_k = \cos^{-1}\left(\frac{tr(A_k) - 1}{2}\right) \tag{32}$$

where $tr(\cdot)$ is the trace of the matrix.
Finally, the camera's angular rates around its three axes can be directly extracted from Ω:

$$\vec{\omega}_{c_l} = \begin{bmatrix} -\Omega(2,3) \\ \Omega(1,3) \\ -\Omega(1,2) \end{bmatrix} \tag{33}$$

As the IMU and the camera are rigidly-connected, it holds that the norms of their angular rate $|\omega|$ are equal. Before being able to solve the rotation equation between the sensors, the data needs to be synchronized in time. Taking the Fast Fourier Transform (FFT) of the norm over time of both angular rates, the time delay can be obtained by analyzing the phase shift between them in the frequency domain. This method of time-series synchronization is implemented in the *Syncing* Python module by V. Arcidiacono [56] and will therefore be utilized.
After the data is synchronized in time, the rotation matrix ${}^{c}R_{b}$ between the sensors can be estimated by Least-Squares method by solving the Procrustes problem using Singular Value Decomposition (SVD) [57]:

$${}^{c}R_{b} = UDV^T \tag{34}$$

where U and V^T are obtained from the SVD of $\vec{\omega_{c_l}}\vec{\omega_b}^T$ and D is the matrix preventing mirroring:

$$D = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & det(UV^T) \end{bmatrix} \tag{35}$$

Note that it is assumed that, in order to calibrate the camera-IMU system, the cameras have been calibrated beforehand and that $\vec{\omega_b}$ has been corrected for bias, scale factor and misalignment errors.

4.1.6 Acquisition

Recording a ground truth of the robot's position is not possible at the Twente Safety Campus as GPS or an optical tracking system such as Optitrack or VICON are not available. The dataset can therefore not be used to evaluate the accuracy of a TIO algorithm. Instead, the acquisitions at the TSC can only be used to make a qualitative assessment on the possibility of thermal-inertial SLAM inside an environment with smoke and fire.

Between each acquisition, the fire is fueled to further increase the production of fire. Both fire sources are fueled using fire wood. As wood has a high combustion level, it produces thinner smoke compared to materials typically found in parking garages. Thicker smoke particles from car materials like composites increase the chances of reflecting infrared rays, possibly reducing the thermal cameras' ability to see through smoke. However, despite the thinner smoke present in this project's dataset, the effects of the smoke's thermal radiation on both the environment and the thermal cameras can still be evaluated.

To acquire an indication of the accuracy obtained by the proposed thermal-inertial SLAM algorithm, recordings will also be executed inside a laboratory room equipped with an Optitrack motion capture system. To increase the texture of the room, it is filled with objects of different materials such as a metal table, a radiator, a heating panel covered with a circular calibration pattern, a window and a number of foam boards placed in various orientations against one of the walls. The objects are spaced across the room such as to create distinct locations for loop closure detections.

4.2 Gradient-based feature tracking

Along with the classical feature detection and description algorithms commonly found in the OpenCV library, a gradient-based feature tracking algorithm based on the one from KTIO [47] is also included in the benchmarking. For the initial testing, only a version capable of tracking features along the epipolar line in a stereo pair is implemented.
In this method, features are extracted according to their gradient value. These features are then tracked in the second image using template matching via cross-correlation on the gradient image. For comparison, gradient-based feature tracking will be both implemented on raw 16-bit radiometric data as well as on the converted 8-bit thermal images.
The gradient-based feature extraction in KTIO [47] utilizes an image pyramid to extract and track features across two consecutive monocular frames. Different image scales are used to increase the tracking accuracy initialized by IMU measurements. Here, using stereo pairs, the location of the features are more predictable as they lie on (or near) their corresponding epipolar line. A single image scale will therefore suffice to characterize the viability of this feature tracking method. The full flowchart of this algorithm is depicted in figure 4.8

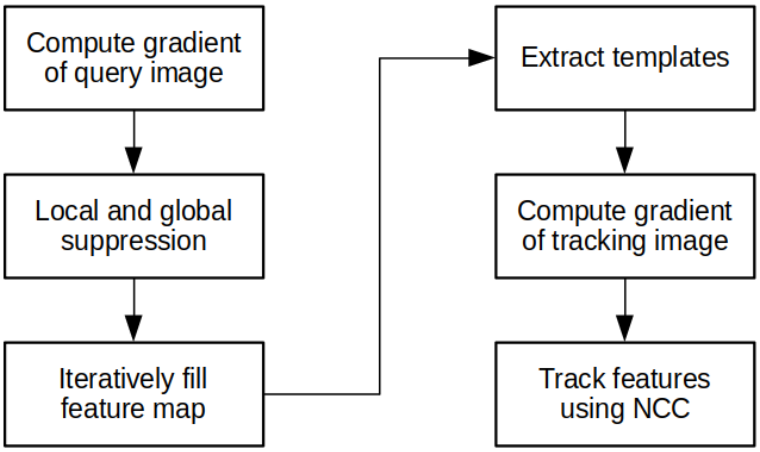


Figure 4.8: Gradient-based features flowchart

To extract features based on their gradient-value, the gradient of the image is first computed using a Sobel filter in both horizontal and vertical direction. The gradient amplitude image is then divided into blocks of 32×32. For each block, low gradient amplitudes are suppressed by thresholding based on a global offset and the block's median gradient amplitude. The offset ensures a minimum global quality of the feature candidates whereas the block median threshold guarantees the local quality of points.
First, the maximum gradient in each block is prioritized and taken as a feature. The remaining candidates then have equal chance to be selected. To do so, using an iterative process, each point is inserted onto a global feature map. To ensure that features in proximity with each other are not selected, a filled circle mask of radius d_{min} and with the feature point as center is drawn onto the feature map. New candidates within the circle can than not be selected as feature points, ensuring a more uniform distribution of the gradient features.

To find the feature locations inside the second image of the stereo pair, gradient template matching along the epipolar lines of the rectified image is performed. Because the image may not be perfectly rectified, the search area is extended to a number of rows above and below the epipolar line. The tracked location of a feature is taken as being the location with the maximum normalized cross-correlation (NCC) within the search area, given that the value is above a minimum acceptance threshold. If this is not the case, the feature is discarded.

4.3 CNN-based feature matching

With the increasing trend in recent years of utilizing machine learning for different image processing tasks, its suitability for matching features from thermal images will also be assessed during this project. A Convolutional Neural Network (CNN) will be trained to create a unique descriptor based on an image patch around the feature. The distance between a feature pair can than measured using the L2-norm to determine the most-suitable matches in a pair of images.
Both the training and deployment are performed using Tensforflow 2.3 and Keras 2.4, meaning that, contrary to the rest of this project, this section is implemented in Python. The latter may therefore raise some concerns as for real-time execution as Python is notoriously slower than C++.

4.3.1 Model architecture

Training of the CNN-based descriptors is performed using the siamese CNN network in figure 4.9. A siamese network consists of two parallel CNN models which extract interest regions from the feature image patch. Both models are identical, meaning that their composition and weights are shared. For both features, the output filters from the last CNN layer are then flattened and compressed into a 1-dimensional descriptor vector. The L2-norm between the two descriptors is finally computed and

transformed into a similarity measure between 0.0 and 1.0.

The architecture used to compute the descriptors has been taken from *Image Feature Matching Based on Deep Learning* by Y. Liu et. al. [58]. The CNN architecture is composed of three sequential pairs of convolutional layers separated with a batch normalization layer. All three pairs have a kernel size of 3×3 and the number of filters is doubled after each pair, starting at a depth of 32. The three pairs are followed with a dropout layer and finally a last convolutional layer with a larger kernel size of 8×8 and 128 filters together with a final batch normalization layer. Both the batch normalization layers and the dropout layers prevent overfitting the model to the training data and are therefore deactivated during deployment. The former re-centers the input filters by subtracting its mean and re-scales it by its variance whereas the latter randomly sets different inputs to zero at a given iteration. The non-zero elements are then scaled such that the sum of all inputs remains unchanged. All convolutional layers are activated using a Rectified Linear Unit (ReLU) function.
After the last batch normalization, the different filters are flattened together into a single 1-dimensional vector which serves as the input to the fully-connected dense layer that compresses the data into the 1-dimensional feature descriptor.

During training, the L2-norm between the two descriptors is computed to obtain the descriptor distance. The final dense layer finally re-scales the descriptor distance into a similarity score between 0.0 and 1.0 using the logistic function in such a way that a clear boundary between matching and non-matching features is created.
A more in-depth view of figure 4.9 can be found in figure E.1 of Appendix E.

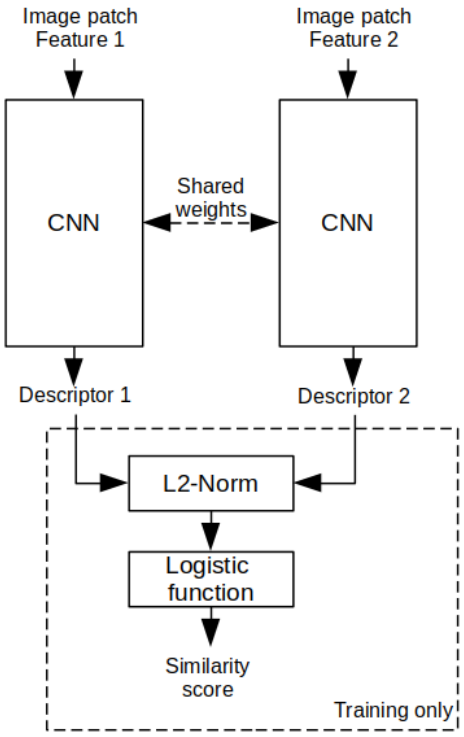


Figure 4.9: Siamese CNN architecture for feature matching

4.3.2 Training dataset

The training dataset is composed of a total of 56872 feature pairs divided equally over the two match and non-match classes. For each class, 1620 pairs have been manually-selected in stereo images from multiple datasets recorded during this project inside a laboratory and at the Twente Safety Campus. The remaining 26816 pairs for each class have been generated using the KAIST Multispectral Pedes-

trian Detection Benchmark campus dataset [59], where features are detected and matched using ORB on monocular RGB-image pairs. The matched feature pairs are then projected onto the LWIR thermal images in order to extract the features' region of interest. The corresponding RGB and thermal images are loosely aligned, meaning that both images do not perfectly overlap. However, because the misalignment is consistent across the dataset, this problem does not affect the extraction of the images patches for the training data.
For each image, the non-matches were created by randomly shuffling the regions of interest.
In order to experiment with different levels of image information to construct the feature descriptor, patches of both sizes 32×32 and 64×64 are extracted.

4.3.3 Training

The optimization of the network weights is conducted using the ADAM optimizer. The ADAM optimizer is a state-of-the-art stochastic gradient descent-based optimizer which requires little tuning of its hyper-parameters for a model to converge rapidly to a solution [60], making it very suitable for prototyping. As a trade-off, the ADAM optimizer has a higher risk of not fully converging to the solution under certain conditions [61]. During the feasibility analysis performed during this project however, this issue is negligible.
As the loss function, the contrastive loss function in eq. (36)[62] is utilized. This loss function puts more of an emphasis on the predicted similarity measurement between the two input patches itself than more popular classification loss functions such as the binary cross-entropy. Given the true match label $y = \{0, 1\}$ of the feature pair and the predicted similarity distance $0.0 \leq \hat{y} \leq 1.0$, the contrastive loss L is defined as:

$$L(y, \hat{y}) = \frac{1}{2}(1 - y)\hat{y}^2 + \frac{1}{2}\max(m - \hat{y}, 0)^2 \tag{36}$$

where m is a margin generally set to 1.0.

To both further prevent overfitting the model to the training dataset as well as avoid large weights and gradients, weight regularisation is also applied to the loss function. L1-regularization adds an element proportional to the sum of the weights to the loss whereas L2-regularization adds an element proportional to the sum of the squared weights. The former, on top of penalizing high weights, pushes the weights to zero. This increases the sparsity by removing meaningless data and thus reducing the model complexity. L2-regularization on the hand further penalizes high weights at the cost of less sparsity. During this project, the information provided by thermal images is already limited and thus further reducing it is not desired. Therefore, L2-regularization will suffice.

The quality of the trained model will be assessed using a Receiving Operating Characteristic (ROC) curve. The ROC-curve represents the True Positive Rate (TPR), i.e. the true matching pairs classified as being matches, with respect to the False Positive Rate, i.e. the pairs being wrongly classified as matches, according to various classification thresholds θ. The TPR and FPR are computed respectively as follows:

$$TPR(\theta) = \frac{TP(\theta)}{P(\theta)} \tag{37}$$

$$FPR(\theta) = \frac{FP(\theta)}{N(\theta)} \tag{38}$$

where TP are the number of matches that have been correctly classified as matches, P are all the true matches in the dataset, FP are the number of non-matches that have been wrongly classified as

matches and N are all the true non-matches in the dataset.
More specifically, the area under the curve (AUC) of the ROC-curve describes how discriminative a model is, where an AUC of 0.5 is a model that randomly guesses and an AUC of 1.0 is a model that perfectly discriminates between matches and non-matches.
The ROC-curve is computed using the validation set, a set never seen by the model during training. It was however constructed from the same images contained in the training dataset, meaning that although the data is new, the environment is not. The ROC-curve should despite give a good first impression of the model's behavior on unknown data.

4.3.4 Deployment

The performance of CNN-based feature matching on the TSC datasets will be evaluated for both different images patches as well as descriptor sizes will be investigated.
The deployment is performed on a workstation equipped with an Intel Xeon Gold 6144 3.6 GHz 8-core CPU as well as a NVIDIA Titan XP GPU.
Because matching features in two images by the highest similarity score is not achievable in real-time applications, matching will instead be performed by brute-force with OpenCV using the L2-norm between two candidates. This matching method is in line with the method used with the remaining classical descriptors in the benchmarking.

4.4 Feature extraction and matching benchmarking

During the feature extraction and description benchmark, different combinations of extractors and descriptors will be tested onto a thermal dataset recorded in a fiery environment.
As features are much more difficult to extract from thermal images due to their lower contrast, the default settings for the different candidates do not suffice. Instead, the different methods will be tuned until a satisfying amount of features is obtained within a respectable computational time. It is therefore more interesting to have metrics in the benchmark that are independent from the obtained number of features, such as rates.

4.4.1 Candidates used in this study

The following feature extractors with their settings will be included in the benchmark:

- **SIFT**: *nFeatures*=2500, *nOctaveLayers*=3, *contrastThreshold*=0.01, *edgeThreshold*=3.0, *sigma*=0.5
- **SURF**: *hessianThreshold*=5.0, *nOctaves*=4, *nOtaveLayers*=3
- **ORB**: *nFeatures*=1500, *scaleFactor*=1.3, *nlevels*, *edgeThreshold*=1, *firstLevel*=0, *WTA_K*=4, *scoreType*=FAST, *fastThreshold*=2.0
- **FAST**: *threshold*=8, *nonMaxSuppression*=True, *detectorType*=9_16
- **GFTT**: *maxCorners*=1500, *qualityLevel*=0.0005, *minDistance*=0.25, *blockSize*=3, *k*=0.001
- **Gradient-based**: *globalOffset*=15.0 (8-bit)/1000.0 (16-bit), *minDistance*=15

Furthermore, the following descriptors and their settings will be compared to match the previously-detected features:

- **SIFT**
- **SURF**: *extendedDescriptor*=True
- **ORB**: *patchSize*=50
- **BRIEF**: *descriptorSize*=64
- **BRISK**: *threshold*=1.0, *octaves*=3, *patternScale*=1.0
- **FREAK**: *orientationNormalized*=True, *scaleNormalized*=True, *patternScale*=15.0
- **CNN-based**: *inputSize*=32/64, *descriptorSize*=128/256

To match features, the L2-norm is utilized for the floating-point descriptors such as SIFT, SURF and CNN-based whereas the Hamming distance is used for the binary descriptors such as ORB, BRIEF,

BRISK and FREAK.

4.4.2 Benchmarking process and criteria

In figure 4.10, the different major steps taken during the feature extractors and descriptors benchmarking are summarized. Only the operations related to the feature detection and matching for the odometry estimation itself are timed. This means that the entropy computation is not included. Furthermore, the image rectification process will be performed beforehand as it is identical for all candidates and is therefore also excluded from the computational time.

Following the benchmarking of feature extractors and descriptors on a thermal dataset by J. Johansson et. al. [13] and T. Mouats et. al. [49], the performance of the different combinations of feature extractors and descriptors will be evaluated based on the amount of keypoints detected, the total amount of features matched in a frame and the recall, i.e. the amount of matches that are considered to be correct. A match is deemed correct if it is considered as an inlier by the RANSAC outlier removal. As the destined application of this project requires real-time performance, the computation speed of the combinations will also be evaluated.
In addition, a uniform distribution of the features is desired for a more accurate estimation of the odometry [63]. Therefore a metric capable of measuring the feature distribution across a frame is also included.
In table 3, a summary of the different criteria and their description is given.

Criterion	**Description**
Mean frames per second	Average frames processed in a second
Mean number of keypoints	Average number of keypoints detected across the frames
Mean matching ratio	Average percentage of matched keypoints in the left image
Mean recall	Percent of matches considered as correct by RANSAC outlier removal
Mean efficiency	Percent correct matches for a given number of detected keypoints
Feature distribution entropy	Metric to determine the distribution of features across the frame

Table 3: Benchmarking criteria used and their description

The benchmark criteria are particularly lead by ratios, as to make the matching evaluation independent from the number of detected keypoints.
The mean number of matches is the average ratio of the number of matches N_m over the number of keypoints N_{kl} detected in the left image:

$$\bar{r}_m = \frac{1}{N} \sum_{i}^{N} \frac{N_{m_i}}{N_{kl_i}} \tag{39}$$

where N is the number of images in the dataset.
Furthermore, the mean recall is the average ratio of correct matches N_{cm} from RANSAC over the total amount of matches N_m in a stereo pair:

$$\bar{r}_{recall} = \frac{1}{N} \sum_{i}^{N} \frac{N_{cm_i}}{N_{m_i}} \tag{40}$$

Finally, the mean efficiency of a feature extractor-descriptor combination is determined by the product of the mean matching ratio and the mean recall:

$$\bar{r}_e = \bar{r}_m \cdot \bar{r}_{recall} \tag{41}$$

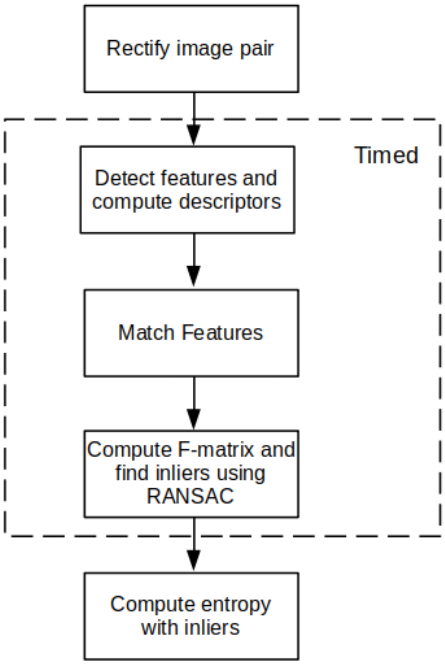


Figure 4.10: Benchmarking steps

4.4.3 RANSAC outlier removal

Verifying every match over the dataset manually would be very time-consuming. The decision has therefore been made to test the validity of a match between two features using Random Sample Consensus (RANSAC). This approach is also consistent with the two previously-mentioned benchmarks by J. Johansson et. al.[13] and T. Mouats et. al.[49].
As explained in section 2.1.1, RANSAC finds inliers by creating hypotheses of the fundamental matrix F (i.e. the epipolar geometry) using a random sample of features. The hypothesis with the most inliers above a specified level of confidence is then selected as the most probable model. A set of matched homogeneous points $\{\tilde{p}, \tilde{p}'\}$ is considered an inlier when the residual, i.e. the reprojection error, of $\tilde{p}'^T F \tilde{p}$ is below a distance threshold.
In order to obtain a fair assessment of the different feature extractors and descriptors, a distance threshold of 3.0 and a confidence level of 0.99 will be used for all participating candidates.

The fundamental matrix F relates a point in one image to the line equation of its corresponding epipolar line in the second image. As this is a linear mapping process, it is important for both images to be free of lens distortion. In this project, the pair of stereo images are rectified before being processed which also removes this distortion.

4.4.4 Feature distribution entropy

One method to assess the quality of a set of points for odometry estimation is by their distribution across the image. Ideally, a uniform distribution of the inliers across the frame is desired as it paints a more detailed picture of the optical flow between two frames.
To evaluate the feature distribution, the entropy of the image is computed. To do so, the image is divided into 320 equally-sized bins. It is assumed that the bins are small enough that it can be considered that the features are uniformly-distributed inside. For each bin, the statistical divergence D in (eq. 42) is computed using Pearson's chi-squared metric [64] in eq. (43):

$$D = f(P, Q) \cdot Q \tag{42}$$

$$f(P,Q) = \left(\frac{P}{Q} - 1\right)^2 \tag{43}$$

where P is the observed distribution in the bin (the number of inliers features inside the bin divided by the total number of inlier features) and Q is the desired uniform distribution (one over the number of bins).
Finally, the entropy H over the image is obtained by taking the sum of the divergences across the bins:

$$H = \sum_{i=0}^{N_{bins}-1} D_i \tag{44}$$

From eq. (43), it can be observed that the more the features are uniformly distributed, the lower the entropy will be. An entropy of $H = 0$ means that $P = Q$ across the bins and thus that the features are perfectly uniformly-distributed.

4.5 Thermal-inertial SLAM

4.5.1 Loop detection

A popular loop closure detection method in feature-based visual SLAM algorithms is the usage of bag-of-visual-words (BoVW) to measure the similarity between frames over an entire dataset.
To assess the feasibility of BoVW loop detection, some experiments will be conducted on the datasets recorded at the Twente Safety Campus. As no ground truth is available, these tests will only remain qualitative.

The experiments will be conducted using the DBoVW 2 library by D. Galvez-Lopez et. al. [65]
To create the hierarchical tree of visual words, features and their descriptors from the 1217 images present in the second TSC dataset. The obtained vocabulary will then be tested on the third TSC dataset. As SLAM algorithms in general do not search for loop closures in all frames but rather only in keyframes, every tenth image in the dataset will considered in the test. Furthermore, only the top candidate from the feature extraction and description benchmark are included in these tests.

After the visual vocabulary for a descriptor is obtained, similarity scores between images in the third TSC dataset are computed. Only closures that are located far apart in time are of interest as, due to the slow velocity of the UGV, close detections may increases the chances of false positives. Therefore, only every tenth image from the dataset is considered. Additionally, a frame is only tested for loop closures with previous frames.
The scaled L2 similarity score s between the bag-of-words feature vectors $\vec{v_1}$, $\vec{v_2}$ of two keyframes is computed in DBoW 2 as follows:

$$s(\vec{v_1}, \vec{v_2}) = 1 - \sqrt{1 - \sum_i \frac{v_{1i}}{|\vec{v_1}|} \cdot \frac{v_{2i}}{|\vec{v_2}|}} \;\forall\; i \in (v_{1i} \neq 0 \wedge v_{2i} \neq 0) \tag{45}$$

where i is the word index in the vocabulary and the score s is a value between 0.0 and 1.0.
The quality of the loop closure detection is measured by the difference in matches features coordinates, using the RMS-error e_{RMS} as followed:

$$e_{RMS} = \sqrt{\frac{1}{N}\sum_i^N (u_{1i} - u_{2i})^2 + (v_{1i} - v_{2i})^2} \tag{46}$$

where N is the number of feature matches between the two images, (u, v) are the image coordinates and the subscripts $1, 2$ designate their respective image in the observed pair.

It can be argued that this assessment metric is not optimal as it is dependent on the robot's perspective of the scene. However, given the lack of ground truth as well as the robot's trajectory in the acquired TSC datasets, the RMS-error will suffice to provide an answer to the possibility of BoVW-based loop closure detection with thermal images.

4.5.2 Trajectory estimation and 3D reconstruction

To estimate the trajectory of the robot as well as compute a 3D reconstruction of the environment, a state-of-the-art feature-based visual-inertial SLAM framework will be used to handle the classical SLAM operations itself. These include computing the odometry, creating the 3D map, merging thermal odometry estimation with IMU data, bundle adjustment and the loop closure detections. These operations all function independent of the type of images that is fed through the system, and therefore, given the image feature locations and camera intrinsic and extrinsic parameters, have the possibility to perform identically using both thermal and visual images.

In this project, the recently-released ORB SLAM 3 algorithm by C. Campos et. al. [14] was chosen as the SLAM framework for thermal-inertial SLAM. ORB SLAM 3 is a camera-independent visual(-inertial) SLAM framework, meaning it supports monocular vision, stereo vision and RGB-D cameras with both pinhole and fisheye lens models. It furthermore does not necessarily require rectified images. Compared to ORB SLAM 2, ORB SLAM 3 now includes IMU sensor-fusion support [14].
Compared to other state-of-the-art visual-inertial SLAM algorithms, ORB SLAM3 offers greater robustness against loss of localization. The framework is not only able to utilize the information acquired within a few seconds, but also the information gained since the start of the acquisition using a multi-map "atlas" system. This method makes it more robust when the loss of tracked features occurs compared to previous ORB SLAM iterations and other SLAM algorithms. In case not enough features can be tracked, a new map is created which can later be incorporated again into a previously-obtained map inside the atlas when an overlap is detected. This ability of long-term data association together with its full reliability on Maximum-a-Posterior (MAP) estimation ensured it was able to be between two to five times more accurate than other state-of-the-art SLAM approaches on popular benchmarking datasets such as EuRoC and TUM-VI [14].

