RAM ROBOTICS AND MECHATRONICS

PERCEPTION ALGORITHMS FOR LOCALIZATION OF A PNEUMATIC ENDOSCOPE

L.I. (Laura) Rusu

MSC ASSIGNMENT

Committee:
dr. ir. M. Abayazid
Y.X. Mak, MSc
dr. ir. B. C. Schwab

August, 2022

038RaM2022
Robotics and Mechatronics
EEMCS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

UNIVERSITY OF TWENTE. | TECHMED CENTRE UNIVERSITY OF TWENTE. | DIGITAL SOCIETY INSTITUTE

Abstract

"Perception algorithms for localization of a pneumatic endoscope" aims to help increase the level of safety and efficiency of colonoscopies by tracking the tip of the colonoscope and localizing it in terms of the intestinal anatomy. While there are visual, electromagnetic, or other position detection solutions currently in use, they mostly focus on colons with typical anatomy, whereas this project aims to take into consideration difficult scenarios, such as deformation or poor quality of intestinal features.

Using as the primary method a monocular Simultaneous Localization and Mapping (SLAM) algorithm, this research focuses on localizing the distal end of the colonoscope visually, in terms of the surrounding features. Based on a literature comparison of multiple SLAM algorithms, ORB-SLAM (Oriented FAST and Rotated BRIEF SLAM) was considered the best alternative. The validation of the results is done with an Aurora electromagnetic sensor.

The report will start with a general context and a literature review that serve as the starting point for choosing the methods. Furthermore, the approach is described. After a theoretical overview of the employed methods (ORB-SLAM3, blob detection, camera calibration), the report will contain a description of the software and hardware, as well as the experimental setup. Finally, the results are presented, along with a discussion on limitations and future work, and conclusions.

Index Terms — Colonoscopy, ORB-SLAM, Tracking

Preface

Student name: Laura Irina Rusu
Student number: 2379864

Research group: Robotics and Mechatronics
Study field: Electrical Engineering
Project period: 12th July - 8th August 2022

A presentation on this project was held at the University of Twente, Enschede on the 8th August 2022

Supervisor: Ir. Yoeko X. Mak

This project is part of the student's master thesis at the University of Twente, Faculty of Electrical Engineering, Mathematics and Computer Science (Dutch: EWI, Elektrotechniek, Wiskunde en Informatica), Specialization Robotics and Mechatronics.

Document version number: 15

Acknowledgements

I would like to express my sincere gratitude to my primary supervisor, Yeoko Mak, and Dr. Momen Abayazid who guided me throughout this project.

Contents

1 Introduction

1.1 Background

Since the advent of the modern minimally invasive surgery (MIS) in 1983, through a laparoscopic appendectomy performed by Semm [1], this new technique became the staple for appendectomies [2], cholecystectomies [3], adrenalectomies [4]. This is due to great benefits, both aesthetic and medical, compared to open surgery (OS).

Minimally invasive procedures either involve small incisions or make use of natural body openings through which an endoscope is inserted. A few examples are *endoscopy* (a procedure used to examine the interior of a hollow organ or cavity of the body), *laparoscopy* (an operation performed in the abdomen or pelvis using small incisions with the aid of a camera), *arthroscopy* (performed on a joint).

1.1.1 Comparison between OS and MIS

Minimally invasive surgeries can be robotic or endoscopic. Surgeries of the robot-assisted variety allow doctors to operate from a console equipped with controllers that maneuver robotic arms. Surgical instruments directed by software have very precise and easily reproducible movements and the high-definition 3-D imaging allows the surgeon to see the surgical procedure better than in an OS that uses a conventional camera.

The other type of MIS is endoscopic surgery. During endoscopic surgery, the surgeon inserts a flexible tube equipped with a camera through a small incision or a natural orifice, such as the mouth. The tube has a channel to utilize microinstruments, which the surgeon uses while viewing the organs on a computer monitor [5].

A short comparison of procedures that can be performed by one or both MIS methods is shown in table 1.1. For brevity, only up to three surgeries from each category are mentioned. As shown in the table, some surgeries are only suitable for one of the two MIS types, with endoscopy allowing for a larger variety. However, each surgery is unique, and some methods may not be suitable for a given patient. Minimally invasive surgeries require general anesthesia and take longer to perform than open surgery, making the latter more suitable for some patients [6].

Furthermore, MIS presents a prolonged learning curve for most surgeons, in comparison with the learning process in open surgery. It is also more expensive, as the necessity of disposing of the used equipment and the prolonged operating times increase the cost. In the various healthcare systems around the world, these increased costs are not always compensated for by shorter hospital stays.