In figure 4.11, the proposed architecture of this project's thermal-inertial SLAM architecture is displayed. As can be seen, the different components investigated during this project, namely the thermal image features as well as the bag-of-visual-words, will serve, along with the raw accelerometer and gyroscope data, as the inputs for the ORB SLAM 3 SLAM estimation and optimization.

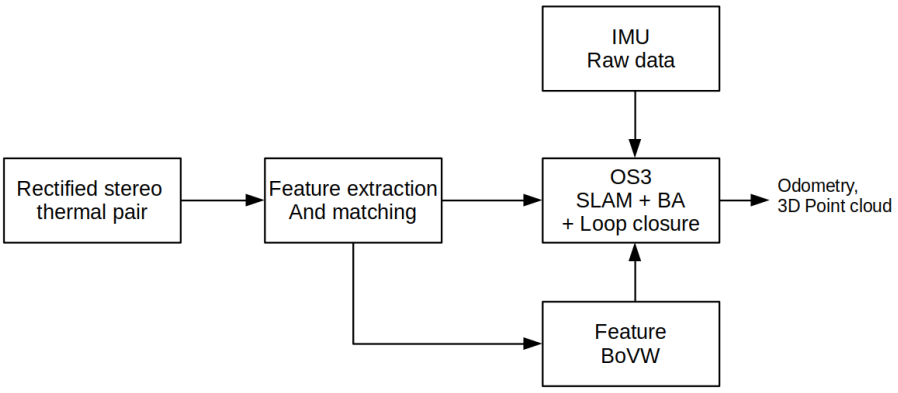


Figure 4.11: Firebot thermal-inertial SLAM architecture

The best combination of feature extractor and descriptor based on the results obtained from the benchmark performed during this project will be implemented inside the ORB SLAM 3 framework.

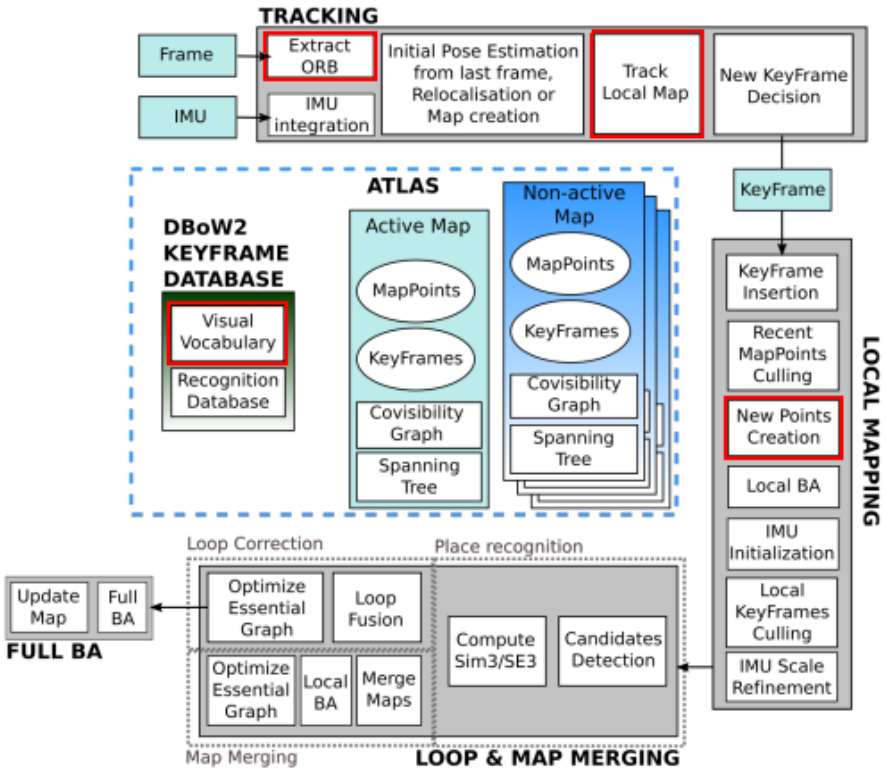


Figure 4.12: ORB SLAM 3 architecture diagram [14]. The elements that have been modified to accommodate SURF-BRIEF are highlighted by red bounding boxes

In figure 4.12, the ORB SLAM 3 architecture diagram is depicted, where the elements modified during this project to integrate SURF-BRIEF have been highlighted by red bounding boxes.
The ORB feature extractor has been replaced by the OpenCV implementation of the SURF extractor. Although both ORB and SURF provide scale invariance, in contrast to ORB which achieves it by extracting FAST corner features at different image scales, SURF provides scale invariance by modifying the filter size used to compute the second order derivatives. As a result, the algorithm does not need to deal with the additional uncertainty associated with the down-sampling of the image. On the other hand, limiting the feature matching between features of the same octave, i.e. for SURF with the same filter size, to reduce the changes of outliers remain.
ORB SLAM 3 utilizes the DBoW2 bag-of-visual-words library to compute a hierarchical vocabulary tree of image "words" from the features descriptors. Apart from loop closure detection, the bag-of-words is also used as the primary feature matching method when tracking a local map. During map tracking, feature matching is namely limited between features which belong to the same node at a prescribed tree depth. Both ORB and BRIEF are binary descriptors and the Hamming distance as a matching metric is kept. Additionally, as the ORB descriptor is derived from BRIEF, the vocabulary tree shape of six layers of depth and ten branches per layer adopted by ORB SLAM 3 will be taken as a starting point. As with ORB, only SURF features then belonging to the same node at the fourth layer as well as to the same octave (or directly neighboring octave) will be matched together.
Loop closure detection in ORB SLAM 3 is performed on a separate thread at regular intervals of 5.0 seconds. In this project's approach however, the interval has been reduced to 1.25 seconds for better results.

As ORB features and description belongs to the top candidates in the benchmarking, the odometry accuracy of SURF-BRIEF thermal-inertial SLAM algorithm will be compared against the stock ORB SLAM 3 algorithm. As recording a ground truth odometry was not feasible at the Twente Safety Campus, a separate dataset is recorded inside Saxion University using an Optitrack motion capture system.
Gradient-based feature tracking, despite its promising results, was omitted as an implementation can-

didate in ORB SLAM 3. This because gradient-based features have no repeatability, meaning that when a certain scene is perceived from a different perspective, features are not detected at the same location on the 3D objects. This makes this method unsuitable for feature matching by descriptors as is heavily done, notably for loop closure detection, in ORB SLAM 3 and all other state-of-the-art feature-based visual SLAM algorithms. In case gradient-based features were to be implemented, additional research into alternative loop closure detection method would have been required. Alternatively, ORB features and descriptors would be used for loop closure detection along with tracking of gradient-based features for odometry estimation. This would however, due to the aforementioned problem with gradient features, lead to the need of two 3D point triangulation algorithm operating in parallel. This in turn would result to an unnecessary enlarged complexity which would deter real-time performance of the system.

The odometry estimation and point-cloud computations will not be performed on-board on the Intel NUC i7, but rather off-board using a PC equipped with an Intel i7 processor. At a larger stage, the different software components will be adapted to operate in real-time on the on-board Intel NUC i7.

5 Results

In this section, the results from the different experiments performed during this thesis will be displayed. First, the tuning of the dynamic 16-bit to 8-bit conversion is shown followed by the calibration of the thermal cameras, the IMU and the thermal-inertial system. Then, the results from the preliminary experiments, i.e. from the gradient-based feature tracking and CNN-based descriptors are demonstrated. Next the results from the feature extraction and description benchmark are presented. Finally, the odometry and mapping results in a smoke-filled environment obtained using the top candidate from the benchmark will be displayed.

5.1 Linear Automatic Gain Control

Looking at the normalized histograms across the hottest run in the datasets in figure 5.1, it can be seen that the majority of the 16-bit information is steadily located between 20500 and 33000, with the frequency peaking at around 21000. This range can be attributed to the environment, with the peak corresponding primarily to the floor.

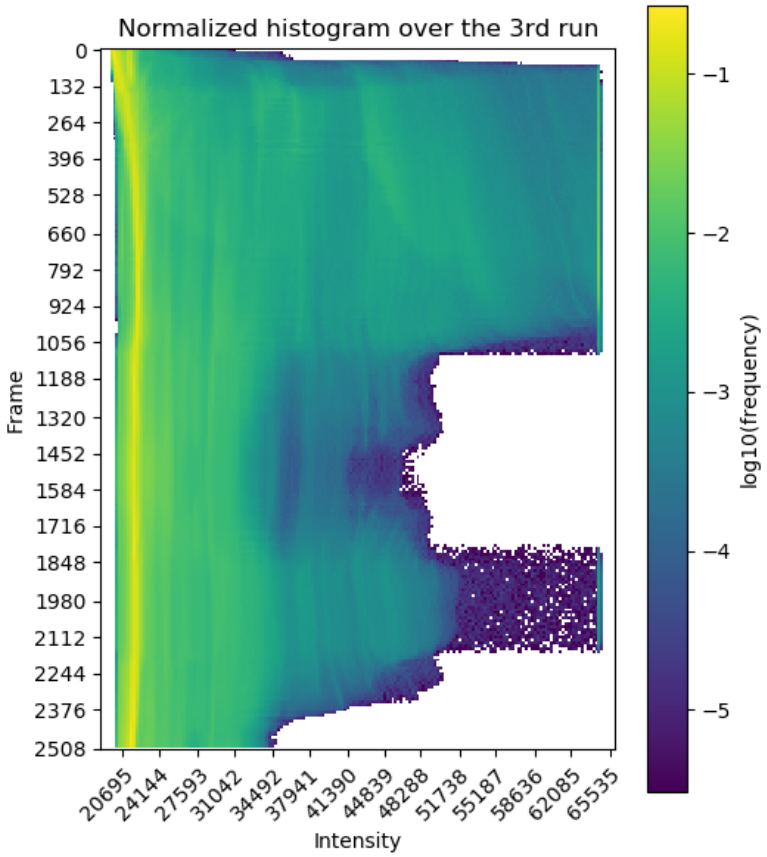


Figure 5.1: Normalized histogram across the third run

Close to 40000, a local minimum in frequency can be observed across the dataset. This decline represents the transition between radiometric data from the environment and the very hot objects such as the fire source. In the first third of this third run, the robot directly face both fire sources, namely inside the car and a barrel to the left of it. These fire sources can be found in figure 5.2, where the areas above 40000 are masked in red. Additionally, the ceiling right above the fire is also included in the mask and actually represents its majority.

Figure 5.2: Mask showing dataset areas above 40000 (red)

From the results presented above, it can be concluded that 40000 is an appropriate value for the threshold I_{max} as it represents the transition between the environment and high-temperature objects which increase the risk of darkening or saturating the useful information.
The lowest percentage in the cumulative histogram p_l is set to 0.0%. Furthermore, the upper percentage p_u to 99.0%, meaning that the first 99.0% of intensity values in the dynamic range are converted to 8bit. This prevents the very small upper part of the dynamic range, generally above 32000, from unnecessarily increasing the range and thus reducing the level of detail in the most prominent parts.

5.2 Hardware calibration

In this section, the results from the thermal camera calibrations as well as the IMU calibration and the calibration between the two will be displayed.

5.2.1 Camera calibration and image rectification

Table 4 shows the different settings used in the ROS *image_pipeline* camera calibration module. For all calibrations, no calibration flags specifying additional assumptions such as no tangential distortion or fixing the principal in the image center were adopted. The *image_pipeline* calibration tool selects the images used in the estimation not only based on whether the circle grid has been detection in the image, but also based on the skewness of the detected board as well as its relative position from the previous frame to prevent motion blur. The reprojection error was computed separately on 1234 undistorted images using eq. (16) based solely on whether the acircular grid was detected in the stereo pair.
During the first calibration, settings 1, the default number of two k-coefficients was used in the estimation of the lens distortion. At first glance, the calibration results look acceptable as the RMS reprojection error is well below 1.0. However, looking at the rectified stereo pair in figure 5.3 obtained using this camera calibration, it can be observed that some residual distortion remains, especially in the right image. Physically-straight lines in 3D must appear straight in the rectified image, which is not the case here in the seems of the wall and the ceiling in the right image.

Calibration	Settings 1	Settings 2	Settings 3	Settings 4
Type	stereo	stereo	mono	stereo
Pattern	acircular	acircular	acircular	acircular
Size	11×4	11×4	11×4	11×4
Square	0.043 m	0.043 m	0.043 m	0.043 m
k-coefficients	2	3	3	4
Images	100	100	120 (left), 122 (right)	100
Reprojection error	0.0311	0.00619	0.00707	0.0244

Table 4: Settings and reprojection errors for the three camera calibrations

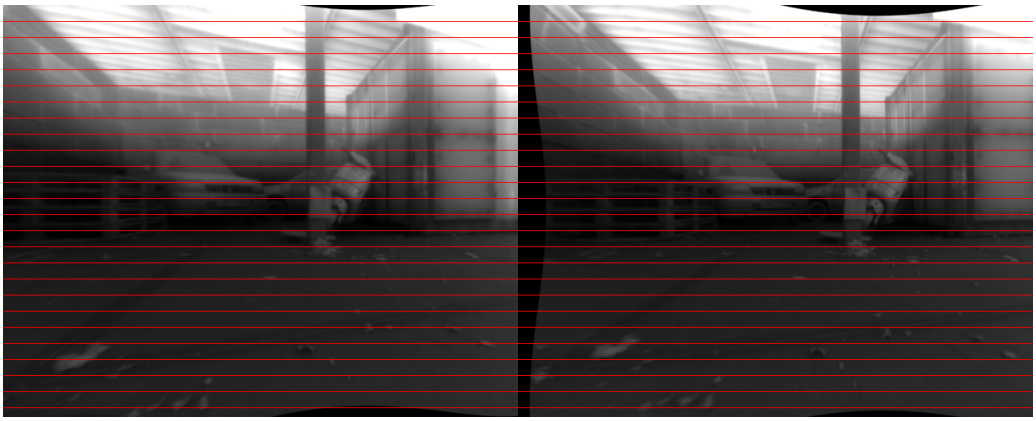


Figure 5.3: Rectified stereo pair with two k-coefficients and free scaling parameter $\alpha = 0.2$, obtained from the calibration with settings 1

Increasing the number of coefficients to estimate the image distortion by one, as done in the second calibration, settings 2, yields a RMS reprojection error that is five times smaller than the calibration with settings 1. As can be seen in figure 5.4, less image distortion is present in the rectified stereo pair. In both figures 5.3 and 5.4, the stereo pairs are best rectified at the bottom of the image, with the error increasing with the height of the image. With three k-coefficients as in the calibration with settings 2, however, the increased accuracy in the distortion yields a smaller rectification error, seen by the smaller vertical distance between a horizontal line in the right image and the corresponding pixel in the right image of a pixel on the line in the left image.

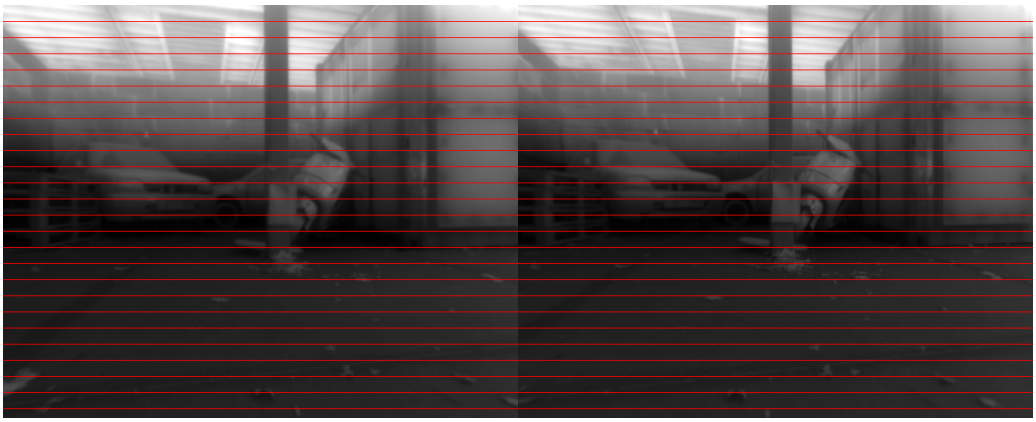


Figure 5.4: Rectified stereo pair with three k-coefficients and free scaling parameter $\alpha = 0.2$, obtained from the calibration with settings 2

When performing a stereo camera calibration, images are only used if the calibration board is detected in both the left and right image. As a result, images where the board is located on the edges, i.e. where the distortion is the strongest, are often discarded. Therefore, two monocular calibrations where performed separately in the third calibration, settings 3, in order to better estimate the cameras' intrinsics, in particular the edge distortions. For the extrinsics, the results from the previous stereo calibration, settings 2, were used. Despite the increased number of used images, the RMS reprojection error did not improve during this method, and was slightly higher than the stereo calibration.
Additionally, further increasing the number of k-coefficients, as in the calibration with settings 4, did not yield better calibration results, as the RMS-error with four k-coefficients is double the RMS-error with three k-coefficients.
In eqs. (47)-(49), the calibration results from the second run, settings 2, are shown. The estimated camera intrinsics only slightly deviate from the expected values in eq. (15), suggesting a correct calibration. Furthermore, the rotation between the two cameras is very close to identity and the baseline is computed to be 81 mm, only a millimeter difference from the measurement given on the technical drawing of the camera mount. Considering that the y-axis defines the vertical axis in the world camera coordinates and the z-axis the depth axis, no height difference between the two cameras has been estimated, as well as only a 4 millimeter depth difference.
Using eq. (19), the obtained Δy_{RMS} using this calibration is equal to 1.09 pixel. This means that, on average, a point in the left image is located on the row above or below in the right image.

$$K_l = \begin{bmatrix} 408 & 0 & 326 \\ 0 & 410 & 258 \\ 0 & 0 & 1 \end{bmatrix} \ pixels \tag{47}$$

$$K_r = \begin{bmatrix} 408 & 0 & 315 \\ 0 & 410 & 253 \\ 0 & 0 & 1 \end{bmatrix} \ pixels \tag{48}$$

$$R = \begin{bmatrix} 0.9999 & 0.0052 & 0.0102 \\ -0.0053 & 0.9999 & 0.0111 \\ -0.0101 & -0.0112 & 0.9999 \end{bmatrix} \tag{49}$$

$$t = \begin{bmatrix} -8.122 \cdot 10^{-2} \\ 3.094 \cdot 10^{-6} \\ 3.895 \cdot 10^{-3} \end{bmatrix} \ m \tag{50}$$

It should be noted that, along with the estimation errors from the calibration, inaccuracies in the camera production as well as in the fabrication of the camera mount using the 3D-printer also contribute to the uncertainty of the results.

5.2.2 IMU calibration

Unsatisfying results were obtained while trying to calibrate the XSens Mti-620 with the thermal cameras as a thermal-inertial system. Notably high drift and noise were observed in the data of the z-axis accelerometer. It was therefore suggested that the sensor of the axis in question may be defective. To verify this hypothesis, the Allan curves of the z-axis accelerometer measured from the Mti-620 and the one provided by XSens for the Mti-600 series were compared, as shown in figure 5.5. It can be observed that the Allan curves are different. The descent of the measured curve occurs at a later time cluster size then the provided curve, meaning that the velocity random walk is higher. Additionally, while the rebound was not measured in the provided Allan curve, it is clearly visible in the measured Allan curve from the Mti-620. This means that the rate random walk of the sensor is also much higher than is specified by the manufacturer. The Mti-620 used in this project is thus not conform to the XSens' specifications and is therefore indeed defective.

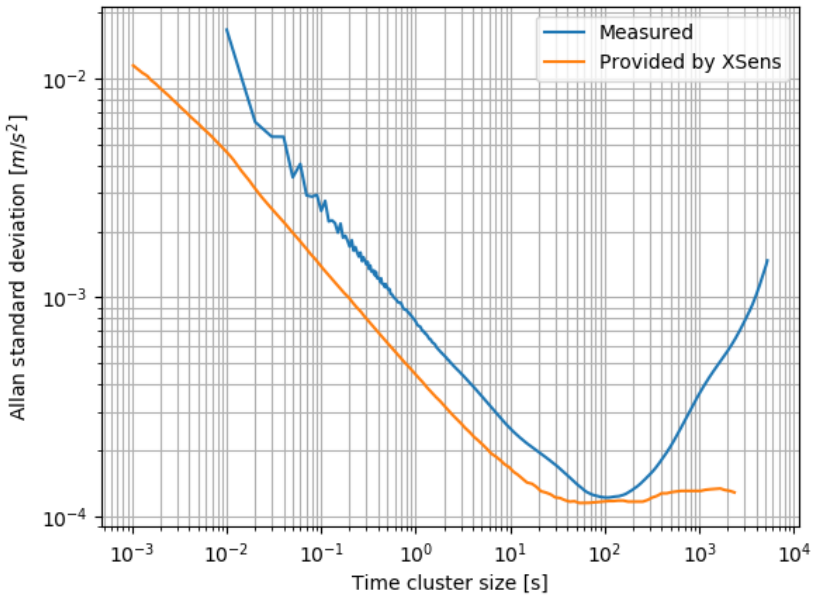


Figure 5.5: Measured Allan curve of the z-axis accelerometer from the Mti-620 along with the provided one from XSens for the Mti-600 series

Although the datasets at the Twente Safety Campus were already recorded before this diagnosis, a spare XSens Mti-630 was calibrated and installed in the setup in order to be able to perform the accuracy measurement of the thermal-inertial SLAM system inside the Optitrack room.
Table 5 shows the average IMU bias, scale factor and misalignment error parameters over three calibrations of the spare Mti-630. According to the manufacturer, the scale factor and misalignment should not be higher than of the order 10^{-3} for both the accelerometer and the gyroscope. As seen in the table, this is the case for the accelerometer. The scale factor and misalignment of the gyroscope are on the other hand very high. The scale factor and misalignment of the gyroscope are computed by using the Earth's rotation as reference, which is of the order 10^{-5}. It can therefore be deduced that the white noise on the gyroscope sensors is more prevalent, and thus the resolution of the sensors is not sufficient to use the Earth's rotation to determine the scale factor and misalignment.
As the manufacturer specifies that the scale factor and misalignment are identical for the accelerometer and gyroscope, the scale factor and misalignment from the accelerometer will be used to compensate the errors in the gyroscope.

	Accelerometer	**Gyroscope**
Bias	$[-0.005569 \quad -0.003369 \quad -0.001814]^T$	$[0.001523 \; 0.0003445 \; 0.002034]^T$
Scale factor	$\begin{bmatrix} -2.296 \cdot 10^{-3} & 0.0 & 0.0 \\ 0.0 & 1.847 \cdot 10^{-5} & 0.0 \\ 0.0 & 0.0 & -5.745 \cdot 10^{-4} \end{bmatrix}$	$\begin{bmatrix} 3.477 & 0.0 & 0.0 \\ 0.0 & 1.91 & 0.0 \\ 0.0 & 0.0 & 18.88 \end{bmatrix}$
Misalignment	$\begin{bmatrix} 0.0 & 6.801 \cdot 10^{-3} & 8.348 \cdot 10^{-3} \\ -2.322 \cdot 10^{-3} & 0.0 & -6.664 \cdot 10^{-3} \\ 1.653 \cdot 10^{-2} & 2.902 \cdot 10^{-3} & 0.0 \end{bmatrix}$	$\begin{bmatrix} 0.0 & -5.010 & 0.7373 \\ -4.102 & 0.0 & -5.902 \\ 4.470 & -8.178 & 0.0 \end{bmatrix}$

Table 5: IMU error parameters of the XSens Mti-630

5.2.3 IMU-camera calibration

Figure 5.6 shows the angular rates of the system measured by the camera and the IMU. Visually inspecting the data, the rotation matrix ${}^{c}R_{b}$ should closely resemble the following expected rotation ${}^{c}R_{b_e}$:

$$ {}^{c}R_{b_e} = \begin{bmatrix} 0 & 1 & 0 \\ -1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{51} $$

It can be noticed that the larger oscillations present around 40 and 65 seconds in the y-axis from the IMU are present in the corresponding x-axis from the camera (roll axis) but are drowned by significant noise.
Performing the time synchronization, it was observed that the IMU measurements lag 63 milliseconds behind the camera angular rates.
Having shifted the IMU measurements in time to match the camera angular rates, the following rotation matrix was obtained by solving the Procrustes problem:

$$ {}^{c}R_{b} = \begin{bmatrix} -0.1080 & 0.9858 & 0.1284 \\ -0.9941 & -0.1054 & -0.02631 \\ -0.01241 & -0.1305 & 0.9914 \end{bmatrix} \tag{52} $$

The solution found in eq. (52) has a determinant of 1.0000 making it a valid rotation matrix.
Figure 5.7 shows the rotated IMU angular rates after the calibrations. Comparing eq. (51) and eq. (52), the obtained rotation closely resembles the expected one, confirming the correctness of the Procrustes solution.
The angular rates measured by the left camera and the IMU are, as expected, very similar in amplitude, especially when the system is primarily rotating around a given axis. The noise present on the camera's x-axis did not appear to influence the solution. The shape of the two corresponding axes are matching, with the camera's x-axis being higher in amplitude.