However, with more research done every year, it seems that the advantages of MIS far outweigh the negatives. Research carried out on the stress response during and after minimally invasive procedures shows great improvements to similar tests performed on OS patients [7].

Category	Surgery Type	Robotic	Endoscopic
General	Pancreatic cancer	✓	✓
	Liver tumours	✓	✓
	Hernias		✓
Lung	Some lung tumours	✓	✓
	Esophageal cancer	✓	✓
Gynecological	Endometriosis	✓	✓
	Uterine prolapse	✓	
	Conditions req. hysterectomy		✓
Head and Neck	Head and neck cancer	✓	
	Skull base brain tumors		✓
Heart	Mitral valve repair	✓	✓
	Atrial fibrillation	✓	
	Aortic regurgitation		✓
Neurosurgery/Spine	Spine conditions		✓
	Cervical disk hernias		✓
Vascular	Varicose veins		✓
	Venous insufficiency		✓
Urological	Kidney disorders	✓	✓
	Kidney cancer	✓	
	Kidney donation		✓

Table 1.1: Comparison regarding usage of robotic and endoscopic procedures [5]

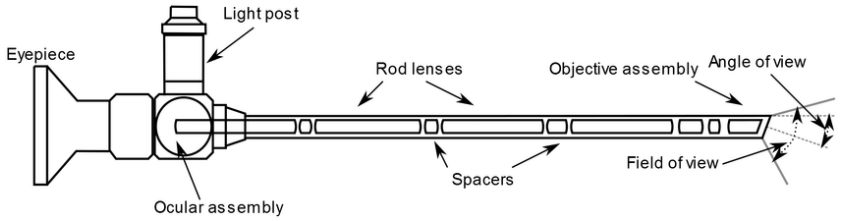
While open surgery is, by definition, an invasive procedure that leaves large, painful wounds that take a long time to heal, MIS is performed through multiple, smaller incisions. This means the recovery time is shorter and less painful in comparison to OS [2], [8]. More so, it yields fewer wound infections and hernias, reduced pain, and a reduction in ileus (bowel obstruction) compared with the open appendectomy (OA).

1.1.2 Evolution of Endoscopy

Since its first implementations in 400 BC [9], endoscopy has evolved tremendously. Throughout its long history, its design was constantly improved as new challenges came to light. The main limitations of these incipient endoscopes were the lack of an adequate light source to illuminate the area exposed by the distal end of the instrument and the rigidity of the endoscopic rod [10]. Without proper illumination, endoscopies could either not be performed or would not yield sufficiently valuable information. Without flexibility, the scope of the operation was severely restricted.

In modern times, these particular issues have long been solved. More so, endoscopes have evolved to better fit each region of the human body. At the moment there are rigid, flexible, and rigid-flexible endoscopes, each with its own purpose. Figure 1.1 shows a simple rigid endoscope [11] and a schematic representation of its components [12].

(a) Rigid endoscope, photograph [11]

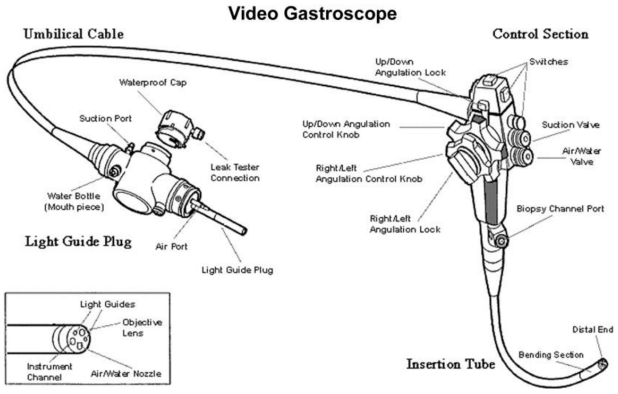


(b) Rigid endoscope, schematic [12]

Figure 1.1: Rigid endoscope

A flexible endoscope can be seen in figure 1.2, both as a photograph [13] and as a schematic [14]. The distal end of a flexible shaft is a steerable tip, of 4 to 10 cm in length, which contains a digital camera chip, light provided by light-emitting diodes (LEDs) or glass fibers, and the exits of air, water, and instrument channels. The instruments embedded in the distal end (small forceps, needles, electrocautery instruments) are used to do biopsies or surgeries close to the endoscope tip [15].