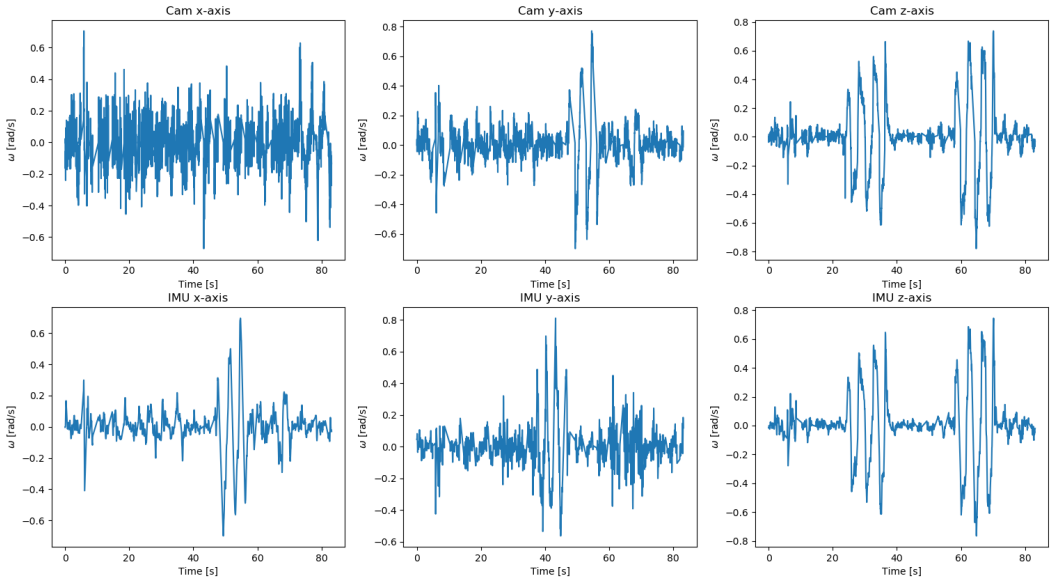


Figure 5.6: System angular rates measured by the camera (top) and the IMU (bottom) before the calibration. The axes are named according to their respective sensor's body-fixed reference frame.

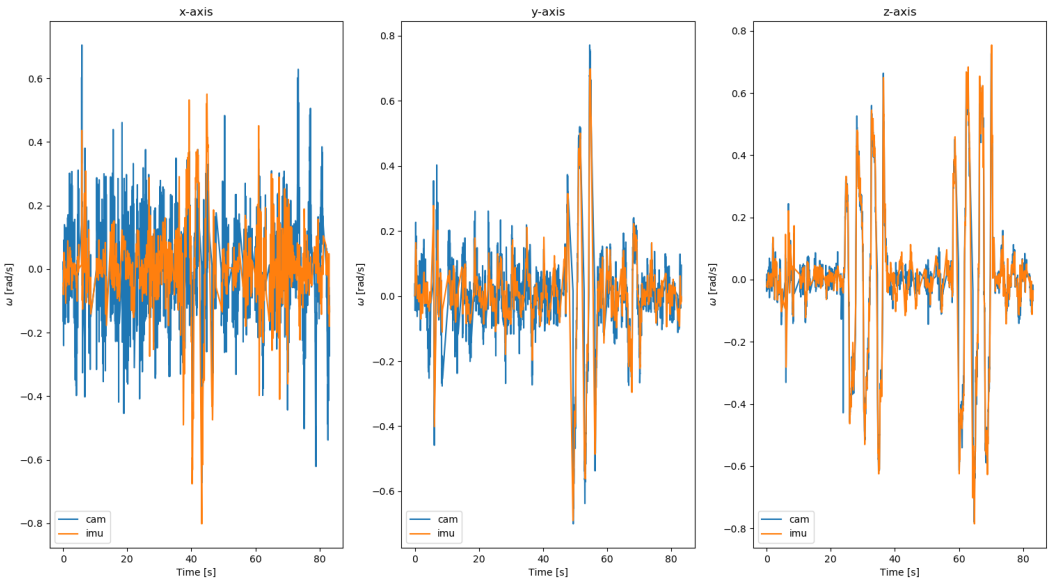


Figure 5.7: Rotated IMU angular rates after the calibration

Including the translation obtained from the 3D CAD model, the transformation matrix ${}^{c}H_{b}$ is equal to:

$$ {}^{c}H_{b} = [{}^{c}R_{b}|{}^{c}t_{b}] = \begin{bmatrix} -0.1080 & 0.9858 & 0.1284 & 0.0477 \\ -0.9941 & -0.1054 & -0.02631 & -0.0068 \\ -0.01241 & -0.1305 & 0.9914 & -0.01524 \end{bmatrix} \tag{53} $$

where ${}^{c}t_{b}$ is the position of the body-fixed IMU origin in the coordinate system of the left thermal camera.

5.3 Preliminary experiments

Before the benchmarking is performed, the performance from the two feature extraction and matching methods developed implemented with further depth during this project are examined.
In a first part, gradient-based feature matching is examined in more detail, investigating the benefits of utilizing the 16-bit radiometric data when tracking the features in a stereo image pair. Secondly, the training results from the CNN-based descriptors are showcased.

5.3.1 Gradient-based features

The main difference between applying gradient-based feature extraction and tracking on 8-bit thermal images or the 16-bit radiometric data is the tracking accuracy. The number of obtained keypoints is namely largely dependent on the prescribed global offset. Theoretically, under the right conditions, the maximum number of obtained keypoints in both 8-bit and 16-bit is therefore equal to the number of pixels in the image. The resolution of the gradient of the radiometric data is however much higher than the gradient from the 8-bit image. This results in the gradient images having more texture, and thus reduces the ambiguity.
In table 6, the results from using 8-bit or radiometric data for tracking are shown. Here, in order to insure that only the tracking quality itself is evaluated, both tracking methods use keypoints obtained from the 8-bit image as the keypoints in a different datatype lie at different locations in the image. As can be noticed, the rate of successfully-tracked points, i.e. whose NCC-value is above a threshold of 0.7, is significantly higher when tracking using the 16-bit raw data. The recall however, i.e. the

number of successfully tracked point pairs accepted by the RANSAC outlier removal, is not affected by the increase in gradient resolution. It can therefore be concluded that lower cross-correlation values, and thus a lower tracking certainty is obtained when opting for the 8-bit thermal images. This effect can be explained by the loss of information when converting the 16-bit radiometric data into 8-bit values. By reducing the intensity resolution, the number of zero gradients increases leading to lower NCC-values when the tracked points lie inside an area of poor gradient in the query image. This explanation is supported by the higher entropy obtained when tracking the features in 8-bit, meaning that the correctly-tracked points are less uniformly-distributed across the image. As gradient-based features are selected with a scattered distribution in mind, this would suggest that certain parts of the 8-bit image contain features that are harder to track with a high NCC and are thus discarded.

Tracking	**Keypoints**	**Tracked rate**	**Recall**	**Correct points**	**Efficiency**	**Entropy**
8-bit	946	0.809	0.846	647	0.684	1466
16-bit	946	0.932	0.849	797	0.792	1071

Table 6: 8-bit and 16-bit tracking results using features obtained on an 8-bit image

Among the correctly-tracked points accepted by RANSAC, some small tracking errors are however still visible, as can be seen in figure 5.8. Such errors can be limited by decreasing the width of the search area. The tracking errors are much more likely to occur when the tracked points are located onto objects situated close to the camera in the 3D scene. In such a case, the matching template and searching area are more likely to be less diversified in texture as it will often be taken from a single surface. On the other hand, objects close to the camera also have a greater disparity in the stereo image pair, meaning that, if the search area is too small, its true position in the tracking image might be out of bound. Along with the size of the search area, the size of the matching template is also of importance. A small matching template may not provide enough detail whereas a larger template may encapsulate too much information, blurring the details that constitute its singularity with respect to its location. The latter is caused by the fact that NCC only indicates the global similarity between two patches, and cannot perform a pattern comparison within the patches provided by more complex methods such as a siamese CNN image similarity network.
From practical experience, it was observed that a search area of size 15×15 pixels along with a matching template of size 11×11 pixels provided the highest efficiency, i.e. the highest tracking rate and recall.

Figure 5.8: Gradient feature tracking errors among accepted points by RANSAC (16-bit)

Finally, in table 7, the results of 16-bit gradient-based feature extraction and tracking can be seen across the three recorded datasets at the Twente Safety Campus. For all datasets, the same param-

eters were used, meaning that no modifications were performed in between tests to obtain better results in a given situation. It can be noticed that the number of selected features increases as the room temperature increases between the first and second acquisition. On the other hand, the rate of successfully-tracked points slightly diminishes. This suggests that not all of the additional points in the second and third dataset are of higher quality, i.e. their location does not provide a unique environment for a matching template. The steady recall across the three datasets, and not increased recall, also supports this hypothesis.
Looking at the entropy over the acquisitions, it can be deduced that the increase in temperature between the first and second drive leads to more texture resulting in more features across the entire image. During the third acquisition however, although the same number of points are obtained with respect to the second, the entropy of the correctly-matched features is much higher. Due to further increasing of the temperature, a larger part of the ceiling is now masked-out from feature extraction as its 16-bit intensity exceeds the threshold between environment and fire set at $4 \cdot 10^4$. It should however be noted that the entropy of 797 obtained in the third dataset can still be considered as a relatively uniform distribution across the images.

Dataset	**Keypoints**	**Tracked rate**	**Recall**	**Correct points**	**Efficiency**	**Entropy**
TSC 1	950	0.945	0.859	771	0.812	579
TSC 2	1085	0.924	0.851	854	0.787	395
TSC 3	1089	0.926	0.852	859	0.782	797

Table 7: 16-bit gradient feature extraction and tracking over the three datasets with increasing room temperature

5.3.2 CNN feature matching

Training of the different feature matching architectures was performed over 200 epochs, i.e. 200 iterations over the entire training dataset. The four architectures are created from different combinations of input image patch sizes (32×32 and 64×64) and output descriptor sizes (128 and 256). In table 8, the different parameters used to tune the Adam optimizer are displayed. All parameters are equal for the training of the different architectures with the exception of the initial learning rate l_r. For an input image patch size of 32×32, the initial learning rate is equal to 0.1. For the larger input patch size of 64×64, the initial learning rate is increased to 0.5 for an output size of 128 and 1.2 for a descriptor size of 256.

Parameter	**Value**
l_r	0.1/0.5/1.2
ϵ	0.5
β_1	0.9
β_2	0.999
L2-regularization	0.001
batch size	128
epochs	200

Table 8: Parameters settings for the Adam optimizer

In table 9, the area-under-the-curve (AUC) and accuracy for each classification architecture on the validation dataset is shown. It can be observed that the four combinations all have a high selectivity

in differentiating matching image patches from non-matching ones as the AUC is close to 1.0. The ROC-curve on the validation set of the model with an input image patch size of 64×64 and a descriptor size of 256 is shown in figure 5.10. Furthermore, the increased descriptor size does not seem to impact the performance of the classifier as both the AUC and accuracy are very similar given the same input image patch size. On the other hand, increasing the size of the input image patch slightly increases the accuracy of the classifier.

Architecture	**AUC**	**Accuracy**
(32, 128)	0.99726	0.98734
(32, 256)	0.99506	0.98681
(64, 128)	0.99683	0.99139
(64, 256)	0.99717	0.99121

Table 9: Classification results from the validation dataset

In figure 5.9, both the value of the loss function and the accuracy during training for the model with an input image patch size of 64×64 and a descriptor of size 256 are shown. The ADAM optimizer rapidly convergences to a solution as the accuracy increases close to 1.0. As expected, the accuracy on the unknown validation dataset is slightly lower than the accuracy on the training dataset. The loss function does not diminish in a totally smooth manner, which is typical for the ADAM optimizer. Theses jitters in the training loss are amplified in the validation loss. This is especially visible when observing the accuracy in figure 5.9. The training behavior and ROC-curve of the remaining three models are very similar to the current one, as expected from the results in table 9.

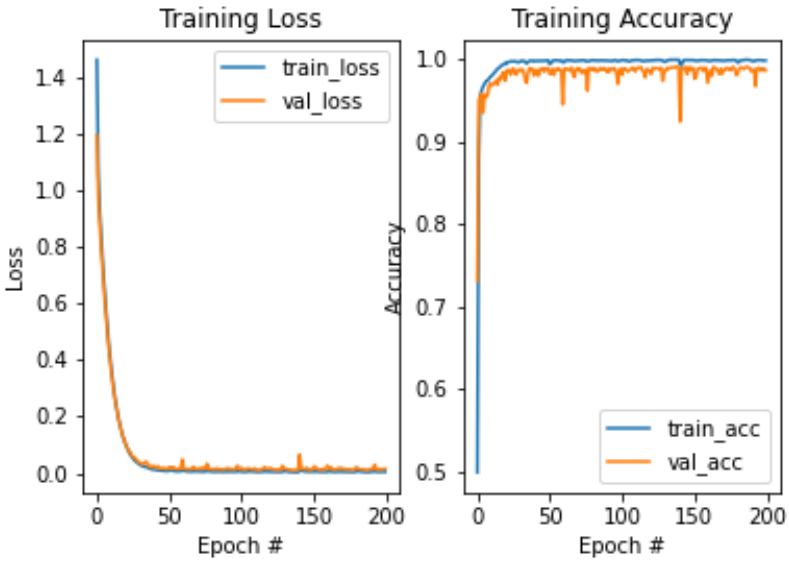


Figure 5.9: Training history of a ROI of size 64 and a descriptor of size 256

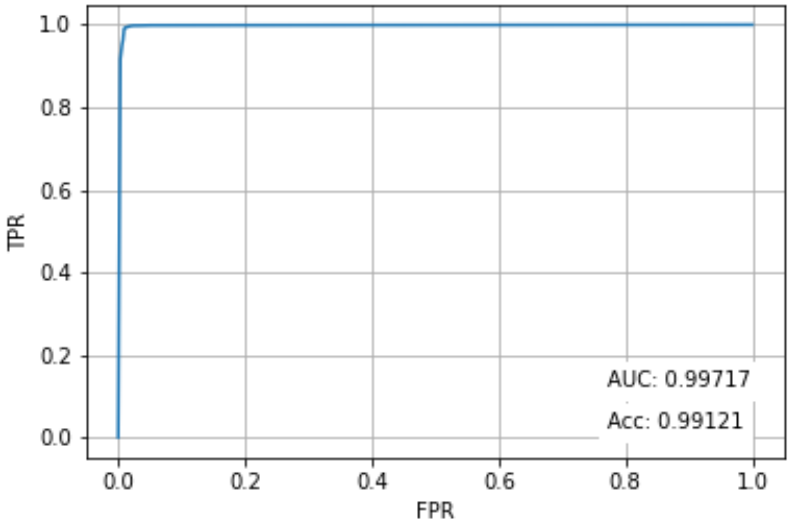


Figure 5.10: ROC-curve for the model with a ROI of size 64 and a descriptor of size 256

Evaluating the four CNN-based matching models on the third and warmest dataset recorded at the TSC yields the rate metrics graph in figure 5.11. As can be observed for both blob features from SURF and corner features from ORB, increasing the input image patch size and the output descriptor size increase the matching recall. The former does however have a greater impact on the recall compared to the latter. Furthermore, the recall is higher when matching ORB features compared to SURF features. The difference may be explained by the location of the features. Blob features are centers of image patches with a similar gray-scale intensity, i.e. of similar texture, whereas corner features are located in image regions of high gradient, and thus of high texture. The latter therefore ensures that the features' surroundings provide richer information to the input of the CNN. The training data contains a combination of visually-selected features and ORB features, both being corners. Including blob features in the training data may push the optimizer to find more intricate details in the image patches which may not have been necessary with the current richer image patches.
The higher matching rate obtained with SURF features for all four models suggests that, although blob features are located in a low-gradient region, SURF features enable the CNN to create more distinct descriptors compared to ORB features. For a match to be accepted when performing cross-check, the features in a pair have to be the mutual top candidates when comparing it with other features in the corresponding image. The higher matching rate then implies that there is less ambiguity between descriptors from SURF features.
Looking at the efficiency, i.e. the product of the matching rate and the recall which describes the percentage of correct matches from a given number of detected keypoints, CNN-based matching performs superior when combined with SURF features, despite it being trained using ORB features and visually-located corners. Using SURF features, increasing the descriptor size is only beneficial if the amount of useful input information, the image patch size, is increased accordingly.

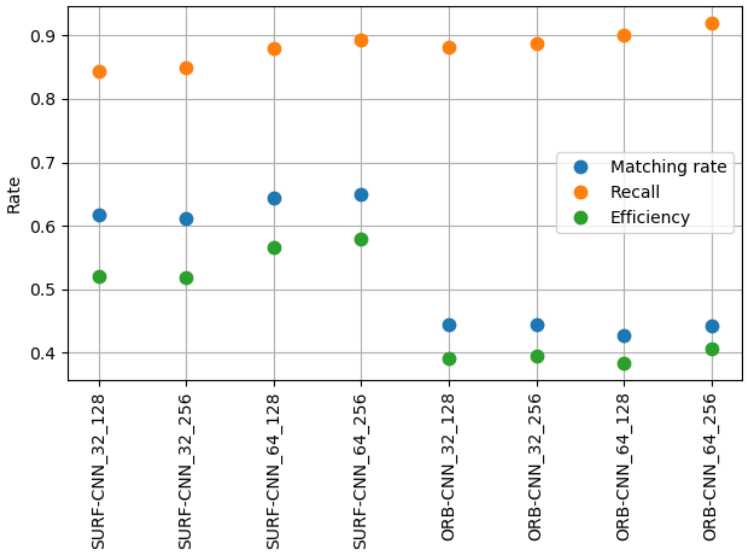


Figure 5.11: Rate metrics of CNN matching on TSC3

The change in performance when operating in colder environments is examined in figure 5.12. In figure 5.12a, the decrease in rate metrics between the hottest (TSC 3) and the coldest (TSC 1) recorded dataset is shown. It can be observed that, overall, the larger image patch size results in a smaller decrease of the different rates. For ORB features, the matching rate is more affected by the lower temperatures, and thus lower texture, than the recall. ORB features seem to be more robust in these changing conditions compared to SURF features. The reason behind it is the location of the different feature types, as explained previously for the lower recall with SURF features. Despite the higher decrease in rates with SURF features compared to ORB features, the initial efficiency of SURF features in TSC 3 was remarkably higher meaning that SURF still out-performs ORB in TSC 1 and TSC 2. The difference between the second and third TSC dataset is less prominent, as can be seen in 5.12b where the decrease does not exceed 3%.

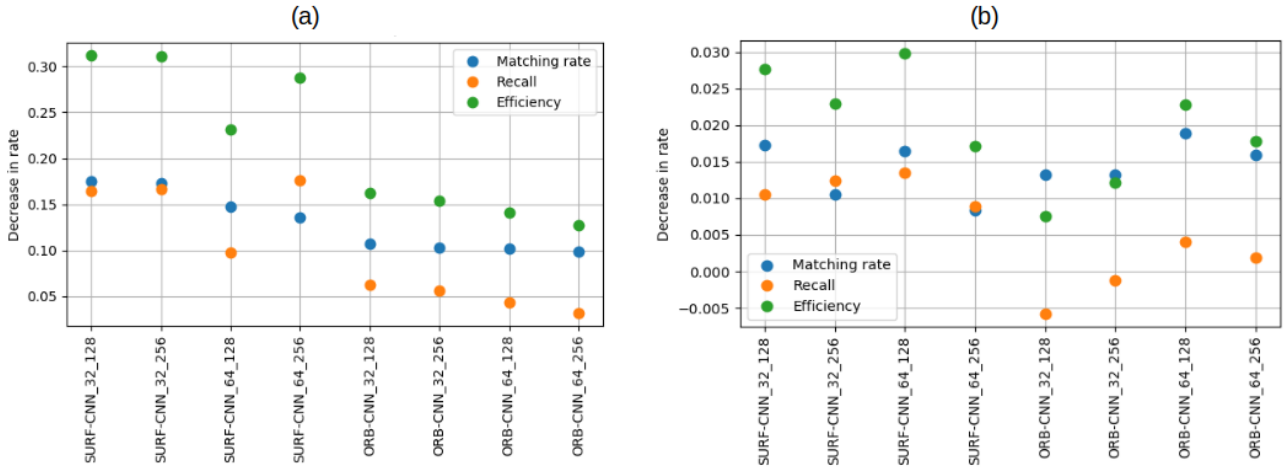


Figure 5.12: Decrease in CNN matching rate metrics: TSC1/TSC3 (a) and TSC2/TSC3 (b)

Finally, from figure 5.13, it becomes apparent that CNN-based matching is not suitable for real-time applications at this stage. Increasing the size of the descriptor does not significantly increase the computational time, whereas increasing the input image patch size greatly impacts the real-time performance of the algorithm.
The main obstacle to real-time operation stems from the pre- and post-processing in Python of the image patches and descriptors as only about 90 milliseconds is required to compute 3200 descriptors

in a single batch using Keras.

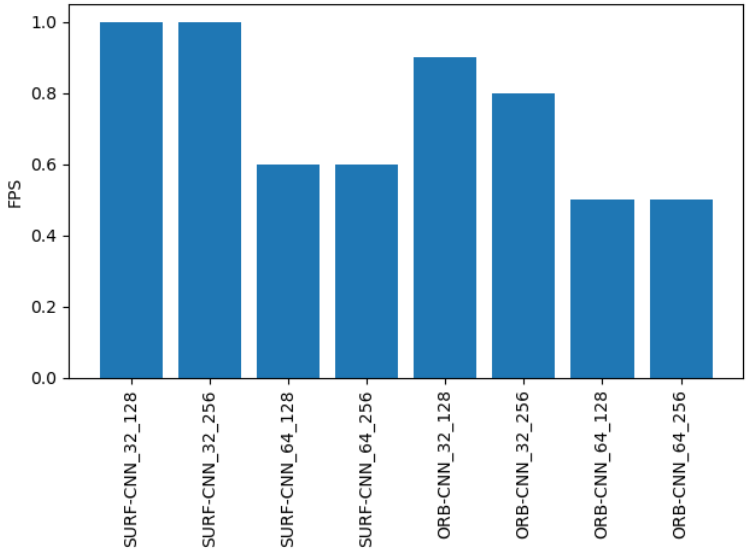


Figure 5.13: Average frames per second in CNN matching

CNN-based matching shows promising results and can definitely be considered as one of the candidates in the benchmark. The largest input image patch size in combination with the largest descriptor size offers the best performance for both SURF and ORB features and will therefore be the only one to be compared against the classical descriptors.
If CNN-matching shows high performance in the benchmark and proves to be a serious choice to use for the estimation of odometry, further investigation to enhance its real-time operability such as a C++/Python interface only the computation of the descriptors itself is performed using Keras in Python and the pre- and post-processing remains in C++.

5.4 Feature extraction and matching benchmark

In tables 10-12, the benchmark results for all the candidates across the three datasets respectively are displayed. The dataset results are shown from the coldest, TSC 1, to the hottest TSC 3. The overall top performers are highlighted in bold. To keep the tables clear, not all the results from all possible combinations of feature extractors and descriptors are included. For example, if a certain extractor provides a low amount of features, it will only be displayed with its best-performing descriptor.

Combination	FPS	Keypoints	Match rate	Total matches	Recall	Correct matches	Efficiency	Entropy
SIFT	8.4	580	0.533	309	0.670	207	0.357	7723
SURF	4.6	1241	0.479	595	0.692	412	0.332	3316
SURF-BRIEF	**7.3**	**1241**	**0.510**	**633**	**0.837**	**530**	**0.427**	**3129**
SURF-FREAK	8.6	1241	0.393	488	0.777	379	0.305	4075
SURF-BRISK	8.1	1241	0.428	531	0.823	437	0.352	3604
SURF-ORB	11.5	1241	0.446	553	0.776	429	0.346	3613
SURF-CNN (64, 256)	**0.6**	**1241**	**0.561**	**696**	**0.736**	**512**	**0.413**	**2681**
ORB-SURF	1.3	1795	0.474	850	0.729	620	0.345	7644
ORB-BRIEF	12.3	1795	0.263	652	0.887	578	0.322	8333
ORB-FREAK	30.4	1795	0.176	316	0.845	267	0.149	17007
ORB-BRISK	19.3	1795	0.263	472	0.837	395	0.220	13451
ORB	**15.2**	**1795**	**0.445**	**798**	**0.773**	**617**	**0.344**	**8961**
ORB-CNN (64, 256)	**0.4**	**1795**	**0.399**	**717**	**0.890**	**638**	**0.355**	**7339**
Gradient-based (8-bit)	43.0	955	0.835	797	0.847	675	0.707	939
Gradient-based (16-bit)	**44.0**	**950**	**0.945**	**898**	**0.859**	**771**	**0.812**	**579**
FAST-BRIEF	12.6	1789	0.447	799	0.765	661	0.342	5684
Harris-Laplace-BRIEF	4.9	257	0.436	112	0.866	97	0.377	27804
GFTT-SURF	10.2	1484	0.343	509	0.057	29	0.020	26575
GFTT-BRIEF	**16.6**	**1484**	**0.482**	**716**	**0.846**	**606**	**0.408**	**6524**
GFTT-FREAK	16.4	1484	0.362	537	0.631	339	0.228	7700
GFTT-BRISK	12.3	1484	0.374	555	0.748	415	0.280	7313
GFTT-ORB	16.6	1484	0.480	713	0.823	587	0.396	6306

Table 10: Benchmarking results from TSC 1. The top overall candidates have been highlighted in bold.