(a) Flexible endoscope, photograph [13]



(b) Flexible endoscope, schematic [14]

Figure 1.2: Flexible endoscope

Rigid endoscopes can be used in foreign body removal from children, while a flexible version is more suitable for adults [16]. Ferarri et al. concluded that flexible and rigid endoscopes yield similar results, and the decision to use one over the other should be made for each patient [17]. Cheung et al. concluded that flexible esophagoscopy offered more information, and was easier and safer to perform than rigid esophagoscopy

in esophageal cancer patients [18]. On tests performed during outpatient hysteroscopy, flexible endoscopes proved less painful but rigid hysteroscopes provided superior optical qualities and faster performance with higher success rates at much lower cost [19]. Some tasks, such as endoscopic third ventriculostomy and biopsy to approach posterior third ventricular tumors, are better suited for a combined rigid-flexible endoscope, a technique that overcomes the limitations of using a rigid endoscope by reaching two distant regions [20].

Flexible endoscopes are superior to the rigid ones in terms of maneuverability but are not stiff enough to perform surgery, and localization inside the body can be difficult to track. Research is conducted currently to develop flexible endoscopes that can be maneuvered and tracked precisely. Advances have been made in terms of haptic feedback [21], but there are still areas to improve, especially in terms of precise positioning of the endoscope's end-effector.

1.2 Problem statement

Flexible endoscopy has evolved to a point where it combines accurate internal imaging, low invasiveness, and increased patient comfort in comparison to rigid endoscopes. However, some issues still remain [22].

Kurniawan et al. [23] have compiled a selection of clinical challenges related to the use of flexible gastrointestinal (GI) endoscopes such as avoiding incomplete endoscopies or improving adenoma detection, along with the technical approaches used to tackle them (for the problems mentioned before, using colonoscopes of variable stiffness or increasing the number of attached cameras, respectively). For the purposes of this paper, only a pair will be brought into focus as the scope would be too broad otherwise.

First of all, the distal end of the endoscope is difficult to locate and track accurately. This is especially important, as the clinicians executing the endoscopy have to know exactly where the problem areas are located in order to be able to act upon the information received from the endoscope. Relatively experienced clinicians would be able to identify the main sections of the colon (such as the sigmoid colon, descending colon, transverse colon) as they explore it. However, it is still difficult to localize the endoscope tip with regards to a smaller section, such as a surgical target the endoscopists will have to return to. At times, these small targets might not show up on a CT scan. For example, the study of Lee at al. [24] showed a CT scan accurately localize colon tumors in 50% of the cases, with the rest being either incorrectly localized (17.3%) or not detected at all (32.7%).

Secondly, optimal detection and classification of internal lesions can be solved by enhancing the image procured by the endoscope, either by improving the camera resolution or by using chromoendoscopy [25] [26], an expensive endoscopic technique [27].The problem of high-quality images also affects the tracking process, as better video input yields better localization.

While the second issue presented does not relate directly to tracking, which represents the purpose of this project, suggests that using just a camera might not be sufficient for localization. This is especially important since there is a very important trade-off between the size and the resolution of the camera. A camera that is too large would involve the use of an anesthetic, which might be dangerous, as mentioned above.

1.2.1 Challenges with tracking

Situations in which tracking is a significant issue are enumerated below:

1. strangely shaped colons, with too many twists and turns;
2. areas of the colon with smoother textures, lacking in features;
3. deformations, caused as natural responses to the insertion of a foreign object;
4. colons that require treatment or surgery following a precursory endoscopy, in which it is important to locate the endoscope in terms of the anatomy of the colon.

While performing a colonoscopic examination, physicians control the colonoscope based on the images provided by the colonoscope, which only show views from the colonoscope tip. The physicians then have to estimate how the colonoscope is traveling inside the colon based on their own experience. However, position estimation is difficult because both the colon and the colonoscope have long and winding shapes and the intestine changes its shape significantly during the colonoscopy. Some inexperienced physicians may cause complications such as looping or colon perforation during examinations [28]. Even experienced physicians may have issues at times, especially if the colon has an atypical shape. Even in an ordinary case, the elasticity of the colon can lead to the misestimation of the colonoscope position or colon anatomy by the physicians performing the operation.

One example of an unusual colon shape is volvulus, twisting of the colon, shown in figure 1.3.