Combination	FPS	Keypoints	Match rate	Total matches	Recall	Correct matches	Efficiency	Entropy
SIFT	9.2	521	0.582	303	0.762	231	0.443	6111
SURF	5.6	1173	0.575	675	0.816	551	0.470	2164
SURF-BRIEF	**7.3**	**1173**	**0.610**	**716**	**0.897**	**642**	**0.547**	**2142**
SURF-FREAK	8.3	1173	0.482	565	0.864	488	0.416	2784
SURF-BRISK	7.9	1173	0.535	628	0.884	555	0.473	2443
SURF-ORB	11.0	1173	0.564	662	0.861	570	0.486	2359
SURF-CNN (64, 256)	**0.6**	**1173**	**0.643**	**755**	**0.885**	**668**	**0.569**	**1953**
ORB-BRIEF	13.1	1700	0.400	680	0.903	614	0.361	6788
ORB-FREAK	31.5	1700	0.192	326	0.871	284	0.167	15046
ORB-BRISK	20.3	1700	0.285	484	0.855	414	0.244	11499
ORB	**15.2**	**1700**	**0.499**	**848**	**0.811**	**688**	**0.405**	**7247**
ORB-CNN (64, 256)	**0.5**	**1700**	**0.439**	**741**	**0.818**	**680**	**0.400**	**6051**
Gradient-based (8-bit)	43.9	909	0.813	739	0.846	625	0.688	805
Gradient-based (16-bit)	**37.0**	**1085**	**0.924**	**1003**	**0.851**	**854**	**0.787**	**395**
FAST-BRIEF	25.8	923	0.562	519	0.844	438	0.475	5946
GFTT-SURF	12.4	1433	0.348	499	0.084	42	0.029	17925
GFTT-BRIEF	**13.9**	**1433**	**0.579**	**829**	**0.900**	**746**	**0.521**	**4442**
GFTT-FREAK	17.3	1433	0.441	632	0.756	478	0.334	5013
GFTT-BRISK	12.3	1433	0.498	713	0.846	603	0.421	4513
GFTT-ORB	16.4	1433	0.573	821	0.884	726	0.507	4251

Table 11: Benchmarking results from TSC 2. The top overall candidates have been highlighted in bold.

Combination	FPS	Keypoints	Match rate	Total matches	Recall	Correct matches	Efficiency	Entropy
SIFT	9.7	479	0.595	285	0.772	220	0.459	11714
SURF	5.2	1177	0.586	690	0.826	570	0.484	4325
SURF-BRIEF	**7.7**	**1177**	**0.624**	**735**	**0.901**	**662**	**0.562**	**4341**
SURF-FREAK	9.3	1177	0.493	580	0.874	507	0.431	5589
SURF-BRISK	7.9	1177	0.555	653	0.891	582	0.494	4904
SURF-ORB	11.3	1177	0.585	689	0.867	598	0.508	4699
SURF-CNN (64, 256)	**0.6**	**1177**	**0.649**	**764**	**0.893**	**682**	**0.579**	**3924**
ORB-BRIEF	12.9	1682	0.405	681	0.907	618	0.367	12645
ORB-FREAK	33.2	1682	0.168	285	0.877	250	0.148	31607
ORB-BRISK	21.5	1682	0.260	440	0.859	378	0.223	22690
ORB	**15.2**	**1682**	**0.505**	**855**	**0.820**	**701**	**0.414**	**13290**
ORB-CNN (64. 256)	**0.5**	**1682**	**0.443**	**745**	**0.919**	**685**	**0.407**	**11418**
Gradient-based (8-bit)	41.2	946	0.809	765	0.846	647	0.684	1466
Gradient-based (16-bit)	**35.8**	**1089**	**0.926**	**1008**	**0.852**	**859**	**0.789**	**797**
FAST-BRIEF	29.0	844	0.565	477	0.847	404	0.479	11164
GFTT-SURF	11.9	1492	0.346	516	0.091	47	0.032	34021
GFTT-BRIEF	**13.3**	**1492**	**0.589**	**879**	**0.903**	**794**	**0.532**	**8266**
GFTT-FREAK	15.3	1492	0.452	674	0.764	515	0.345	9350
GFTT-BRISK	12.7	1492	0.513	765	0.850	650	0.456	8377
GFTT-ORB	16.9	1492	0.587	876	0.889	779	0.522	7888

Table 12: Benchmarking results from TSC 3. The top overall candidates have been highlighted in bold.

In figure 5.14, the number of keypoints extracted by the feature extraction candidates across the three datasets recorded at the TSC are shown. It can be observed that, except for the gradient-based features, the number of extracted features decreases as the temperature increases. This is rather counter intuitive as it can be expected that the number of features increases with temperature as the thermal gradients, and thus image texture, then also increase. This phenomenon is exactly what results in more features being detected using gradient-based features in the later datasets. With the remaining candidates, it still occurs but the features' locations are favored in areas that already have a high thermal gradient in the cold dataset such as around the fire source and at the ceiling. As the smoke warms the environment, a number of these locations are then too hot and are thus masked out of the detection.
Notably, SIFT does not provide a satisfying amount of features. As explained in Appendix B.4, SIFT extracts features by creating a pyramid of different Gaussian-blurred versions of the image at increasingly smaller resolutions. Both resizing the image to a smaller size as well as blurring the image leads to significant texture loss. As thermal images already have limited contrast, such operations result in few distinct blob centers.
The benefits of ORB with respect to FAST are visible in the amount of detected features across the second and third TSC dataset. Detecting FAST corners over multiple image scales here results in more features detected in heated regions whereas FAST struggles to find features outside of regions with high thermal gradient.

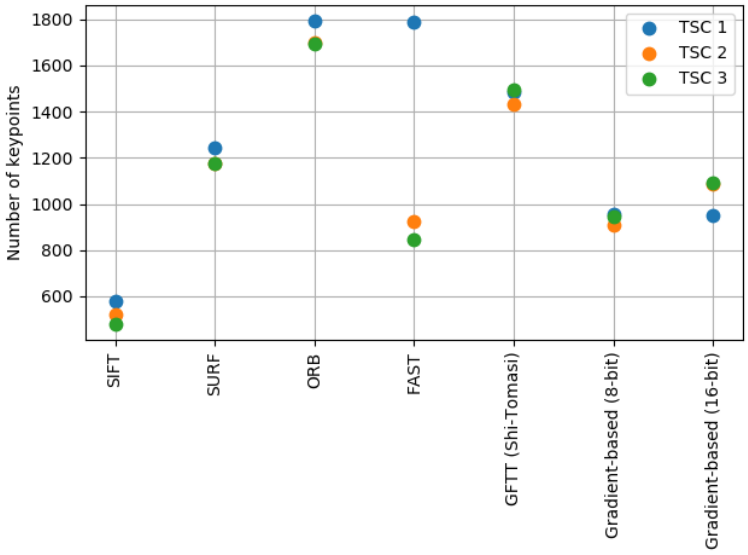


Figure 5.14: Number of keypoints per extractor over all three TSC datasets

Figure 5.15 shows the different rate metrics from the third TSC dataset. It can be seen that blob features present an advanced matching rate with respect to corner features. The BRIEF descriptor appears to be the best classical descriptor as it offers a high matching rate and recall with all tested feature extractors. In general, the performance difference between float-point and binary descriptors is less apparent compared to visual images. Binary descriptors compare image intensities at select locations to formulate the descriptors, making it more robust against image noise typically associated with thermal images. The FREAK binary descriptor is the worst performing descriptor, which is notable as it was proposed as the best option in combination with SURF features for thermal odometry by T. Mouats et. al. [49].
When matching of SURF features, CNN-based matching offers the highest matching rate and recall. For ORB features, it provides an improved recall compared with the ORB descriptor but has a slightly lower matching rate resulting in the same efficiency for the two. It can be observed that CNN-matching, in terms of efficiency, performs at least equally to the top classical descriptor for a given feature extractor. The increase in gained performance is however not groundbreaking.

In terms of corner matching, Shi-Tomasi corners offer better matchability than ORB corners. Looking at the rates, FAST corners also seem to have greater performance with the BRIEF descriptor compared to ORB features with the same descriptor or its improved variant in ORB.
The overall best efficiency is achieved by gradient-based feature tracking, meaning that the method results in the most correct matches given a number of detected keypoints. Both the 8-bit and 16-bit version obtain a higher matching rate than the descriptor-based approaches, which can be explained by a more local search, i.e. (partially) along the corresponding epipolar line. The use of an entire image patch around the query feature provides a less ambiguous description of its environment compared to the select number of pixels in a descriptor. This can especially be seen by the further increase in the matching rate when adopting 16-bit tracking.

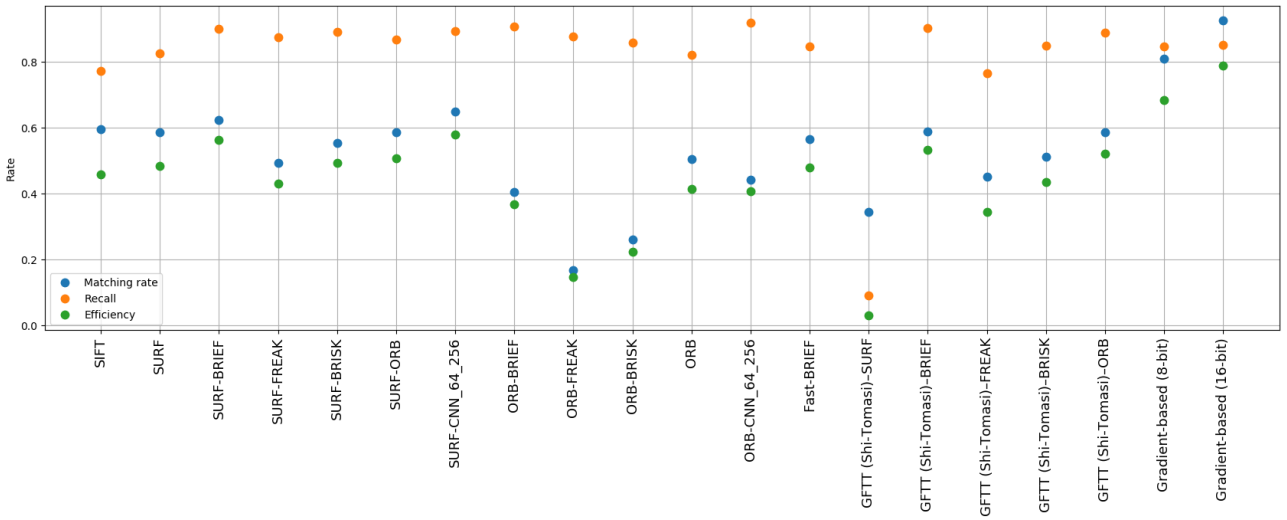


Figure 5.15: Rate metrics from TSC 3

Figures 5.16 and 5.17 display the decrease in rate metrics respectively in the coldest dataset TSC 1 and the intermediate dataset TSC 2 with respect to the warmest dataset TSC 3. There are no noticeable differences between TSC 2 and 3 where the decrease from all candidates, with some exceptions, is not higher than 5%. An increase in matching rate can be observed from ORB-FREAK and ORB-BRISK but these two combinations obtained the lowest two matching rates TSC 3, making this slight increase rather insignificant.
A decrease in performance is however visible for all candidates in the coldest dataset with respect to the warmest. No clear difference is visible between corner and blob features. The extractor that is the least affected by the drop in temperature is ORB. For SURF blob features, the decrease is similar for all descriptors whereas for Shi-Tomasi corners, the decrease in performance is more dependent on the descriptor used.

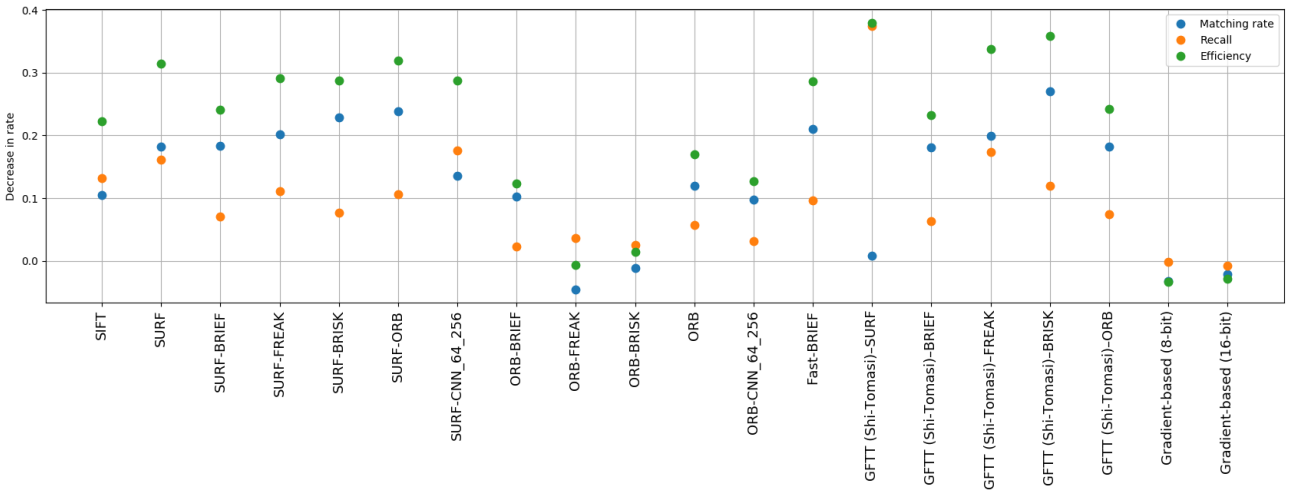


Figure 5.16: Decrease in rate metrics in TSC 1 with respect to TSC 3

The matching rate looks to be more impacted than the recall for all candidates. CNN-based matching does not seem to provide more robustness to temperature changes compared to the classical descriptors.
The most robust performance can be observed by the gradient-based feature tracking approach, which seems to be agnostic to the temperature changes by displaying a slight increase in the matching rate both for 8-bit and 16-bit.

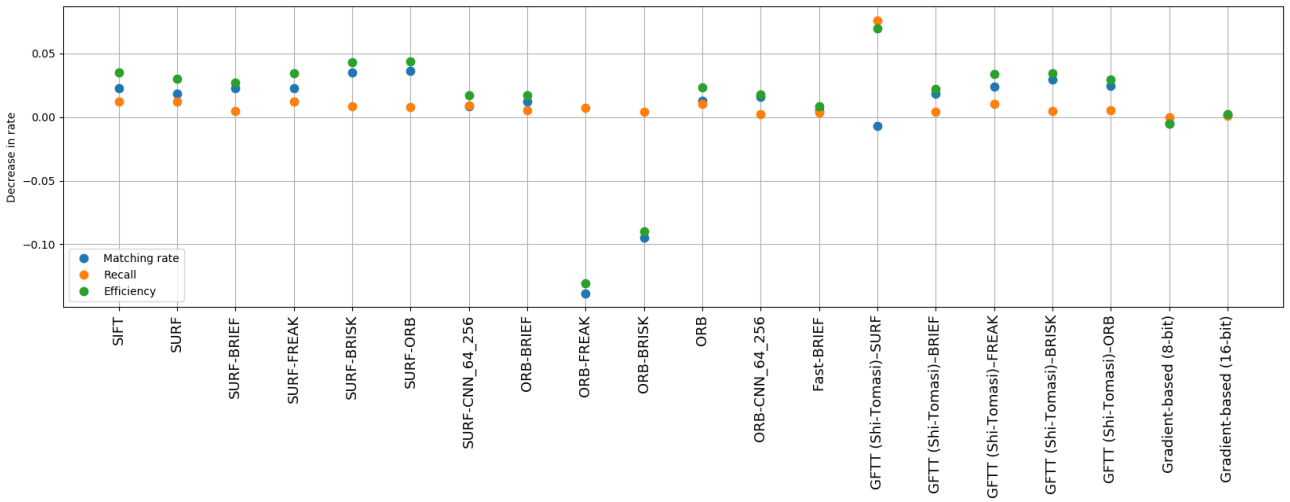


Figure 5.17: Decrease in rate metrics in TSC 2 with respect to TSC 3

The lower mean entropy per frame of all SURF extractor combinations in figure 5.18 compared to all combinations with corner extractors suggests that blob features result in a more uniform distribution of the features used for the odometry estimation. The lowest entropy is however obtained by the gradient-based feature tracking. This can largely be accredited to its ability to locally adapt its detection threshold and thus maximize the number of quality features in all regions of the image.
A pattern in the entropy between the different descriptors for all extractors is visible, where the lowest entropy is obtained by the CNN-based descriptors followed by the BRIEF descriptor. FREAK results in the highest mean entropy for most extractors, with the exception of the SURF descriptor in combination with Shi-Tomasi corners. SIFT has a higher entropy mainly due to the general lack of features to match compared the other blob extractor, SURF.
No significant difference in mean entropy is visible across the three datasets. In general, the mean entropy is slightly higher in TSC 1 for the blob extractors and Shi-Tomasi whereas it is slightly lower for ORB and FAST. The latter in combination with the overall higher entropy of mean that ORB

features are more likely to be located in regions of higher thermal gradient, i.e. around the fire source and the ceiling, which become more striking as the temperature increases. As ORB tries to detect a prescribed number N_f of features, the N_f features with the highest corner responses are selected, likely located in these regions.

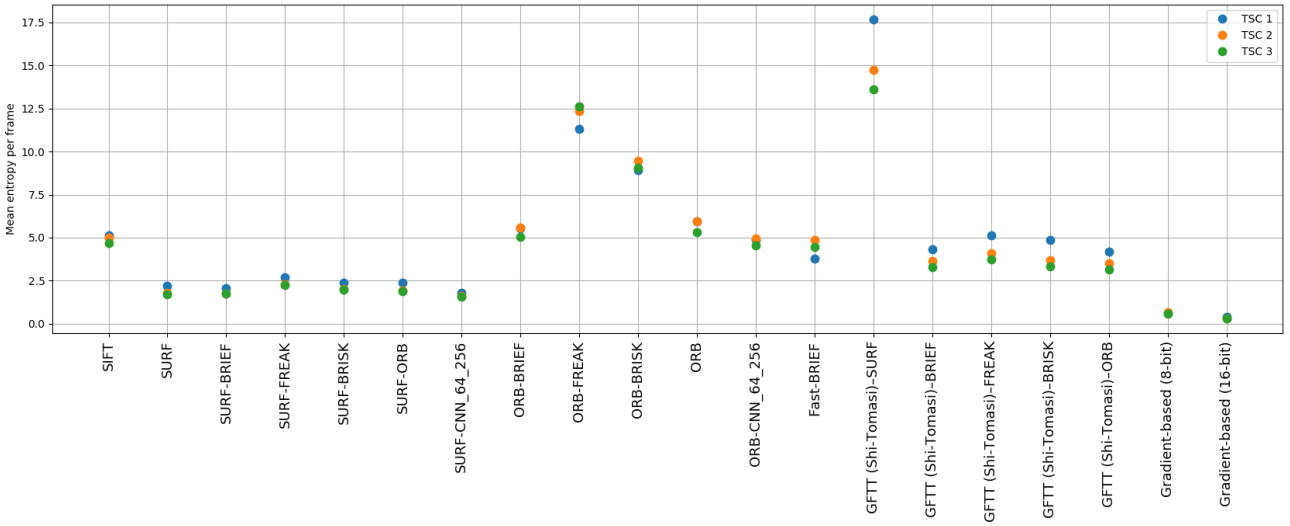


Figure 5.18: Mean entropy per frame over the three TSC datasets per candidate

From the average frames per second for each candidate in figure 5.19, it can be observed that binary descriptors result in a faster run-time compared to float-point descriptors. Especially the CNN-based descriptors are too slow to operate in real-time conditions necessary for SLAM. Despite being known to be rather slow, SIFT here demonstrates a higher processing rate due to the low amount of detected features. Similarly, FAST-BRIEF here also has a higher lower average computational time due to the lower amount of matchable features compared to TSC 1.
The fastest processing rate is achieved by the gradient-based approach, namely due to its localized feature tracking rather than global feature matching as done by the other candidates.

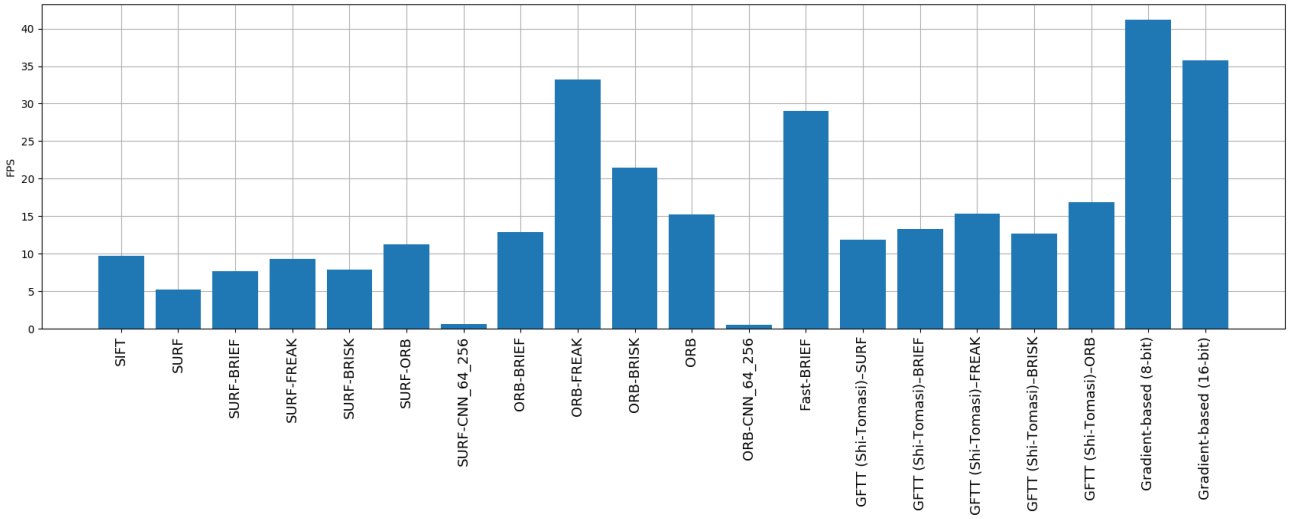


Figure 5.19: Average frames per second per candidate in TSC 3

In table 13, the top candidates are compared according to the five most important criteria for feature extraction and matching in a SLAM algorithm. Table 14 provides further explanation of the different ratings in the form of a scoring range from 0 to 10.
Despite the overall excellent performance of gradient-based feature tracking, its feature extraction has

no repeatability meaning that features are never detected at the same location on an object seen from different perspectives. It is therefore not suitable for feature-based loop closure detection, as utilized by the chosen SLAM framework ORB SLAM 3.
The CNN-based descriptors showed to be a promising approach for feature matching in thermal images but are nonetheless not ground-breaking compared to the classical descriptors. Additional training using blob features and training data in a larger variety of environments are future steps that could be taken to further increase its performance. A well-known problem of CNNs are their lack of flexibility, meaning that high performance in a certain test environment does not guarantee similar performance in a different location. Flexibility however is a property that is very important for firefighters, together with reliability, as they face time-sensitive situations in diverse unknown environments. The slow processing time of CNN-based descriptors could be remedied by using more powerful hardware, notably a better GPU.
The ORB feature extractor and descriptor provides a high number of keypoints and are designed for real-time applications. Its efficiency is however only sufficient, meaning that more features need to be detected to obtain a given amount of usable matches. Its keypoint distribution across the image is also not the best, which impacts both the odometry estimation as well as the detailing of the 3D-map.
SURF-BRIEF and GFTT-BRIEF exhibited similar performance. GFTT shows a slightly higher processing rate whereas SURF provides a better feature distribution, which is also more consistent over the three TSC datasets. Blob features are situated on surfaces, enabling them to better describe the environment's surfaces in contrast to corners. Given this property, SURF-BRIEF is preferred over GFTT-BRIEF and will therefore be selected as the candidate to be implemented inside the ORB SLAM 3 SLAM framework.

candidate	**Number of keypoints**	**Efficiency**	**Distribution of keypoints**	**Real-time**	**BoVW compatible**
SURF-BRIEF	Good	High	Very good	Sufficient	Yes
SURF-CNN	Good	High	Very good	No	Yes
ORB	High	Sufficient	Sufficient	Very good	Yes
ORB-CNN	High	Sufficient	Good	No	Yes
GFTT (Shi-Tomasi)-BRIEF	Good	Very good	Good	Good	Yes
Gradient-based (16-bit)	Sufficient	Excellent	Excellent	Excellent	No

Table 13: Comparison of top candidates

Rating	Insufficient	Sufficient	Good	Very Good	High	Excellent
Score	< 5.5	[5.5, 6.5)	[6.5, 7.5)	[7.5, 8.5)	[8.5, 10.0)	10.0

Table 14: Rating and associated score range

5.5 Thermal-inertial SLAM

5.5.1 Loop closure detection

Before integrating the best feature extractor and descriptor from the performed benchmark into ORB SLAM 3, its performance in detecting potential loop closures is tested.
The loop closure candidates are detected using SURF features and the BRIEF descriptor. A vocabulary tree of six layers and ten branches per layer was adopted, as utilized by ORB SLAM 3 [14]. Figure 5.20 shows the similarity scores between the 250 keyframes included in the experiment computed using eq. (45). The similarity scores are low in general, not exceeding 0.3. A lot of the higher scores are obtained between images captured in proximity to each other (within 1.5 seconds). Two high

scores standing out that are separated further apart are at frame 100 and 30, and at frame 175 and 65.

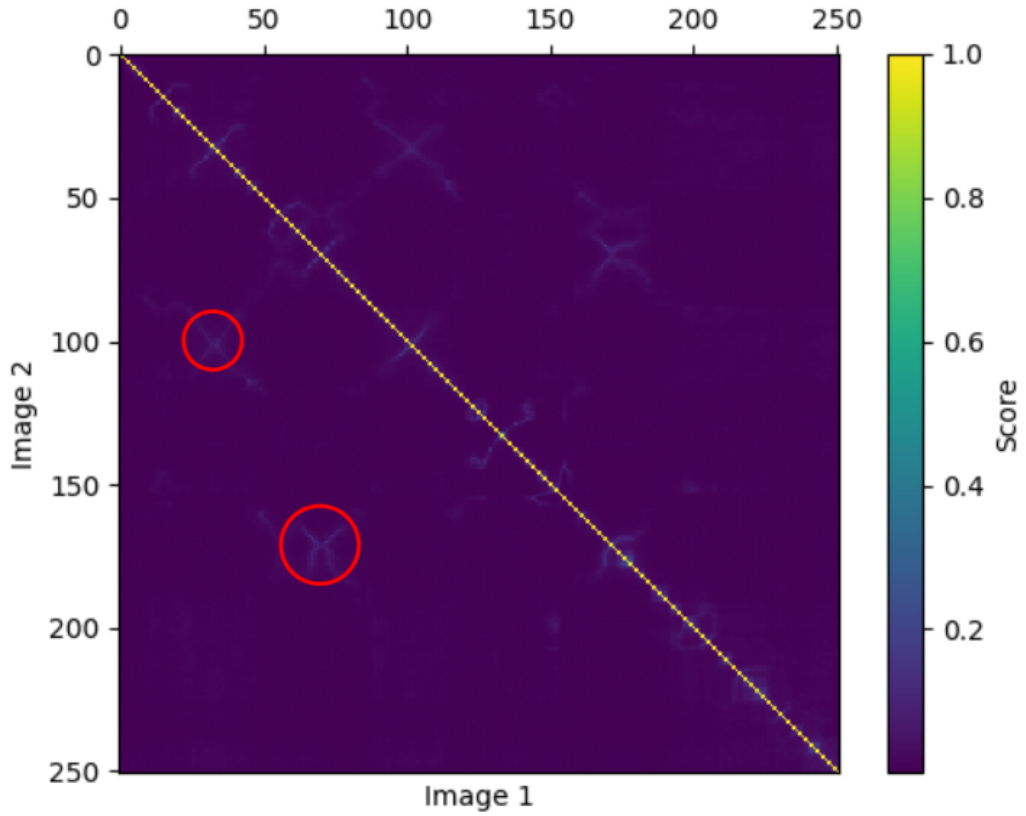


Figure 5.20: Similarity score heatmap between all participating images from TSC 3. The two major potential loop closures are circled in red. Note that the heatmap is symmetrical with respect to its diagonal.

Table 15 shows the fifteen possible loop closures detected by the DBoW2 library. To obtain the candidates, a similarity threshold of 0.095 and a maximum RMS error of 20.0 pixels was used as to maximize the true potential candidates without obtaining multiple consecutive matches for a single query image. Loop 2 and 3 as well as loop 6 and 7 correspond to the approaching and backing away from the car situated in the left side of the garage, as in figure 5.21. No loop closure candidate was obtained between the two separate visits to the left car however, suggesting that the robot's location was not close enough to qualify as a loop closure. The remaining loop closure candidates obtained that have a time difference within a couple seconds occur in the same manner, when the vehicle changed its driving direction.

Loop	Query image	Match image	Score	RMS error	Time difference
1	17	13	0.0997	16.71	2.0 s
2	35	31	0.113	9.000	2.0 s
3	36	30	0.132	15.23	3.0 s
4	72	68	0.139	8.229	2.0 s
5	74	66	0.115	18.19	4.0 s
6	104	99	0.129	9.300	2.5 s
7	105	98	0.156	13.13	3.5 s
8	114	45	0.0956	14.37	34.5 s
9	141	128	0.110	17.70	6.5 s
10	156	145	0.108	13.96	5.5 s
11	168	66	0.131	4.763	51.0 s
12	169	67	0.148	3.630	51.0 s
13	170	68	0.133	6.187	51.0 s
14	171	69	0.0107	15.43	51.0 s
15	174	169	0.102	19.23	2.5 s

Table 15: Loop closure candidates detected in TSC 3 using the bag-of-visual-words approach

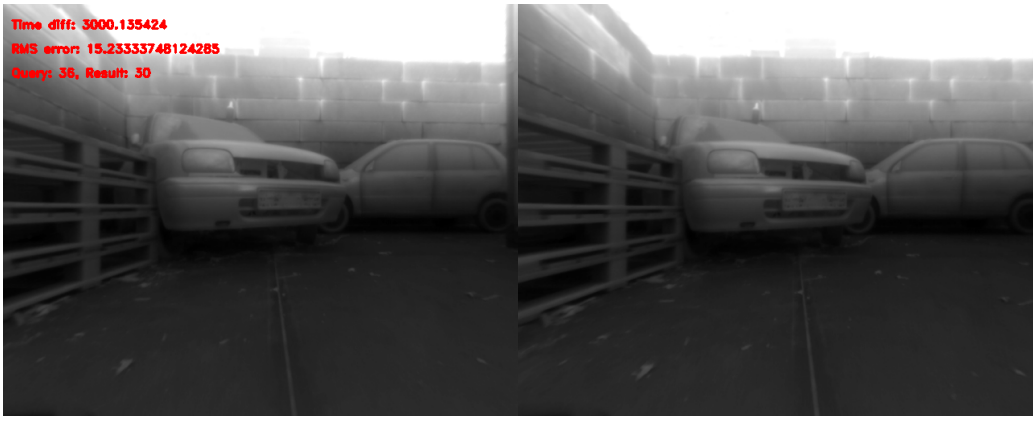


Figure 5.21: Loop closure candidate 3

The two peaks in similarity score found in figure 5.20 where detected in loop 8 and loops 11-14 respectively. The former is obtained as the UGV is located in the middle of the garage and faces the scooter and pillar, as seen in figure 5.22. The latter occurs as the robot visits the burning car again, seen in figure 5.23, 51 seconds after the first pass.

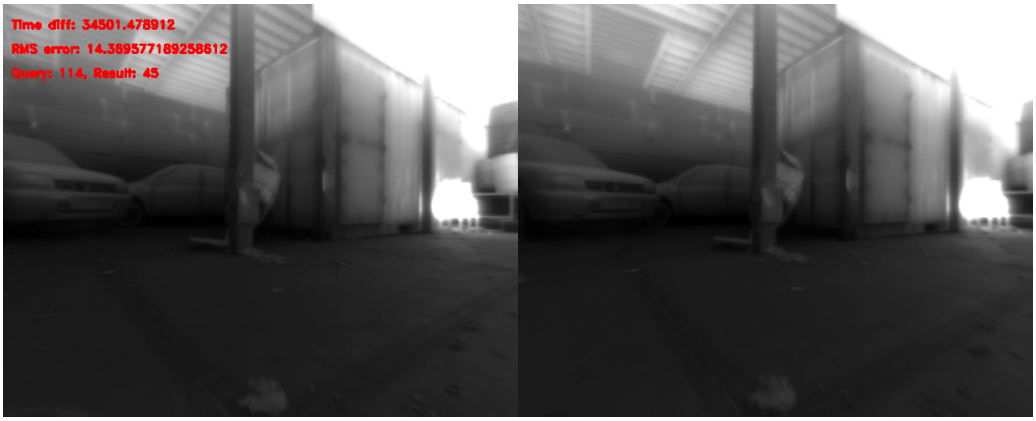


Figure 5.22: Loop closure candidate 8

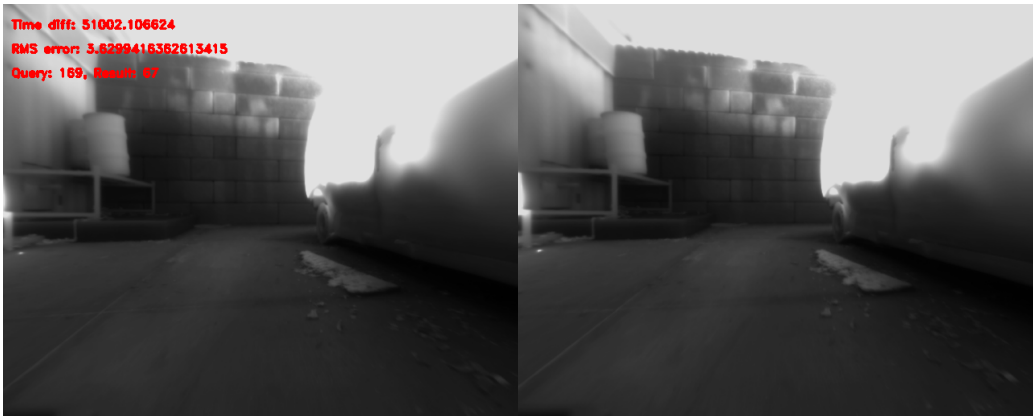


Figure 5.23: Loop closure candidate 12

From these results, it can concluded that using SURF features in combination with the BRIEF descriptor is a viable option to detect loop closure candidates. This feature extractor and descriptor combination can therefore be implemented in ORB SLAM 3.

5.5.2 Trajectory estimation

In this final part of the results, the SLAM results of the proposed Firebot TIO algorithm will be displayed.
The obtained results are compared with the point cloud and trajectory computed by ORB SLAM 3. When using ORB SLAM 3, the stock implementation is used meaning that only the ORB feature extraction and description settings are modified with the values in section 4.4.1. Only the feature extraction exclusion mask, hiding areas near the fire source, has been incorporated.

5.5.2.1 Thermal SLAM

In a first part, the performance of thermal-based SLAM only (no inertial data) was tested to investigate the capabilities of SLAM inside a smoke-filled environment using thermal images.
In figures 5.24, 5.26 and 5.27, the obtained trajectory and 3D point cloud obtained from TSC 1-3 respectively are superposed over the floor plan of the building. Although no ground truth is available, this will provide sufficient insight to make a qualitative assessment of the algorithm's mapping accuracy.
The floor has been omitted from the point clouds by removing all points with a height below 5 centimeters. Additionally, outlier removal was performed by deleting all points that have less than eight neighbors in a 25 centimeter radius.

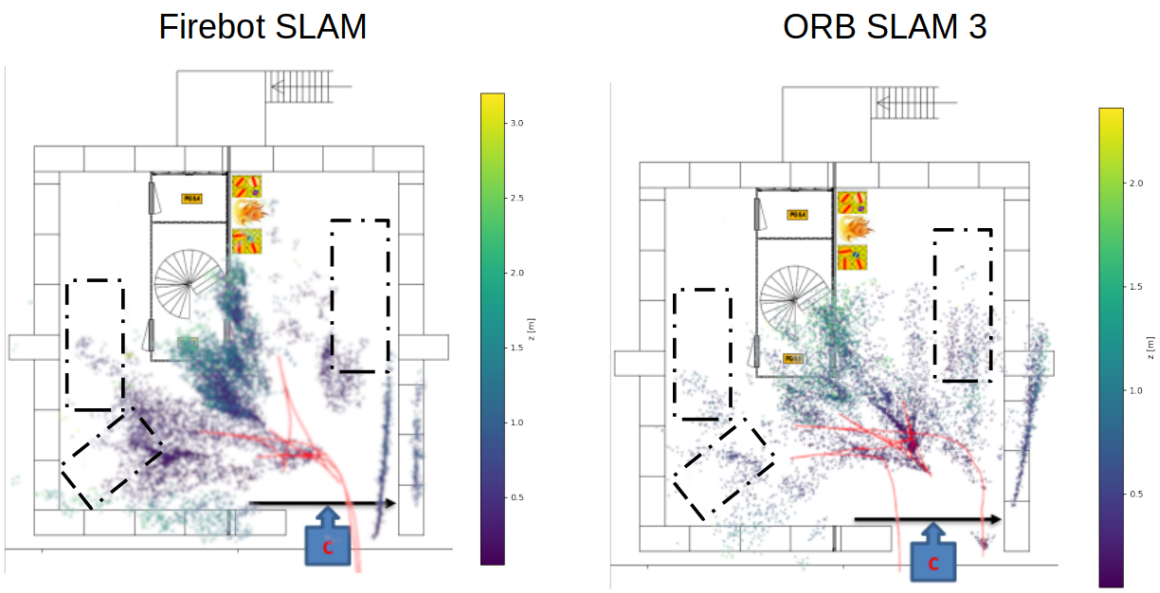


Figure 5.24: Thermal-only SLAM results of the proposed method and ORB SLAM 3 on TSC 1 (1 min 15s). The dotted lines represent the approximate location of the cars.

During the coldest recording in figure 5.24, the obtained trajectory by the proposed algorithm, at first glance, strongly resembles the trajectory deduced by the thermal video. In figure 5.25, the layout of the TSC parking garage's content can be seen. All objects present in the room are at least partly mapped by the algorithm. The colder hood of the burning car located in the back right is correctly located as is the container wall to the left of it. The exact placement of the container however possesses a large uncertainty given by the large scatter of points. This side of the container is the most seen element in the room as the robot navigates between the left and back side of the building. The scatter of points near the container wall is then the result of the accumulation of position errors over time.
Some additional noise is also visible near the left car and pallet stack, corresponding to ground points where the height has been erroneously computed. The column and the scooter in front of the container are mapped in TSC 1 but are not precisely detailed enough to be recognized.
In the front left side, a number of 3D points are generated from the left car, the pallet stack and the wall. The latter again displays some uncertainty in its location and most of its points are located towards the hotter ceiling.
Despite that the loop closure between the vehicle entering and exiting the garage was detected, two walls have been mapped. The walls visible in the depth direction of the camera (left and back wall) are not created in the point cloud. This can mainly be attributed to the fact that the vehicle did not sufficiently approach these walls, leading to a lack of features. Additionally, the depth estimation error increases with the distance between the object and the camera, possibly leading to erroneous points on these walls.
The main difference between the trajectories obtained by Firebot SLAM and ORB SLAM 3 are the lack of loop detections in ORB SLAM 3. The latter mainly failed in detecting both the loop closure when the UGV visits the front left car for the second time and when the robot exits the garage in reverse through the opening it entered.
TSC 1 was recorded a number of minutes after ignition. In figure 5.25, it can be noticed that the hot smoke, seen by the glowing haze, is located towards the ceiling as in the simulation results described in Appendix A. Although the smoke causes a haze, it does not seem to negatively affect the feature locations and consequently the odometry estimation. On the other hand, the hot smoke is heating the structure leading to an increase in texture compared to objects lower to the ground.

Figure 5.25: Front left side (left) and back right side of the TSC parking garage in TSC 1

The shortest recorded dataset, TSC 2, displays a rotated point cloud of the right wall with respect to the estimated path, as shown in figure 5.26. These rotations only seem to happen to objects visible in the left or right image border as the vehicle drives alongside it. The correct orientation is however always obtained when the UGV faces a scene head-on. This occurs when the robot is facing the scooter, which leads to the correct mapping of the container wall behind. The pillar and scooter do also have the correct orientation but the latter again does not have enough detail to be recognized. The back, front and left wall of the building are here again not mapped. Part of the front of the left car as well as part of the pallet stack are represented in the point cloud.
In contrast to TSC 1, the loop closure between when the vehicle enters and exits the garage has been detected in TSC 2.
The point cloud produced by ORB SLAM 3 is less dense, with most points from the container being located near the hot ceiling. The algorithm again failed to detect the loop closure between when the vehicle enters and exits the building, leading to a double mapping of the right wall.

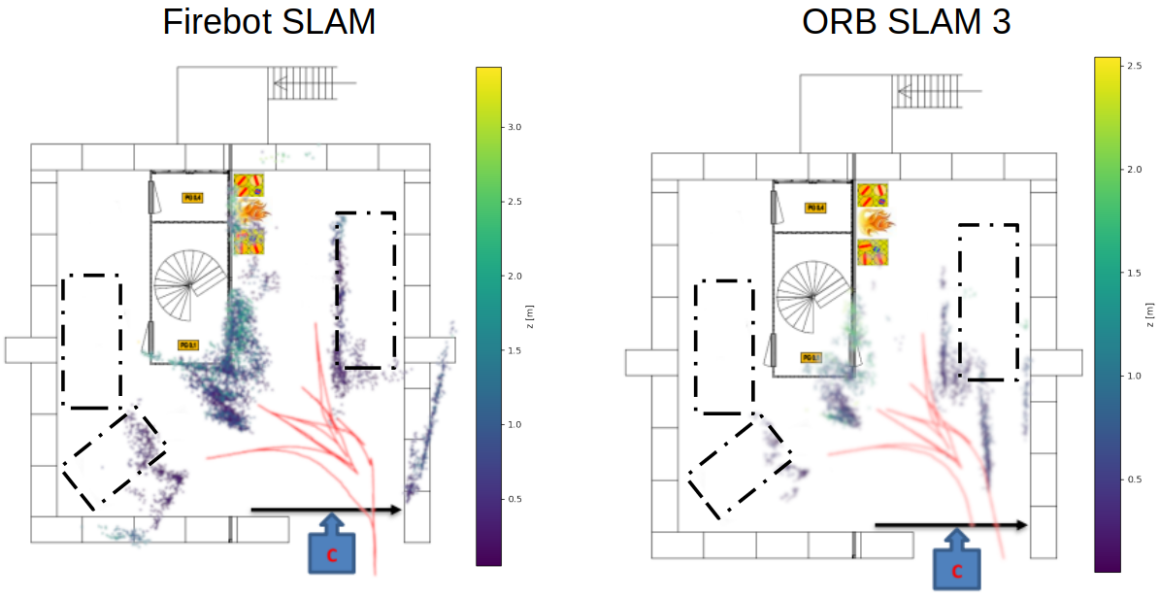


Figure 5.26: Thermal-only SLAM results of the proposed method and ORB SLAM 3 on TSC 2 (1 min 1s). The dotted lines represent the approximate location of the cars.

In the longest and hottest dataset, TSC 3, Firebot SLAM looses its position as it entered the garage. As in this dataset, the robot does not create a loop closure when entering and exiting, this first portion of the map is never merged again with the second point cloud.

The heading of the walls computed by Firebot SLAM is erroneous in the front left part of the garage by about 10°, as can be viewed in figure 5.27. This is despite that the algorithm detects all the loop closures, including the major ones as obtained by the candidates in section 5.5.1.
In TSC 1 and 2, the robot exits the garage in reverse, whereas in TSC 3, the robot rotates before facing the door as it exits. During this final turn, the heading error significantly increases, as seen by the garage door represented by the cyan blue cluster at the entrance in figure 5.27. The large increase in error is mostly attributed to the lack of diverse features as only the door and wall are visible. Additionally, as it is the end of the dataset, no loop closure detection followed to correct the deviation.
In all three datasets, the back wall is never included in the point cloud as the feature density there is was not enough. Note that as the temperature rises, hotter parts of the environment are masked out from feature detection. Although this mainly affects the fire sources and parts of the ceiling, the larger portion of the burning car in the back is now also omitted from feature detection, explaining the lack of modelling despite the increased thermal radiation in that area
The point cloud created by ORB SLAM 3 is here again less dense compared to Firebot SLAM. ORB SLAM 3 did also not detect the major loop closures. The smaller part of the front wall is mapped and the garage door, as with Firebot SLAM, modelled with the wrong orientation. ORB SLAM 3 did however not loose its position in the beginning.

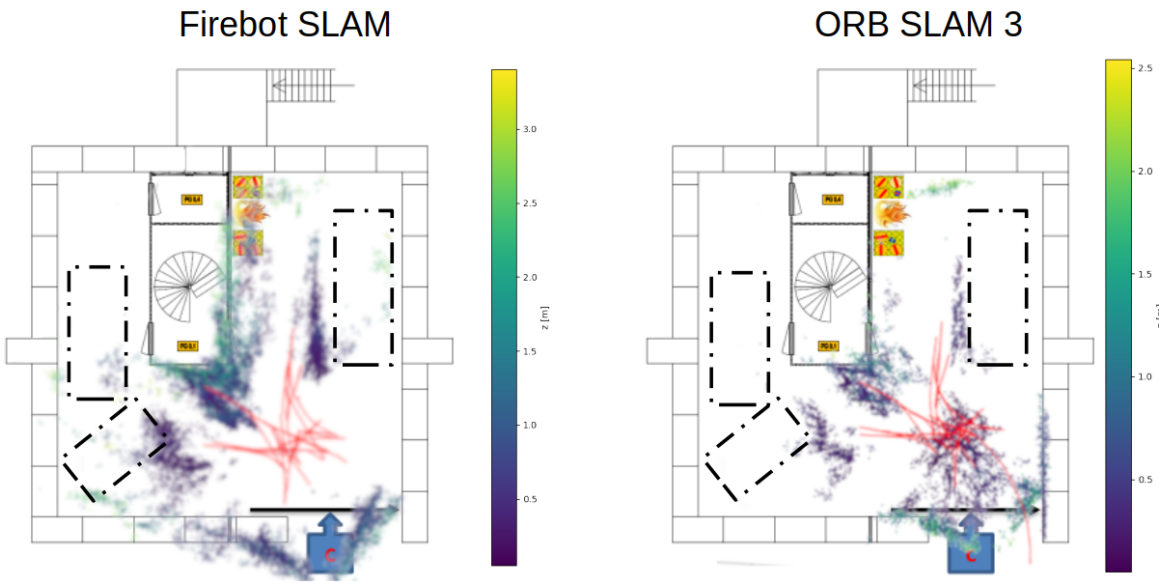


Figure 5.27: Thermal-only SLAM results of the proposed method and ORB SLAM 3 on TSC 3 (2 min 5s). The dotted lines represent the approximate location of the cars.

For all three datasets, a total processing time of 200-250 milliseconds is required per frame.

Testing the accuracy of the proposed SLAM algorithm only using thermal data in the Optitrack room, it can be noticed in figure 5.28 that the system failed to recognize any motion of the robot. In figure 5.29, it can be seen that the differences in infrared emissivity in the Optitrack room are much lower than inside the parking garage at the Twente Safety Campus. As a result, the obtained features are much less distinctive.

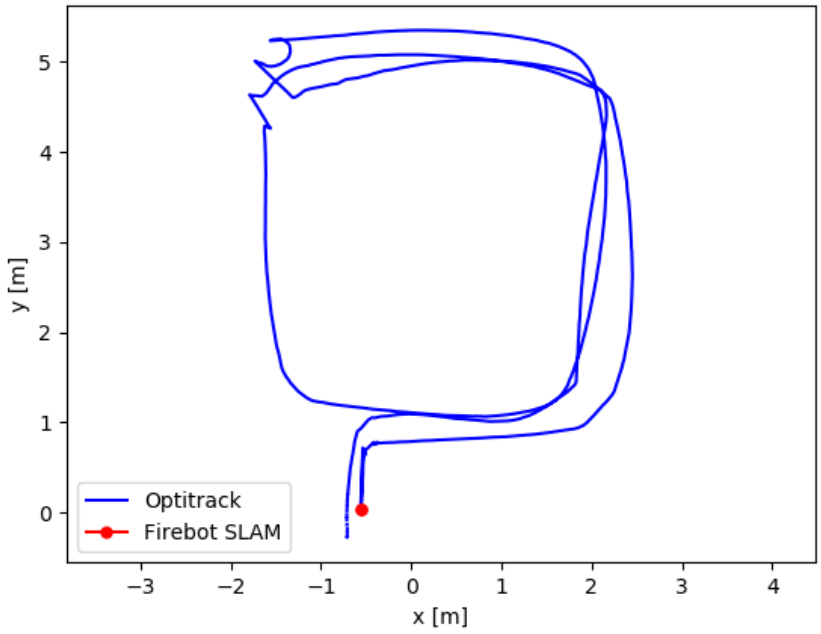


Figure 5.28: Thermal-only SLAM and Optitrack ground truth trajectory in the Optitrack room

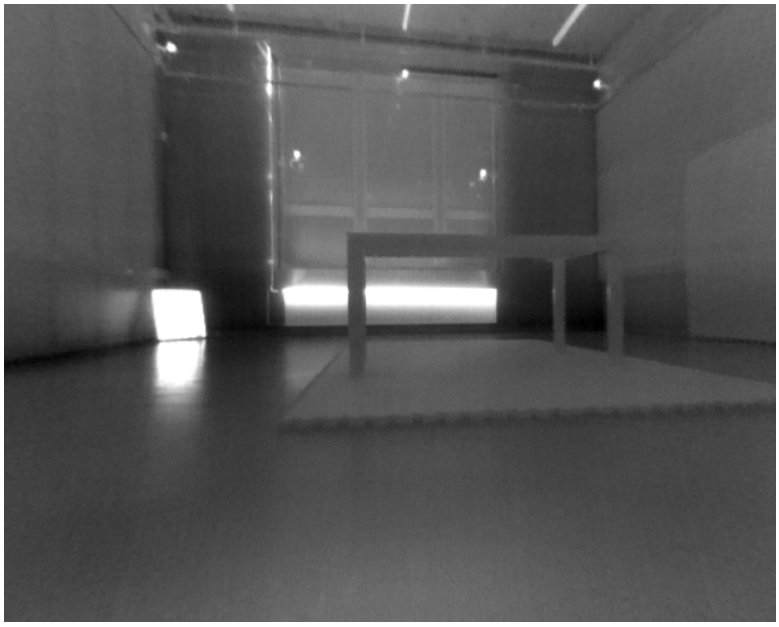


Figure 5.29: Thermal view of the Optitrack room

5.5.2.2 Thermal-inertial SLAM

As explained in section 5.2.2, the XSens Mti-620 included in the setup provided for this thesis had a deficient accelerometer in the driving direction, rendering its data useless. Therefore no thermal-inertial SLAM results from the TSC datasets could be obtained.
Nonetheless, a comparison of the computed absolute velocity on a trajectory in the Optitrack room obtained using only the inertial data from the deficient Mti-620 and the spare Mti-630 is shown in figure 5.30. Due to the accumulation of integration errors from the measurement noise, it is inevitable for the inertial-based velocity to diverge at a given moment. The difference at which the divergence occurs between the two IMU in figure 5.30 is however striking. In the case of the deficient Mti-620, the velocity is only correctly estimated for the first 3 seconds of the trajectory before a significant

jump is observed compared to the Optitrack ground truth. The Mti-620 does not yet diverge however as it still mostly estimates the shape of the ground truth velocity curve, with an added offset, up to 9 seconds after departure. On the other hand, the spare Mti-630 does not introduce significant jumps in the velocity curve and a significant error and consequently divergence is only observable after 9.5 seconds. The difference in the time interval where the two IMU provide a stable velocity estimate is a direct result of the higher noise present in the Mti-620 as shown by the Allan curve in figure 5.5.

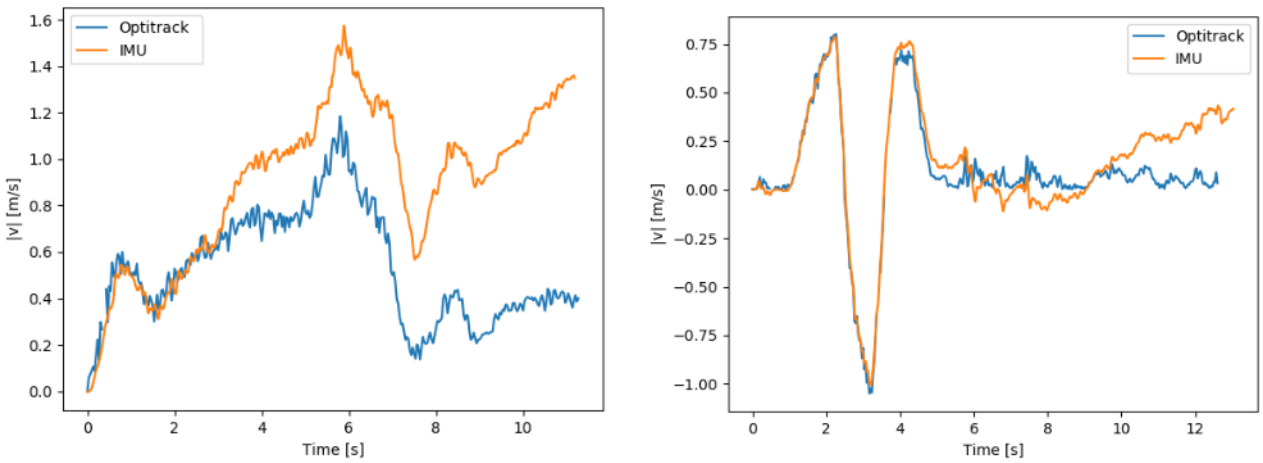


Figure 5.30: Inertial-only absolute velocity computed from a trajectory using the deficient Mti-620 (left) and the spare Mti-630 (right)

While testing the proposed thermal-inertial SLAM algorithm in the Optitrack room, the addition of the inertial data from the spare XSens Mti-630 did not improve the computed trajectory as the system failed to initialize the IMU. During the initialization of the IMU, the algorithm estimates the orientation of the vehicle with respect to gravity as well as the IMU biases and the scale between the thermal and inertial odometry (equal to 1.0 for stereo odometry). This process is performed in three steps in ORB SLAM 3 [14], the framework responsible for the sensor fusion in Firebot SLAM. First a thermal-only trajectory is computed over a number of frames followed by an inertial-only initialization where a Maximum A Posteriori estimation (MAP) computes the IMU biases and rotation matrix between the thermal-only world coordinate frame (first frame of the left camera) and the gravity coordinate frame (z-axis up). During this step, if the IMU is not actively excited around all three axes, the solution may interchange the pitch and roll axes of the IMU. In the third and final step of the initialization, a thermal-inertial bundle adjustment is performed to merge, optimize and verify the two previous steps. In the case of the Optitrack room, the thermal-only odometry failed giving no data in the first step of the initialization. A ground robot is also used thus making it impossible to excite all three axes of the IMU. This resulted in the solution of the second step switching between two solutions between iterations. Setting the biases estimated in the IMU calibration of section 5.2.2 to fixed during this step, removing six unknowns, did not improve the results. If a rather correct thermal-only odometry would have been obtained beforehand, the confusion of the two possible solutions in the second step could have been resolved by comparing the thermal and inertial odometry in the third step, the thermal-inertial bundle adjustment. The wrong solution will incorrectly remove the gravity component from the inertial measurements. The solution which minimizes the difference in computed thermal and inertial odometry would then be accepted.

6 Discussion

6.1 Hardware calibration

6.1.1 Thermal cameras

Calibrating the thermal cameras proved to be more challenging than calibrating visual cameras. As colors are not visible in the long-wave infrared spectrum, the calibration pattern needs to be constructed by generating variances in infrared emissivity from two different materials and/or from temperature differences. Furthermore, it is important to heat the board in an equal manner as to not create additional blobs that could be mistaken as circles from the calibration pattern. After having recorded multiple datasets, the calibration board had endured multiple heating and cooling cycles causing it to warp. If it is not straightened, the lens distortion which is a lot more prevalent in thermal cameras, is erroneously calculated. Additionally, to obtain a good calibration, it is important to move the board slowly as to prevent motion blur. However, moving it too slowly will cause the board to cool down to a point where the calibration pattern cannot be found anymore. A large number of recordings therefore had to be made to find a good balance between the two.

6.1.2 Inertial Measurement Unit

Due to the resolution of the XSens Mti-series, the scale factor and misalignment of the gyroscope could not be determined using the Earth's rotation rate. As a turntable or any other rotating platform with known constant angular rate was not available, it was therefore decided to utilize the scale factor and misalignment from the accelerometer which are, according to the manufacturer's documentation, of the same order.

6.1.3 Thermal-inertial system

Calibrating the thermal-inertial system was the most difficult calibration step. For visual-inertial calibrations, a number of papers propose different methods to perform this calibration but only two are publicly-available software, Kalibr by P. Furgale et. al. [66] and Crisp by H. Ovren´et al. [67]. The former can estimate both the rotation and translation whereas the latter only returns the rotation. Both perform a time-synchronization of the data and estimate the IMU biases. Practical experience showed that both libraries have little to no flexibility and are therefore not suited to work with thermal images. Kalibr utilizes the default settings of the OpenCV blob detection to find the calibration pattern which showed to be insufficient to obtain satisfying results. In contrast to Kalibr, Crisp does not use a calibration board but instead tracks Shi-Tomasi corners using sparse Lucas-Kanade optical flow. It is evident that tracking corners in a dataset only composed of a circular pattern does not function. Modifying Crisp to track the points of the calibration pattern did not yield better results due to the repetitive nature of the grid. Using the gained knowledge from utilizing the previous two libraries, it was decided to create the simplified calibration method suitable for thermal images used during this project. Here again, an optimal oscillation speed needed to be found as to obtain clear motion measurements without letting the board cool down. The accurate time synchronization of the obtained angular rates also proved to be a crucial part into obtaining correct results.

6.2 Feature extraction and matching benchmark

During the feature extraction and matching benchmark, it was observed that blob features offer slightly more robustness as they are more uniformly-distributed across the frame than corner features. Corner features on the other hand, by their nature, are more concentrated in image locations with a high image gradient. In thermal images, these locations correspond to areas of higher thermal gradient such as close to the heat source and the ceiling. Blob feature extractors provide, although still plenty in the case of SURF, a smaller amount of features compared to corner extractors. On the other hand, the matchability, as well as the overall efficiency, of blob features is higher than corner features. This means that less blob features than corners need to be detected to obtain an identical number of correct and useful feature matches.

In visual images, a trade-off between computational speed and recall needs to be made when using binary descriptors. This was not needed during this benchmark as binary descriptors, notably BRIEF and ORB (a rotation-invariant version of BRIEF), performed better than the floating-point descriptors at a faster rate. The efficiency of a given descriptor appears to be dependent on its environment of application. The results from the three benchmarks, by Johansson et. al. [13], T. Mouats et. al. [49] and the one included in this thesis, appear to differ. The top descriptor in Johansson's et. al. benchmark is a float-point descriptor, LIOP, whereas in the remaining two, a binary descriptor was found to be the top contender. T. Mouats et. al. tested in benign environments and suggested the use of FREAK based on their findings, which performed as the worst descriptor in the benchmark performed here. Preliminary experiments conducted inside the Saxion building (excluded from this thesis) were however more in line with T. Mouats et. al. where SURF-FREAK obtained the most correct matches, followed by SURF-BRIEF.
The binary descriptors BRIEF and ORB showed to have a higher matching rate and recall compared to the float-point descriptors SIFT and SURF. The latter compose their descriptors from the image gradient, which in thermal images has a lot less information compared to visual images due to the low contrast. Using binary comparison of pixel intensities appears to be a more robust method of describing thermal features. Additionally, the random testing pattern from (r)BRIEF is much more effective than the circular testing patterns from BRISK and FREAK.
Gradient-based feature tracking showed great potential for a very efficient method to detect and track features across thermal frames. Its feature detection however provides no repeatability, making the features unsuitable for loop closure detection. Using the tracking method with the radiometric data gave invaluable insight into the additional information and thus performance the 16-bit data provides compared to the converted 8-bit data used by all visual-based algorithms. Furthermore, CNN-based descriptors presented to be an interesting alternative to the classical descriptors from Appendix C, performing among the top contenders in the benchmark. Its efficiency was however not groundbreaking, sometimes only slightly outperforming the best classical descriptor. Machine learning solutions are additionally notorious for their lack in flexibility. Therefore, the satisfying performance obtained during the benchmark is not guaranteed in all unknown situations. The CNN-based descriptors lack the very important reliability given by the classical descriptors. The best combination, suited for real-time thermal SLAM in a smoke-filled environment proved to be SURF features described by the BRIEF binary descriptor.

6.3 Loop closure detection

The features extracted from thermal images inside a fiery environment contain sufficient distinct information to create a diverse vocabulary of visual words that can be used to detect loop closures. Using SURF features described by the BRIEF descriptor enabled to detect the different major loop closures present in the dataset without the use of the inertial data, some being over a minute apart. In contrary, the stock ORB SLAM 3 algorithm, also using thermal images, failed to do so.
The new map atlas added to ORB SLAM 3 that allows reusing lost maps has proven to be a crucial element in the operation of Firebot SLAM. Notably in TSC 2, the algorithm fails to track the map to create a new one just before the halfway point of the dataset. During a loop closure around twenty seconds later, the two maps are merged together resulting in no information loss. In the case of other state-of-the-art visual SLAM algorithms, the map would have been discarded right after the failure to track it.

6.4 Thermal SLAM

While running both Firebot SLAM and the stock ORB SLAM 3 multiple times on the same dataset, some inconsistencies in the results were observed. The majority of the time, the system runs flawlessly but in a number of cases, both algorithms would either fail to detect loop closures it detected on previous runs or fail to track the map and crash shortly after trying to merge two maps. The issue is caused by the asynchronous running of the three main threads. The loop closure thread, for

example, is activated to check the presence of a loop closure cyclically every 5 seconds in the stock ORB SLAM 3. If the tracking thread, which handles everything from the feature extraction to the odometry computation, is slowed down by a third party background process on the CPU, it may feed a different frame situated right after the loop closure to the loop closure detection thread causing the system to miss it.
Performing thermal SLAM inside a benign environment such as the Optitrack room proved to be unsuccessful. Despite the attempt to diversify the content inside the room with different materials to increase the differences in infrared emissivity, the lack of a significant heat source such as the fire and smoke inside the parking garage at the Twente Safety Campus made it difficult to obtain distinct features. Trying to track features by projection assuming constant velocity (computed to be zero here) and locally comparing descriptors, the system then always manages to find matching features that validate the supposed standstill. In addition, the ambiguous features regularly trigger false loop closure detections further validating the results computed by the system.

6.5 Thermal-inertial SLAM

Due to a number of factors, unfortunately no results of a thermal-inertial SLAM algorithm could be obtained during this thesis.
The XSens Mti-620 mounted inside the provided setup for the thesis possessed a faulty z-axis accelerometer. Due to the planning at the Twente Safety Campus, the datasets were recorded before the IMU calibrations could be performed and therefore, the issue was only discovered afterwards.
The IMU initialization problem inside ORB SLAM 3 arose to be a very complex problem that has yet to be solved. A combination of a lack of robust features in the Optitrack room together with a lack of movement around all three axes of the IMU resulted in the system being unable to find its correct orientation with respect to gravity. A potential solution to the lack of movement would be to detach the thermal-inertial system from the ground vehicle and initialize the system in a handheld manner before reattaching it to the mobile platform again. This does however not solve the lack of robust features for the required correct estimate of the initial thermal-only odometry needed to complete the initialization. Furthermore, this would also contradict the requirement that the system should function immediately after deployment without any setup time. One could also argue that the Optitrack room is not the targeted environment of operation as no fire incident was present.

7 Conclusion

Despite the notable challenges associated with thermal cameras, the first SLAM algorithm capable of creating a 3D map from thermal images including global loop closure optimization in a smoke-filled environment was developed during this thesis.
The main research question leading the work performed in this assignment was whether it is possible to utilize a stereo pair of thermal cameras potentially alongside an IMU to, in a consistent and computationally cost-effective manner, localize a UGV and create a 3D map in a smoke-filled environment during a fire incident. From the results obtained in this thesis, it can be concluded that it is indeed possible to consistently and computational cost-effectively localize a vehicle and create a 3D map from stereo thermal images in a smoke-filled environment during a fire incident. Despite that the inertial data from the Twente Safety Campus was not usable, it proved not to be essential for a satisfying operation of the system. Without the IMU, Firebot SLAM can already estimate well within the prescribed loop time of 250 milliseconds the proper trajectory taken by the robot. Furthermore, the system can also produce a consistent and coherent 3D map of the environment by successfully detecting all the present loop closures. Finally, while utilizing a stereo thermal setup instead of a monocular thermal camera, the trajectory and 3D map could be directly obtained at the correct scale without needing additional input from an IMU.
Along with the research question, a number of sub-questions were also posed to help guide the research. The first question being what the most efficient method to extract and describe features from thermal images would be. It was found that the SURF-BRIEF combination proved to be the most efficient combination to extract sufficient robust features and correctly match them. The combination offers the best balance between efficiency, i.e. the number of correctly-matched features from a given number of extracted keypoints, and computational speed for real-time operation while guaranteeing a degree of flexibility in unknown situations. Gradient-based feature tracking possesses great potential but lacks repeatability making it unsuitable for loop closure detection. CNN-based descriptors performed as one of the top descriptors but its performance does not justify its lack of the critically-needed flexibility.
The second sub-question asked whether the robust features could be detected in real-time. Between the three TSC datasets, SURF-BRIEF has a mean processing time of 135 milliseconds per frame, or 7.4 frames per second, to extract and correctly match sufficient robust features. This leaves plenty of time for the odometry, mapping and optimization to still be performed within the desired 250 milliseconds loop time. Note that the feature extraction and description is performed in series during the benchmark whereas it is executed in parallel on separate threads in Firebot SLAM. This further decreases the actual time needed to complete this task. The robust features can thus be detected in real-time.
Furthermore, it was asked if the obtained thermal features from the chosen combination can be used to detect loop closures. As both the loop closure candidate experiments and the Firebot SLAM results showed, the SURF blob features described by the BRIEF binary descriptor can be used to create a bag-of-visual-words vocabulary that is rich enough to detect loop closures in a smoke-filled environment. Even if the loop closures are separated for up to two minutes in time, Firebot SLAM was able to successfully perform a global bundle adjustment to provide a correct trajectory.
The accuracy of the proposed algorithm was also questioned in the sub-questions. Although no ground truth was available at the Twente Safety Campus, the obtained odometry was visually-consistent with the driven trajectory. The odometry provides sufficient precision for the operator to reverse an already-driven path while avoiding potential obstacles, as was demonstrated in the results.
Finally, the last sub-question asked whether the proposed system could create a 3D map that provides sufficient situational awareness in a smoke-filled environment. It was shown that the obtained 3D map has enough detail to include the location of the different obstacles present in the building. Given the current sparse point cloud, detail lacked to identify most objects however. Due to the more uniform distribution of SURF features compared to ORB, the 3D map obtained by Firebot SLAM is more meaningful with respect to the 3D map provided by the stock ORB SLAM 3. Overall, the 3D map created by Firebot SLAM increases the situational awareness for both the operator of the vehicle to avoid obstacles and for mission planning.

The smoke present inside the environment did not seem to cause any significant averse effects on the odometry estimation. It could even be argued that it proved to be beneficial to the successful operation of Firebot SLAM. It was namely observed that the heat provided by both the fire source itself and by the emitted smoke increased the differences in measured infrared emissivity resulting in more contrast in the thermal images. It can be concluded that performing thermal SLAM in a smoke-filled building in the presence of fire is equivalent to performing visual SLAM at night with a giant flood light turned on.

Although the acquisitions at the Twente Safety Campus were performed using the on-board Intel NUC i7, the SLAM process itself was tested offline on the development laptop. ORB SLAM 3 uses three main threads for the camera tracking, mapping and loop closure respectively. A fourth thread is occasionally activated to perform the bundle adjustment. The four available threads of the Intel NUC i7 should therefore be plenty to execute the SLAM algorithm on-board, also keeping in mind that the clocking speed of the NUC is double the one of the development laptop. During real-time deployment, additional tasks will be added during operation which were done beforehand in the previous experiments. These include collecting the data from the cameras and IMU, dynamically converting the images from 16-bit to 8-bit and rectifying them. The tasks are all minor but important steps and should therefore also be considered when dividing the tasks over the threads. In case bottlenecks occur in the threads, Intel NUC i7 models are also available with eight threads.

Driving close to a fire source pushes the hardware in the system to its limits by operating near the upper bounds of the rated operating temperature range. Especially the IMU error parameters such as the bias and scale factor are very sensitive to temperature changes. Determining these in a calibration beforehand does therefore not necessarily guarantee a better performance during operation. The on-board thermometer and error compensation of the XSens MTi-600 series IMU sensors may be better at mitigating these errors as the temperature increases, even if a small rest error remains. The intrinsic parameters obtained from the camera calibrations will also slightly deviate in practice as the change in environmental conditions, such as heat, humidity and atmospheric pressure, slightly deform the lenses.

8 Future work

Further developments on this project can be made with respect to feature detection and matching in more benign environments, the IMU integration and the real-time deployment during a fire incident. In a more realistic scenario, the system is to be deployed in larger buildings such as indoor parking garages. It will then be likely that the UGV does not drive near a fire source for the majority of its trajectory. A situation closer to the one at Saxion's Optitrack room may then occur. A more robust form of feature detection and matching in areas of lower infrared radiation would therefore be desired. In combination with inertial data, utilizing the 16-bit radiometric thermal data could provide the needed additional reliability. Investigating the effect of smoke on the infrared radiation with respect to the distance from the fire source could also provide interesting insights on this decrease of feature robustness.
Gradient-based feature tracking in 16-bit could potentially offer this additional robustness. Alternative methods of loop closure detections such as direct methods should then be researched to replace feature-based bag-of-visual-words loop closure detections. Another option to increase robustness would be to further develop CNN-based descriptors. As of now, the CNN algorithm is trained as a classification problem, trained to distinguish matches from non-matches. The recall of CNN-based descriptors can be increased by training the model as a regression problem by taking three parallel siamese networks, the query, a matching feature and a non-matching feature. The model is then optimized to maximize the distance between the L2-norm of the matching and non-matching pair instead of maximizing the number of correctly-classified pairs. This can increase the matching performance by providing more distinctive descriptors.
One of the causes of the IMU's failure to initialize is the lack of movement around all its axes needed with the current initialization method. As the main targeted platform of Firebot SLAM is a ground robot, research into IMU initialization techniques more suited for these types of vehicles should be performed.
Furthermore, the smoke that was generated during the Twente Safety Campus acquisitions were not completely representative of smoke from a burning car, the latter having much larger particles. Further investigation is therefore needed on the visibility of thermal cameras in smoke from burning composites and other materials commonly found in cars. As these smoke particles are larger, it is expected to play a more significant role. At the start of the fire, the smoke will stay towards the ceiling and the upper part of the thermal images will be rendered useless for SLAM operations. As time progresses, smoke will cover increasingly more of the frame. The effectiveness of thermal vision in thicker smoke should especially be experimented during this time.
Currently all captured frames are processed by the SLAM algorithm, regardless of the processing time in relation with the sampling interval of the thermal images. The impact of real-time operation, i.e. reducing the number of frames, on both the quality of the estimated odometry and the details in the map should be examined. Given the slow speed of the vehicle, it is however not expected to become a significant problem. The effects of operating in a high temperature environment on the hardware and the computational performance of the Intel NUC i7 should also be investigated.

References

[1] Howe and Howe. Thermite: First Commercial Firefighting Robot Sold In The U.S. https://www.howeandhowe.com/civil/thermite, 2021.

[2] V. Highfield. Meet TAF20, The Turbine-Aided Firefighting Robot Of The Future. https://www.alphr.com/the-future/1002221/meet-taf20-the-turbine-aided-firefighting-robot-of-the-future/, 2015.

[3] ROBOTNOR. Anna Konda - The fire fighting snake robot. https://robotnor.no/research/anna-konda-the-fire-fighting-snake-robot/, 2012.

[4] L. Plaugic. The US Navy's robot SAFFiR is getting closer to fighting fires at sea. https://www.theverge.com/2015/2/5/7986345/firefighting-robot-saffir-prototype-us-navy, 2015.

[5] Tecdron. Tecdron Scarab TX. https://www.fireproductsearch.com/scarab-tx-firefighting-robot/, 2017.

[6] Omroep West. Nieuwe blusrobot van Brandweer Haaglanden: 'Inzetbaar waar het voor de brandweer niet veilig is'. https://www.omroepwest.nl/nieuws/4000166/Nieuwe-blusrobot-van-Brandweer-Haaglanden-Inzetbaar-waar-het-voor-de-brandweer-niet-veili 2020.

[7] M. Charlton, M. Sims, T. Coats, J.P. Thompsom. The microcirculation and its measurement in sepsis. *Journal of the Intensive Care Society*, 18(3), 2016. doi: 10.1177/1751143716678638.

[8] V. Sluiter. Disparity Calculator. https://saxionmechatronics.github.io/Firebot-DisparityCalculator/disparity.html, 2020.

[9] Y. Liu, X. Xu, F. Li. Image Feature Matching Based on Deep Learning. *IEEE 4th International Conference on Computer and Communications*, 2018.

[10] T. Mouats, N. Aouf, L. Chermak, M. A. Richardson. Thermal Stereo Odometry for UAVs. *Sensors*, 5(11), 2015.

[11] S. Khattak, F. Mascarich, T. Dang, C. Papachristos, and K. Alexis. Robust Thermal-Inertial Localization for Aerial Robots: A Case for Direct Methods. *International Conference on Unmanned Aircraft Systems (ICUAS)*, June 2019.

[12] M. R. U. Saputra, P. B. de Gusmao, C. X. Lu, Y. Almalioglu et. al. DeepTIO: A Deep Thermal-Inertial Odometry with Visual Hallucination. *IEEE Robotics and Automation Letters*, 2020. doi: 10.1109/LRA.2020.2969170.

[13] J. Johansson, M. Solli, A. Maki. An Evaluation of Local Feature Detectors and Descriptors for Infrared Images. *Computer Vision – ECCV 2016 Workshops, Amsterdam, The Netherlands, October 8-10 and 15-16, Part III*, pages 711–723, November 2016. doi: 10.1007/978-3-319-49409-8_59.

[14] C. Campos, R. Elvira, J.J. Gomez Rodriguez, J.M.M. Montiel, J.D. Tardos. ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. July 2020.

[15] A. A. Gawad, H. A. Ghulman. Prediction of Smoke Propagation in a Big Multi-Story Building Using Fire Dynamics Simulator (FDS). July 2015. doi: 10.11648/j.ajee.s.2015030401.12.

[16] E. Rosten, T. Drummond. Machine Learning for High-Speed Corner Detection. *Computer Vision - ECCV 2006 Part 1*, May 2006.

[17] D. Tyagi. Introduction to ORB (Oriented FAST and Rotated BRIEF). https://medium.com/data-breach/introduction-to-orb-oriented-fast-and-rotated-brief-4220e8ec40cf, 2019.

[18] D. G. Lowe. Distinctive Image Features from Scale-Invariant Keypoints. 2004.

[19] S. Leutenegger, M. Chli, R. Siegwart. BRISK: Binary Robust invariant scalable keypoints. *IEEE International Conference on Computer Vision*, November 2011.

[20] A. Alahi, R. Ortiz, P. Vandergheynst. FREAK: Fast retina keypoint. *CVPR, IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, June 2012. doi: 10.1109/CVPR.2012.6247715.

[21] Brandweeracademie. Brand in parkeergarages: Risicoafwegingen in relatie tot brandveiligheidsvoorzieningen (Dutch, Fire inside parking garages: Risk trade-offs in relation to fire safety equipment). Technical report, Instituut Fysieke Veiligheid, 2013.

[22] N. Tilley, B. Merci. Relation Between Horizontal Ventilation Velocity and Backlayering Distance in Large Closed Car Parks. *Fire Safety Science*, 9:777–787, 2009. doi: 10.3801/IAFSS.FSS.9-777.

[23] P.H.E. van der Leur. Onderzoek richtlijnbrandveiligheid parkeergarages (Dutch, Research fire safety guidelines parking garages). Technical Report F.2013.0591.01.R002, DGMR BV, November 2015.

[24] Y.J. van Straalen. Risico's in parkeergarages ten gevolge van elektrisch en waterstof aangedreven personenauto's - Internationale Inventarisatie (Dutch, Risks in parking garages caused by electric and hydrogen-driven passenger cars - International Assessment). Technical Report TNO 2020 R10134, TNO, February 2020.

[25] P. Liu, H. Yu, S.Cang, L. Vladareanu. Robot-Assisted Smart Firefighting and Interdisciplinary Perspectives. *Proceedings of the 22nd International Conference on Automation and Computing*, 2016.

[26] J.-H. Kim , S. Jo, B. Y. Lattimer. Feature Selection for Intelligent Firefighting Robot Classification of Fire, Smoke, and Thermal Reflections Using Thermal Infrared Images. *Journal of Sensors*, August 2016.

[27] J. W. Starr, B. Y. Lattimer. A Comparison of IR Stereo Vision and LIDAR for use in Fire Environments. *SENSORS*, 2012.

[28] D. Scaramuzza, F. Fraundorfer. Visual Odometry - Part I: The First 30 Years and Fundamentals. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 18(4):80–92, December 2011. doi: 10.1109/MRA.2011.943233.

[29] J.S. Hu, C.Y. Tseng, M.Y. Chen, K.C. Sun. IMU-Assisted Monocular Visual Odometry Including the Human Walking Model for Wearable Applications. *IEEE International Conference on Robotics and Automation (ICRA)*, 2013.

[30] X. Ma, Z. Geng, Z. Bie. Depth Estimation from Single Image Using CNN-Residual Network. *Standford CS231n: Convolutional Neural Networks for Visual Recognition*, 2017.

[31] Y. Liu, Z. Chen, W. Zheng, H. Wang , J. Liu. Monocular Visual-Inertial SLAM: Continuous Pre-integration and Reliable Initialization. *Sensors*, 17(11), November 2017. doi: https://doi.org/10.3390/s17112613.

[32] S. Prabu, G. Hu. Stereo Vision based Localization of a Robot using Partial Depth Estimation and Particle Filter. *Proceedings of the 19th World Congress - The International Federation of Automatic Control*, 2014.

[33] D.M. Escriva, R. Laganiere. *OpenCV 4 Computer Vision Application Programming Cookbook*. Packt, 4th edition, 2019.

[34] D. Scaramuzza, F. Fraundorfer. Visual Odometry - Part II: Matching, Robustness, and Applications. *IEEE ROBOTICS AND AUTOMATION MAGAZINE*, 19(2):78–90, February 2012. doi: https://doi.org/10.1109/MRA.2012.2182810.

[35] P. Corke. *Robotics, Vision and Control*, volume 73 of *Springer Tracks in Advanced Robotics (STAR)*. Springer, 2011. ISBN 978-3-642-20143-1.

[36] F. Zheng, S. Li, E. Sun, L. Zhao. Algorithms Analysis of Mobile Robot SLAM based on Kalm and Particle Filter. *9th International Conference on Modelling, Identification and Control*, 2017.

[37] N.M. Yatim, N. Buniyamin. Particle filter in simultaneous localization and mapping (Slam) using differential drive mobile robot. *Jurnal Teknologi*, 77(20), December 2015. doi: 10.11113/jt.v77.6557.

[38] G. Grisetti, R. Kümmerle, C. Stachniss, W. Burgard. A Tutorial on Graph-Based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43, 2010. doi: 10.1109/MITS.2010.939925.

[39] S. Lucey. Direct Visual SLAM. *16-623 - Designing Computer Vision Apps*, 2016.

[40] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, P. Furgale. Keyframe-based visual–inertial odometry using nonlinear optimization. *The International Journal of Robotics Research*, 34(3): 314–334, 2015. doi: 10.1177/0278364914554813.

[41] M. Bloesch, S. Omari, M. Hutter, R. Siegwart. Robust Visual Inertial Odometry Using a Direct EKF-Based Approach. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, September 2015. doi: 10.1109/IROS.2015.7353389.

[42] V. Usenko, J. Engel, J. Stuckler, D. Cremers. Direct visual-inertial odometry with stereo cameras. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, June 2016. doi: 10.1109/ICRA.2016.7487335.

[43] T. Qin, S. Cao, J. Pan, S. Shen. A General Optimization-based Framework for Global Pose Estimation with Multiple Sensors. January 2019.

[44] J. Engel, J. Stuckler, D. Cremers. Large-Scale Direct SLAM with Stereo Cameras. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, December 2015. doi: 10.1109/IROS.2015.7353631.

[45] A.Geiger, P. Lenz, R Urtasun. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[46] P. V. K. Borges, S. Vidas. Practical Infrared Visual Odometry. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS*, 18(7), August 2016.

[47] S. Khattak, C. Papachristos, and K. Alexis. Keyframe-based Direct Thermal-Inertial Odometry. *International Conference on Robotics and Automation 2019*, May 2019. doi: 10.1109/ICRA.2019.8793927.

[48] P. Fischer, A. Dosovitskiy, E. Ilg, P. Hausser, C. Hazırbas¸, V. Golkov. FlowNet: Learning Optical Flow with Convolutional Networks. *2015 IEEE International Conference on Computer Vision (ICCV)*, 2016. doi: 10.1109/ICCV.2015.316.

[49] T. Mouats, N. Aouf, D. Nam, S. Vidas. Performance Evaluation of Feature Detectors and Descriptors Beyond the Visible. *Journal of Intelligent & Robotic Systems*, 92:33–63, 2018.

[50] G. Nannen, G. Oliver. Optimal Number of Image Keypoints for Real Time Visual Odometry. *IFAC Proceedings Volumes*, 45(5):1–6, 2012. doi: 10.3182/20120410-3-PT-4028.00055.

[51] P. Mihelich, J. Bowman. ROS. image_pipeline: camera_calibration. `http://wiki.ros.org/camera_calibration`, 2020.

[52] FLIR Systems, Inc. Compact LWIR -Thermal Camera Core - Boson - Model: Boson 640, 95° (HFOV) 4,9 mm. https://www.flir.eu/products/boson/?model=20640A095, 2020.

[53] A. Kaehler, G. Bradski. *Learning OpenCV3: Computer vision in C++ with the OpenCV library.* O'Reilly, 2017.

[54] M. Vagner. MEMS Gyroscope Performance Comparison Using Allan Variance Method. 2011.

[55] M. Fleps, E. Mair, O. Ruepp et. al. Optimization Based IMU Camera Calibration. *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011. doi: 10.1109/IROS.2011.6095067.

[56] V. Arcidiacono. Syncing: Time series synchronization library. https://syncing.readthedocs.io/en/stable/, 2021.

[57] D.W. Eggert, A. Lorusso, R.B. Fisher. Estimating 3-D rigid body transformations: a comparison of four major algorithms. *Machine Vision and Applications*, 9:272–290, 1997.

[58] Y. Liu, X. Xu, F. Li. Image Feature Matching Based on Deep Learning. *4th International Conference on Computer and Communications*, 2018. doi: 978-1-5386-8339-2.

[59] Soonmin Hwang and Jaesik Park and Namil Kim and Yukyung Choi and In So Kweon. Multispectral Pedestrian Detection: Benchmark Dataset and Baselines. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[60] D. P. Kingma, J. L. Ba. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. *ICLR 2015*, 2015.

[61] S. J. Reddi, S. Kale, S Kumar. ON THE CONVERGENCE OF ADAM AND BEYOND. *ICLR 2018*, 2018.

[62] R. Hadsell, S. Chopra, Y. LeCun. Dimensionality Reduction by Learning an Invariant Mapping. *CVPR 2006*, 2005.

[63] H. Nguyen, S. Lee. An Optimal Feature Selection based on Orthogonality Index for Stereo Visual Odometry. *IEEE Access*, 99, May 2019. doi: 10.1109/ACCESS.2019.2916190.

[64] K. Pearson. On the criterion that a given system of deviations from the probable in the case of a correlated system of variables is such that it can be reasonably supposed to have arisen from random sampling. *Philosophical Magazine, Issue 5*, 50(302):157–175, 1900. doi: 10.1080/14786440009463897.

[65] D. Gálvez-López, J. D. Tardós. Bags of Binary Words for Fast Place Recognition in Image Sequences. *IEEE Transactions on Robotics*, 28(5):1188–1197, October 2012. ISSN 1552-3098. doi: 10.1109/TRO.2012.2197158.

[66] P. Furgale, J. Rehder, R. Siegwart. Unified Temporal and Spatial Calibration for Multi-Sensor Systems. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Tokyo, Japan*, 2013.

[67] H. Ovren, P.-E. Forssen. Gyroscope-based Video Stabilisation With Auto-Calibration. *IEEE International Conference on Robotics and Automation (ICRA)*, 2015. doi: 10.1109/ICRA.2015.7139474.

[68] T. Kiurski. *Understanding Flashover.* Fire Engineering, 2010. https://www.fireengineering.com/2010/06/22/278458/kiurski-flashover/.

[69] E. Rublee et. al. ORB: an efficient alternative to SIFT or SURF. *Proceedings / IEEE International Conference on Computer Vision. IEEE International Conference on Computer Vision*, 2011. doi: 10.1109/ICCV.2011.6126544.

[70] J. Shi, C. Tomasi. Good Feature To Track. *IEEE Conference on Computer Vision and Pattern Recognition 1994 (CVPR94)*, June 1994.

[71] C. Harris, M. Stephens. A Combined Corner and Edge Detector. In *Alvey Vision Conference*, 1988.

[72] H. Bay, A. Ess, T Tuytelaars, Luc van Gool. Speeded-Up Robust Features (SURF). *Computer Vision and Image Understanding*, 10(3):346–359, June 2008.

[73] M. Calonder, V. Lepetit, C. Strecha, P. Fua. BRIEF: Binary Robust Independent Elementary Features. *Computer Vision - ECCV 2010, 11th European Conference on Computer Vision, Heraklion, Crete, Greece, September 5-11, 2010, Proceedings, Part IV*, September 2010. doi: 10.1007/978-3-642-15561-1_56.

[74] Photo Counter. Sensor Size Table. `https://www.photocounter.com.au/wp-content/uploads/2013/01/sensor-size-table.pdf`, 2013.

[75] N. Mansurov. What is crop factor? `https://photographylife.com/what-is-crop-factor`, 2020.

[76] M. Freeman. *The Photographer's Mind: Creative Thinking for Better Digital Photos*. Focal Press, 2010.

[77] S. Stafford, R. Hillebrand, H.J. Hauschild. *The New Nikon Compendium: Cameras, Lenses & Accessories Since 1917*. Lark Books, 2004.

A Smoke Propagation Inside an Enclosed Room

Although thermal cameras offer much better visibility inside smoke-filled environments compared to visible-light cameras, their performance is not completely unaffected by the presence of airborne particles. The quality of the thermal image is mostly dictated by the type of smoke, i.e. the size of the particles, and the smoke density. The likelihood of a particle reflecting an infrared ray namely grows as these smoke properties increase, which can clutter the thermal image.
Therefore, to better understand the environmental conditions the system will experience, some research was done into smoke propagation inside enclosed buildings.

A.A. Gawad et. al. [15] utilized computer simulations to investigate the propagation of smoke inside a multi-story building under different circumstances related to door openings and ventilation.
Using Computational Fluid Dynamics (CFD), the movement of smoke can be simulated using the four Navier-Stokes equations, being the conservation of mass (eq. (54)), the conservation of momentum (eq. (55)), the conservation of energy (eq. (56)) and the equation of state for a perfect gas (eq. (57)) [15]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho u_i = 0 \tag{54}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \nabla \cdot \rho u_i u_i + \nabla p = \rho f + \nabla \cdot \tau_{ij} \tag{55}$$

$$\frac{\partial}{\partial t}(\rho h) + \nabla \cdot \rho h u_i = \frac{Dp}{Dt} + \dot{q}''' - \nabla \cdot q + \Phi \tag{56}$$

$$p = \rho R T \tag{57}$$

where ρ is the density, u_i the particle velocity in direction i, f the sum of external forces, τ_{ij} the shear force, h the enthalpy, $\dot{q}'''$ the heat release rate per unit volume, q the heat transfer, Φ any heat source, R the gas constant and T the temperature.
A.F.A Gawal et. al. [15] predicted the smoke propagation in a multi-story building using the Fire Dynamics Simulator (FDS), which utilizes the mixture fraction model from eqs. (58)-(60) as a combustion model:

$$Z = \frac{sY_F - (Y_{O_2} - Y_{O_2}^{\infty})}{sY_F^I + Y_{O_2}^{\infty}} \tag{58}$$

$$s = \frac{\nu_{O_2} W_{O_2}}{\nu_F W_F} \tag{59}$$

$$\nu_F = 1 \tag{60}$$

where Z is the mixture fraction varying from $Z = 1$ at the fire source to $Z = 0$ in regions fare away from the fire containing only ambient air with undepleted oxygen. Further, Y is the mass fraction, Y_F^I is the mass fraction in the fuel stream, subscripts F and O_2 correspond respectively to the fuel and oxygen, the superscript ∞ means that it is far away from the fire, ν is the stoichiometric coefficient and W is the molecular weight.
The model used was a three-story building with an overall size of $41 \times 17 \times 10$ meters. For each floor, a total of 19 simulations were performed under the different conditions presented in table 16.
In figure A.1, the simulation results at 60 seconds, 300 seconds and at the final period when the smoke has reached steady-state is shown for the different cases on the second story. Because smoke is less dense than the ambient air, it has a tendency to travel upwards. This can be noticed in the results after 60 seconds where, for all cases, the smoke is situated at the ceiling of the second floor and is travelling towards the third floor via the ventilation and the stair case when possible.

No.	First Floor Cases	Second Floor Cases	Third Floor Cases	Main door	First floor stair door	Second floor stair door	Third floor stair door	Emergency vent
1	*F1*	*S1*	*T1*	Open	Open	Open	Open	Closed
2	*F2*	*S2*	*T2*	Open	Closed	Closed	Closed	Closed
3	*F3*	*S3*	*T3*	Open	Closed	Closed	Open	Closed
4	*F4*	*S4*	*T4*	Open	Closed	Open	Closed	Closed
5	*F5*	*S5*	*T5*	Open	Open	Closed	Closed	Closed
6	*F6*	*S6*	*T6*	Open	Open	Open	Closed	Closed
7	*F7*	*S7*	*T7*	Open	Open	Closed	Open	Closed
8	*F8*	*S8*	*T8*	Open	Closed	Open	Open	Closed
9	*F9*	*S9*	*T9*	Closed	Open	Open	Open	Closed
10	*F10*	*S10*	*T10*	Closed	Closed	Closed	Closed	Closed
11	*F11*	*S11*	*T11*	Closed	Closed	Closed	Open	Closed
12	*F12*	*S12*	*T12*	Closed	Closed	Open	Closed	Closed
13	*F13*	*S13*	*T13*	Closed	Open	Closed	Closed	Closed
14	*F1v*	*S1v*	*T1v*	Open	Open	Open	Open	Open (0 m/s)
15	*F2v*	*S2v*	*T2v*	Open	Closed	Closed	Closed	Open (0 m/s)
16	*F3v*	*S3v*	*T3v*	Open	Open	Open	Open	Open (1 m/s)
17	*F4v*	*S4v*	*T4v*	Open	Closed	Closed	Closed	Open (1 m/s)
18	*F5v*	*S5v*	*T5v*	Open	Open	Open	Open	Open (5 m/s)
19	*F6v*	*S6v*	*T6v*	Open	Closed	Closed	Closed	Open (5 m/s)

Table 16: Cases tested for the smoke propagation by A.F.A Gawal et. al. (the case letter indicates the floor where the fire source was located) [15]

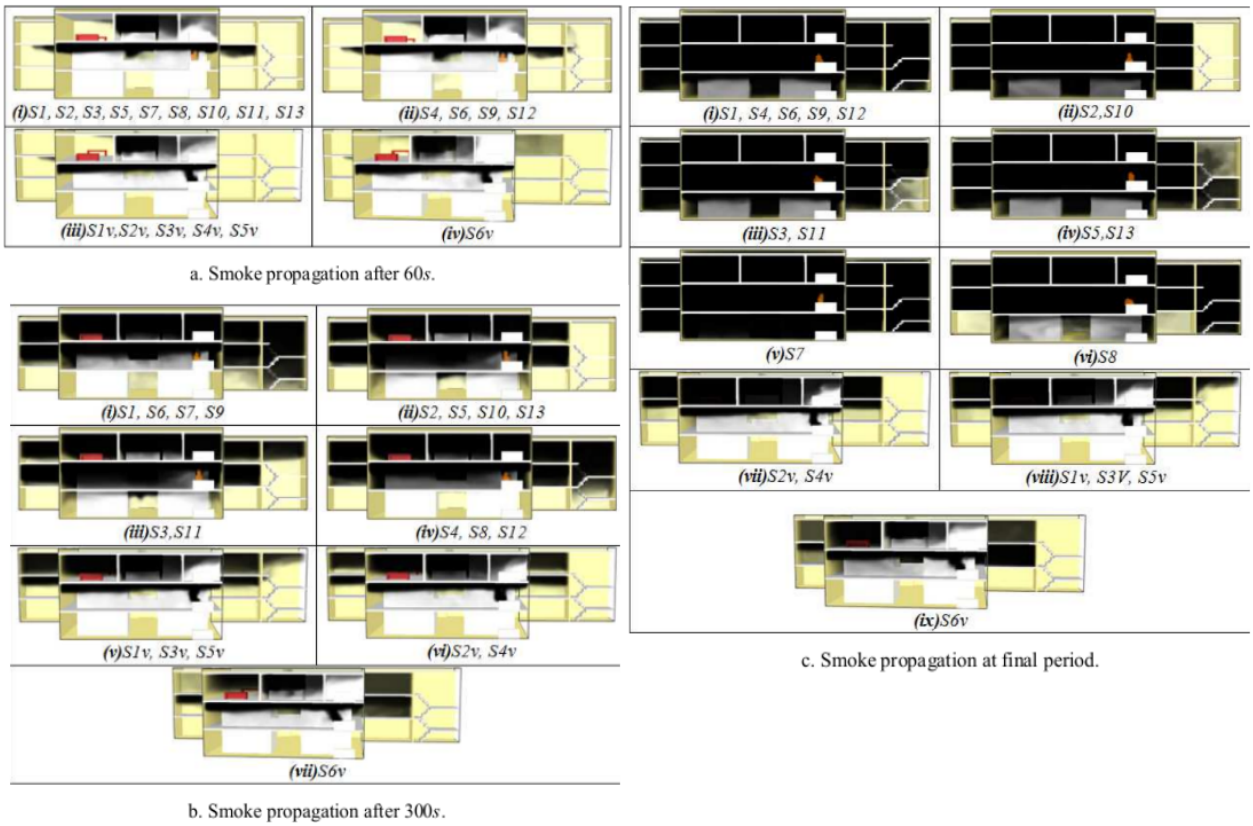


Figure A.1: Simulation results after 60 seconds (a), 300 seconds (b) and a the final period (c) [15]

From the simulation results after five minutes (300 seconds), it can be observed that the smoke will take the path of least resistance. For the cases where one or more doors to the staircase are closed, the particles fill the second and third floor at a faster rate, causing the smoke to enter the first floor at an earlier time. With the fire source on the second floor, smoke has completely taken over the second and third floor in all cases without the emergency vent running when it has reached the final period. The only case that resulted in smoke filling the entire building is in case S7 when the doors to the staircase on the third and first floor are open. Smoke is propagated the least when the emergency vent is running. Then, the particles are mostly concentrated on the third floor and at the ceiling of

the second floor.

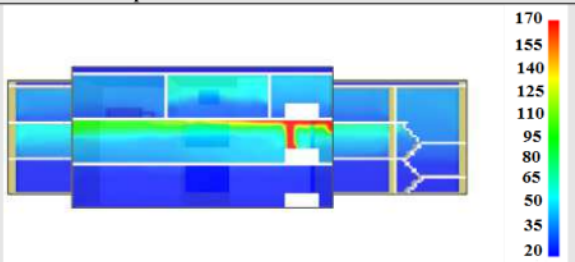
Case	Final time	Maximum Temperature = 170°C
S1v, S3V, S5v, S2v, S4v, S6v	1200*s*	
S7	1300*s*	
S1, S4, S6, S9, S12, S2,S10, S3, S11, S5,S13, S8	1800*s*	

Figure A.2: Final period and maximum temperature for the second floor cases [15]

In figure A.2, the time elapsed until steady-state for each case on the second floor is shown as well as the maximum temperature. In addition to the significantly lower amount of smoke, the final period was reached much faster, after 20 minutes, when the emergency vent is in use. This can mainly be explained by the presence of a steady outgoing flow exiting the building at the top. On the other hand, the researchers have noted that an outgoing flow at the first floor, i.e. the main entrance, has a negligible effect on the amount of smoke when the fire source is on a higher floor. The highest temperature of 170°C was reached at the fire source. A temperature increase, compared to the ambient temperature of 20°C was mostly observed on the second floor and around the ventilation shaft on the third floor whereas the temperature on the first floor remained unchanged. This would suggest that, when smoke is present on the first floor, it is cooled-down smoke that has been pushed down by hotter smoke filling the higher floors.

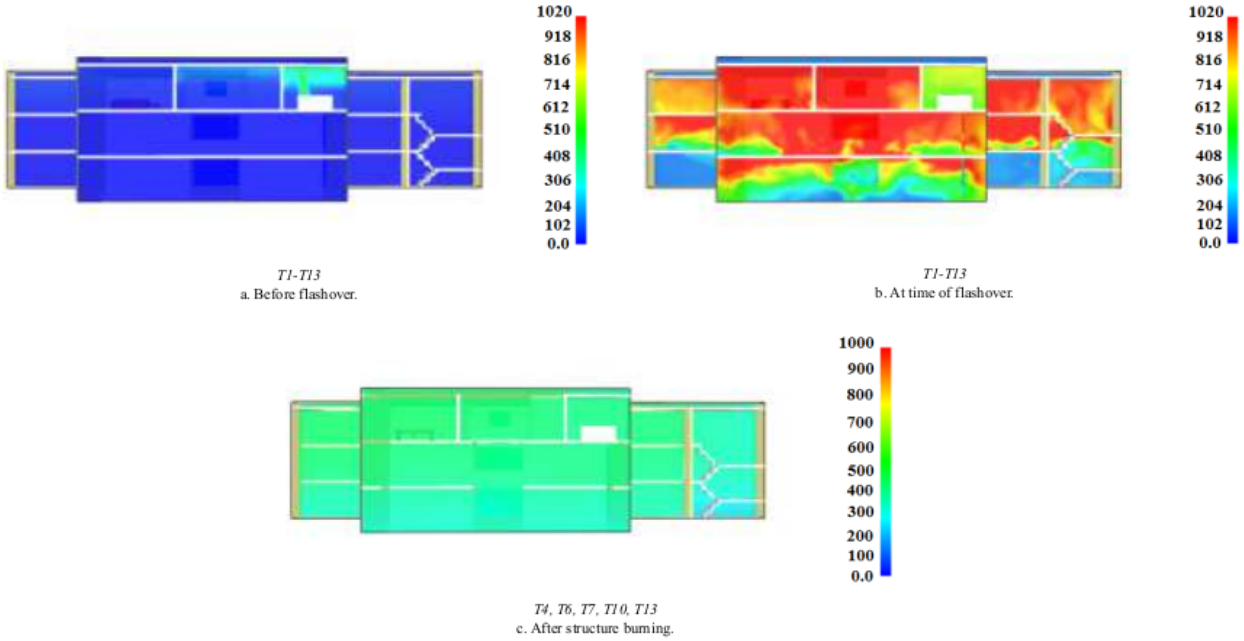


Figure A.3: Temperature before (a), during (b) and after (c) a flashover during the simulations of T1-T13 [15]

During a fire in a confined space, the rapid increase in temperature in combination with the presence of a large amount of flammable materials (solids and gases) can cause a sudden ignition of all the combustibles near the source. Such a phenomenon is called a flashover [68] and was observed during certain simulations when the fire source was on the third floor. A flashover occurred in the cases which resulted in a really high smoke concentration near the fire source. Due to the nature of propagation of smoke, the third floor possesses the highest concentration out of all floors which, in combination with a fire source creates an optimal environment for such an event. Figure A.3 shows the temperature

distribution inside the building before, during and after the flashover. It can be noticed that a flashover occurred for all cases where the emergency roof vent is closed. Between the 13 cases, the flashover happened between 20 and 23 minutes after the start of the simulation. Before the flashover, the temperature around the fire source is at around $300°C$, which is nearly three times the maximum temperature observed in cases where no flashover was observed. Additionnally, the authors note that small sporadic flashes of flames, named rollover, were also signs that a flashover was forming. During the flashover, the heat instantly spreads through the second and third floor where the temperature exceeds $900°C$, demonstrating the danger of this phenomenon. Even after the flashover, the building remains at higher temperatures between $300°C$ and $700°C$.
As a conclusion, the researchers mention that the location of the fire source greatly affects the smoke propagation through the building and that stair doors have shown to inhibit smoke from travelling to lower floor. As smoke travels upwards, an open emergency vent situated on the roof of the top floor, even with the fan turned off, will at least prolong the visibility on the underlying floors. Good ventilation will also prevent the formation of a flashover. Finally, openings at the bottom of the building such as a main entrance did not seem to have any influence on the smoke diffusion.

B Common feature extraction methods

B.1 FAST

The Features from Accelerated Segment Test (FAST) by E. Rosten et. al.[16] detects corners in an image by performing a series of intensity tests of 16 pixels located on a circle of radius 3 around a point of interest, as depicted in figure B.1. In a first step, the intensity of the point of interest I_p is compared with pixels 1, 5, 9 and 13 of the circle. If the intensity of three of these four pixels are either all above $I_p + t$ or all below $I_p - t$ where t is a given threshold, then the algorithm proceeds to compare the remaining pixels in the circle.

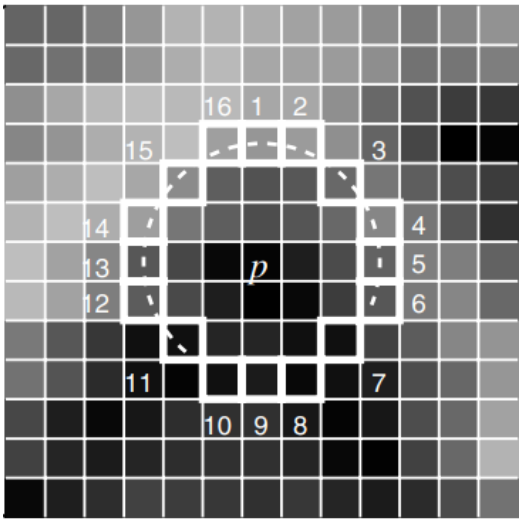


Figure B.1: FAST Bresenham circle [16]

A point is then considered a corner if at least 12 contiguous pixels in the circle out of the 16 are again either all above $I_p + t$ or all below $I_p - t$.
This method can result in a number of close pixels being qualified as a corner. Therefore non-maximum suppression is performed on a set of neighboring pixels by computing the sum of difference between the point of interest and the sixteen pixels on the circle. Finally, the point of interest in the area with the largest sum of difference is kept as a corner feature.

Although FAST has proven to be significantly faster in corner detection compared to previous algorithms such as Harris corner detection, the authors note that its method also makes it more susceptible to image noise. Additionally, FAST can falsely detect one-pixel wide lines as corners if they are a certain angle.

B.2 ORB

The Oriented FAST and Rotated Brief (ORB) feature detector by E. Rublee et. al. [69] is a corner feature extractor that further builds on FAST.
ORB provides additional robustness against scale differences between images by extracting FAST corners at different level of an image pyramid, as seen in figure B.2. Additionally, ORB computes for each detected corner its orientation based on the intensity centroid of a path centered around its location. The orientation angle θ is computed as:

$$\theta = atan2\left(m_{01}, m_{10}\right) \tag{61}$$

where m_{pq} are the patch's moments defined as:

$$m_{pq} = \sum_{x,y} x^p y^q I(x,y) \tag{62}$$

where $I(x, y)$ is the image intensity at location (x, y).

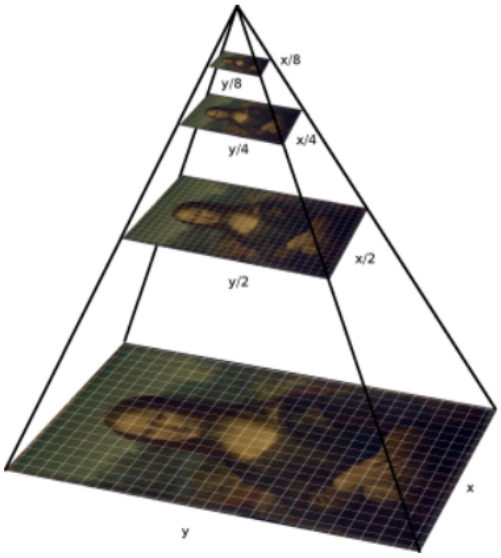


Figure B.2: Image pyramid for scale invariance in ORB [17]

B.3 Good Features To Track

Good Features to Track (GFTT), also known as Shi-Tomasi corner detection, by J. Shi and C. Tomasi [70] is an improved version of the Harris corner detection algorithm by C. Harris et. al. [71].
Both methods start by computing the change $E(u, v)$ between a window centered around (x, y) and its shifted position $(x + u, y + v)$:

$$E(u, v) = \sum_{(x,y)} w(x, y) \left[I_x u + I_y v+\right]^2 \tag{63}$$

where (u, v) is the translation of the shifted window, I_x and I_y are respectively the image derivative in x- and y-direction and $w(x, y)$ is a Gaussian weight function.
Eq. 63 can then be approximated by rewriting it in matrix form as follows:

$$E(u, v) \approx [u \quad v] M \begin{bmatrix} x \\ y \end{bmatrix} \tag{64}$$

where M is the symmetric matrix:

$$M = w(x, y) \begin{bmatrix} \sum\limits_{(x,y)} I_x^2 & \sum\limits_{(x,y)} I_x I_y \\ \sum\limits_{(x,y)} I_x I_y & \sum\limits_{(x,y)} I_y^2 \end{bmatrix} \tag{65}$$

Finally, the Shi-Tomasi corner response R is obtained by:

$$R = min(\lambda_1, \lambda_2) \tag{66}$$

where λ_1, λ_2 are the two eigenvalues of M.
A point is then taken as a corner if R is above a pre-defined threshold λ.

B.4 SIFT

The Scale-Invariant Feature Transform (SIFT) by D. G. Lowe [18] is a blob feature extraction algorithm that offers scale invariance by extracting features across different image sizes and scales. For each octave (image size), different levels of Gaussian blurring are applied to the image (scales) and the Difference of Gaussians (DoG) is consequently computed between two consecutive scales as can been seen in figure B.3.

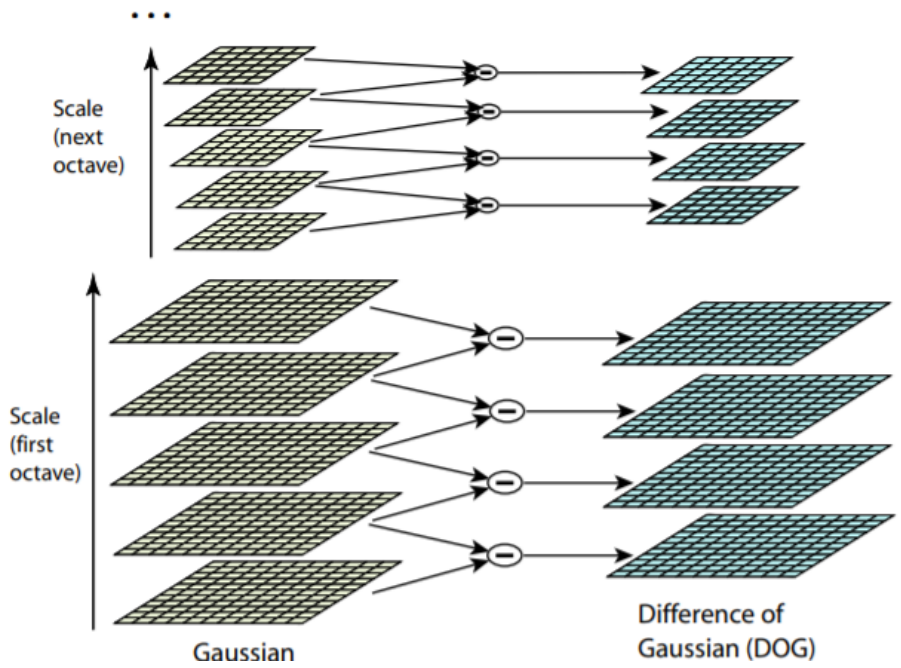


Figure B.3: Computing the Difference of Gaussians in SIFT [18]

In the next step, candidate features are generated by local extrema. Each point is compared with its eight neighboring pixels as well as with its locations and their neighbors in the previous and next scale. If its DoG-value is higher than all twenty-six neighbors, the point is considered as a potential blob feature.
Further filtering of the candidates is then performed by first eliminating features located in flat areas, i.e. points whose absolute DoG-value $|D| < 0.03$. Next, edge locations are removed by rejecting points that satisfy the following equation:

$$\frac{tr(H)^2}{det(H)} < \frac{(r+1)^2}{r} \tag{67}$$

where r is a threshold set in the experiments by D. G. Lowe to 10 and H is the Hessian matrix comprised of the second-order derivatives of the DoG D:

$$H = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix} \tag{68}$$

Finally, the keypoint's orientation is computed by taking a patch around its location and computing its gradient. A histogram is then created from the gradient orientation of each pixel inside the path, where each orientation is weighted using the gradient amplitude. The keypoint's orientation is then determined by the histogram peak inside the path.

B.5 SURF

Speed-up Robust Features (SURF) by H. Bay et. al. [72] is blob feature detector which relies on the Hessian matrix. The Hessian matrix of a point (x, y) is computed as:

$$H(x, y, \sigma) = \begin{bmatrix} L_{xx}(x, y, \sigma) & L_{xy}(x, y, \sigma) \\ L_{yx}(x, y, \sigma) & L_{yy}(x, y, \sigma) \end{bmatrix} \tag{69}$$

where σ is the scale factor and $L(x, y, \sigma)$ is the second order Gaussian convolution.
The blob score is then obtained by computing the determinant of the Hessian matrix:

$$det(H) = L_{xx} \cdot L_{yy} - wL_{xy}^2 \tag{70}$$

where w is a weight factor.
Finally, the features' locations are extracted by taking the non-zero determinants after a 3×3 non-maximum suppression is performed.
In order to provide scale invariance, The former steps are repeated using various filter sizes when computing the Gaussian convolutions.

C Common feature description methods

After features have been detected, a descriptor is used to uniquely describe its proximity in the image in order to match identical points in different images.
The first two descriptors, SIFT and SURF, are float-point descriptors, meaning that the feature description array is composed of float values. On the other hand, FREAK, BRISK, BRIEF and ORB, are binary descriptors meaning that the intensity of specifically-chosen pixels around a feature are compared using bit-wise operations. Finally, the last descriptor discussed utilizes machine learning to describe the image region around two features as well as providing their matching probability.

C.1 SIFT

Together with blob feature extraction, Scale-Invariant Feature Transform (SIFT) by D. G. Lowe [18] also provides a gradient orientation-based feature descriptor.
For each feature, an image patch of 16×16 centered around its location is taken. For each pixel inside the patch, its gradient orientation is then computed. Finally, the descriptor of length 128 is constructed by dividing the image patch into blocks of size 4×4 and, for each block, creating a 8-bin gradient orientation histogram.
To provide feature orientation invariance, the feature's orientation is subtracted from the gradient orientation.

C.2 SURF

To describe a feature, SURF by H. Bay et. al. [72] first computes the horizontal and vertical Haar wavelet response in a circle of radius 6σ around its location, where σ is the current scale. Using a sliding window of size $\frac{\pi}{3}$, the sum of the wavelet responses in each direction is taking. The feature orientation is then obtained by taking the orientation of the sliding window with the highest combined sum of wavelet responses.
Next, a window of size $20\sigma \times 20\sigma$ centered around the feature and rotated according to its orientation is divided into smaller squares of $4\sigma \times 4\sigma$. For each sub-square, the Haar wavelet response is computed in both the local horizontal and vertical direction, d_x and d_y respectively, at regularly-spaced intervals of 5×5 samples. The descriptor is finally formed by smoothing the responses using a Gaussian blur of 3.3σ and then taking, for each sub-square, the sum of horizontal responses $\sum d_x$, the sum of vertical responses $\sum d_y$ and the sum of the absolute responses in both directions $\sum |d_x|$ and $\sum |d_y|$, creating a descriptor of size 64.

C.3 BRISK

Binary Robust Invariant Scalable Keypoints (BRISK) by S. Leutenegger et. al. [19] is a binary descriptor, meaning that it generates the descriptor based on binary intensity comparison between pre-defined pairs of pixels in a pattern.
BRISK places N points at regular intervals in a circular pattern as in figure C.1. The intensity of the points are smoothed using a Gaussian function with a window size of the red circles. The $N \cdot (N-1)/2$ point pairs are separated in short and long pairs according to the distance between the two points. A short pair is then a pair whose distance is below a threshold δ_{max} and a long pair is a pair whose distance is above a different threshold δ_{min}. The long pairs are first used to determine the orientation of the feature by taking the orientation of the mean gradient of all the pairs in the long set. The gradient vector $\vec{g}$ of each pair is computed as:

$$\vec{g}(\vec{p_i}, \vec{p_j}) = (\vec{p_j} - \vec{p_i}) \cdot \frac{I(\vec{p_j}, \sigma_j) - I(\vec{p_i}, \sigma_i)}{||\vec{p_j} - \vec{p_i}||^2} \tag{71}$$

where $\vec{p_i}$, $\vec{p_j}$ are the two points in the pair and $I(\vec{p}, \sigma)$ is the Gaussian-smoothed intensity at location $\vec{p}$.
The short pairs are finally rotated according to the feature's orientation and used to create the de-

scriptor a bit is equal to 1 if $I(\vec{p}_j, \sigma_j) > I(\vec{p}_i, \sigma_i)$ and 0 otherwise.

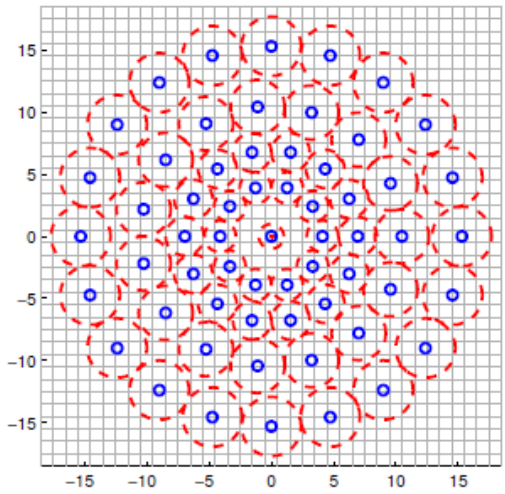


Figure C.1: BRISK pixel pattern [19]. The blue circles represent the pixel locations and the red circles the size of the Gaussian smoothing window

C.4 BRIEF

Binary Robust Independent Elementary Features (BRIEF) by M. Calonder et. al. [73] was the first binary descriptor, published in 2010. It does not offer any scale or orientation invariance.
A square patch $\mathbf{p}$ of width S is taken, centered around the keypoint location and smoothed using a Gaussian filter. A set of N point pairs are then selected, where the coordinates of the points are selected according to a Gaussian distribution with a spread of $0.04S^2$.
For each point pair $(\vec{x}, \vec{y})$, the value of the test $\tau(\mathbf{p} : \vec{x}, \vec{y})$ is equal to 1 if $\mathbf{p}(\vec{x}) < \mathbf{p}(\vec{y})$ and equal to 0 otherwise. Finally, the N-bit descriptor string $f_N(\mathbf{p})$ is related to the sum of the N tests performed in the patch $\mathbf{p}$ as follows:

$$f_N(\mathbf{p}) = \sum_{i=1}^{N} 2^{i-1} \tau(\mathbf{p} : \vec{x}_i, \vec{y}_i) \tag{72}$$

C.5 ORB

The rotated BRIEF (rBRIEF) descriptor utilized in Oriented FAST and Rotated Brief (ORB) by E. Rublee et. al. [69] is similar to the BRIEF descriptor explained in Appendix C.4. However, to provide orientation invariance, the BRIEF pattern is first rotated by the orientation of the feature found by the extractor. Additionally, the pairs are not chosen at random but has instead been determined beforehand using a training dataset. Doing so insures that the different pairs are uncorrelated and thus all provide new information to the descriptor. A high variance between the pair is also desired as to make the descriptor more discriminative.

C.6 FREAK

Fast Retina Keypoint (FREAK) by A. Alahi et. al. [20] is a binary descriptor that blends the properties of both the BRISK descriptor, as it uses a circular test pattern, and the rBRIEF descriptor from ORB, as it determines the pairs beforehand during a training session.
The test pattern used by FREAK is inspired by the receptors found on the retina of the human eye, where the points are located on circles centered around the keypoints. In contrast to BRISK, where the points are uniformly-distributed, the point density in FREAK is much higher near the keypoint

location and exponentially decreases further away from the center. The pairs were determined from a set of 50000 keypoints extracted from a training dataset. For each keypoint, a large descriptor composed of the test results from all possible pairs made from 43 points is computed. Then, for each pair, its mean value over the 50000 keypoints is computed and the pairs are sorted according to the smallest distance to 0.5. The smallest distance is kept and, in an iterative manner, additional pairs are added if their correlation with previously-selected pairs is lower than a given threshold.
A. Alahi et. al. [20] found that the obtained pairs could be classified in four groups in a coarse-to-fine point distribution manner, as depicted in figure C.2. This distribution enables the matching of features in a cascade manner, by first eliminating the obvious non-matches and progressively performing a more discriminative comparison.
The final descriptor string of 512 bits (four groups of 128 bits) is obtained as in BRIEF using eq. (72), where here the test $\tau(\mathbf{p} : \vec{x}, \vec{y})$ is equal to 1 if $\mathbf{p}(\vec{x}) - \mathbf{p}(\vec{y}) > 0$ and equal to 0 otherwise.

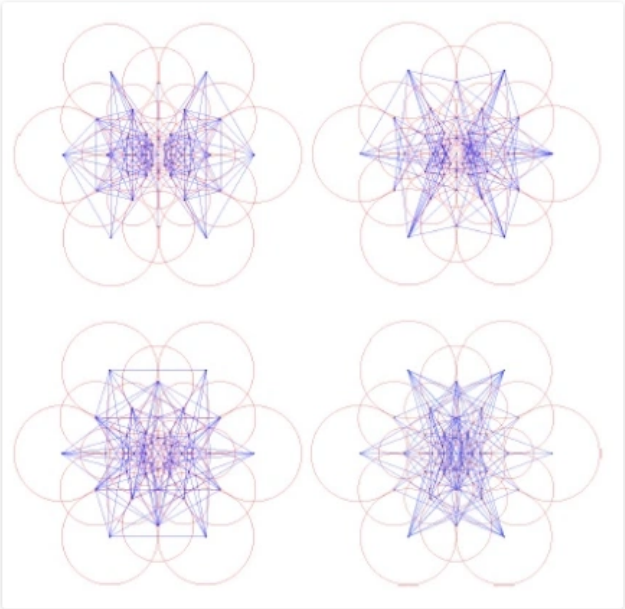


Figure C.2: The four test pattern groups constituting the FREAK descriptor [20]. The circles represent the window size of the Gaussian smoothing function

D Lens type determination

Before estimating the camera parameters, the lens type used needs to be classified as this will determine what projection and distortion model need to be adopted. With a horizontal field of view of 95°, it needs to be determined if the current lenses are (ultra-)wide lenses where the classical pinhole reprojection model is used together with a polynomial distortion model or fisheye lenses which utilize the special fisheye reprojection and distortion model.

The lens type is determined by both its focal length and the sensor size on which the rays are detected. With an image resolution of 640×512 and a pixel pitch of 0.012 mm [52], the FLIR Boson 640 has a sensor size of 7.68×6.14 mm. This is unfortunately not a standardized sensor size in the RGB camera industry and is situated between a 1/7" and a 1/6" sensor [74]. The classification of lenses is performed according to a standardized focal length, the 35 mm film format. The relation between the focal length in 35 mm f_{35} and the lens' focal length f is given as:

$$f_{35} = F_c \cdot f \tag{73}$$

where F_c is the crop factor. With a focal length of 4.9 mm and a crop factor of 4.55 or 4.30 for a sensor size of 1/7" and 1/6" [75] respectively, the lenses have a focal length in 35 mm film format between 22.3 mm and 21.1 mm.

Although there is no standard threshold in the camera industry that distinguishes wide lenses from ultra-wide lenses, lenses with a focal length under 24 mm in the 35 mm film format are seen as ultra-wide lenses by the majority of lens producers [76]. Additionally, lenses are qualified as being fisheye lenses if their 35 film format focal length is under 15-16 mm [77]. It can therefore be concluded that the lenses equipped on the FLIR Boson cameras used in this project can be considered as ultra-wide lenses, meaning that the regular pinhole projection model will be used for the camera calibration.

E CNN-based descriptors architecture

In figure E.1, the kernel information is given as $\langle kernel\ width \times kernel\ height \times input\ layers \times output\ layers \rangle$. For a dense layer, the layer information is given as $\langle input\ size \times ouput\ size \rangle$.

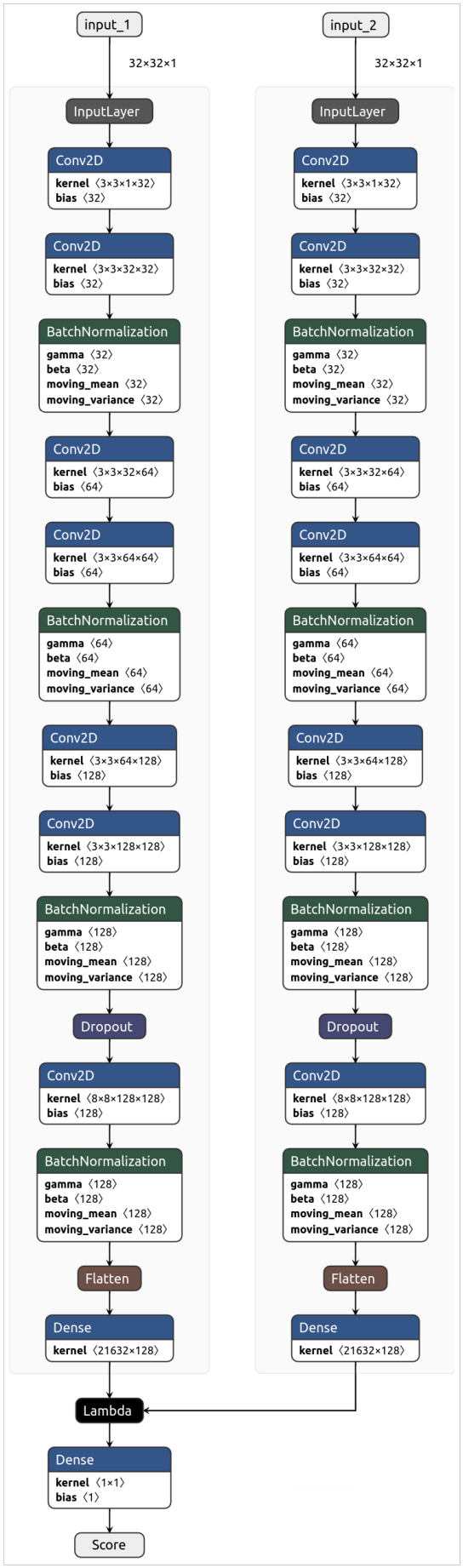


Figure E.1: Siamese CNN architecture for feature matching. Input and output sizes are given for a input size of 32×32 and a descriptor of size 128