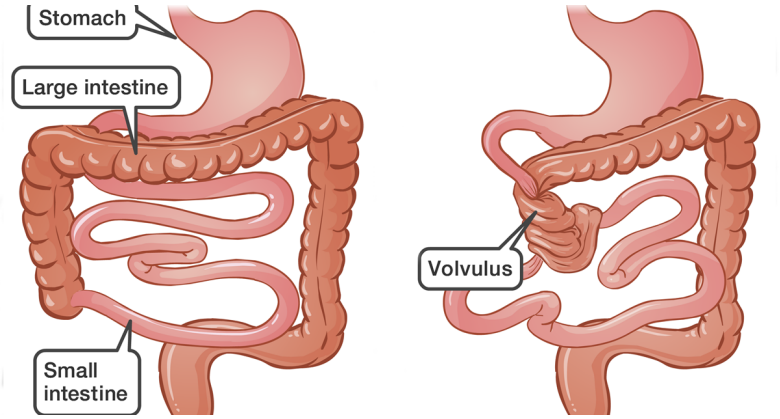


Figure 1.3: Normal colon (left) and congenital GI malformation (right) [29]

"Looping" is illustrated in figure 1.4. This occurs in over 90% of the examinations, although experienced endoscopists know how to prevent or correct it quickly [28]. Looping means the colonoscope stretches and deforms the intestine instead of advancing. Once the endoscope forms such loops, they must be rectified before the procedure can continue. At best, this prolongs the process and at worst, it can prevent the colonoscopy from being fully performed.

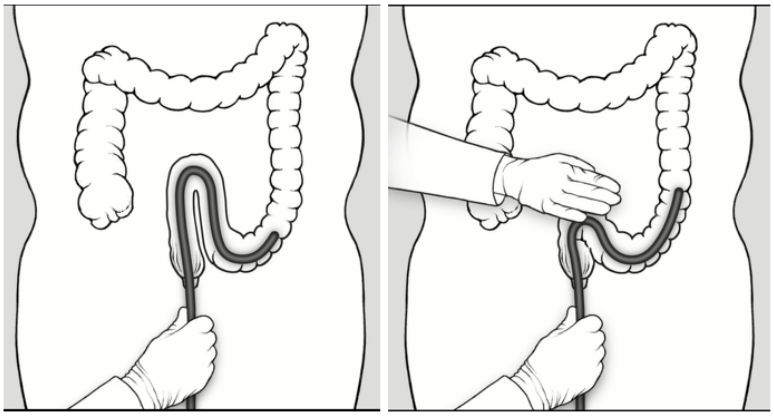


Figure 1.4: Endoscopic looping [30]

The frequency of looping can be reduced if the surgeon receives sufficient visual and spatial information from the endoscope regarding the anatomy surrounding the endoscope. Reinsertion for surgical purposes can be more efficient if the endoscope end can be localized in terms of these problematic areas from an initial consultation.

Finally, the elasticity of the colon can prove detrimental to the localization accuracy of the distal end. Due to friction interaction with the colon wall, the endoscope's tip can suddenly slip, causing the endoscopist to have difficulties in keeping track of the endoscope position with respect to the colon anatomy.

One way to prevent such problems is to introduce a rigorous navigation system for colonoscope examination that estimates the position and the trajectory of the colonoscope in the colon.

1.3 Research Question

In many cases seen in practice, the distal end is unable to be accurately tracked and located when it travels along the intestinal tract. Without a localization system, important information for the diagnosis, such as the distance that the endoscope has traveled or the region of the tract in which the end is located might be missing or is very difficult to estimate, especially when the colon presents difficult scenarios, such as deformation or abnormal anatomy. The lack of the position and orientation data of the end also constrains the capability to return to the areas of interest for re-inspection or follow-up interventions. To be noted that another issue with localizing an area of interest is further complicated by the rotation of the endoscope during the endoscopic process.

Taking into consideration the issues described above, the main research topic of the paper can be phrased as:

How can the end-effector of an endoscope be localized with regard to the anatomy of the colon, even in atypical scenarios, using a SLAM algorithm?

For this to be achieved, the following sub-questions have to be answered:

1. How accurately does an ORB-SLAM-based algorithm perform on various textures?
2. How accurately does the algorithm detect the positions of colon rings along the endoscopic trajectory?

The first question is answered through an experiment in four parts, where the ORB-SLAM algorithm is tested on the same trajectory, but with four different textures. Afterward, a second experiment is conducted to extract positional information of colored dots (markers), which are placed equidistantly along the trajectory. These dots represent the folds delimiting colon segments (haustra), illustrated in figure 1.5. The different textures specified earlier depict a simplified version of the texture of each segment of the colon.

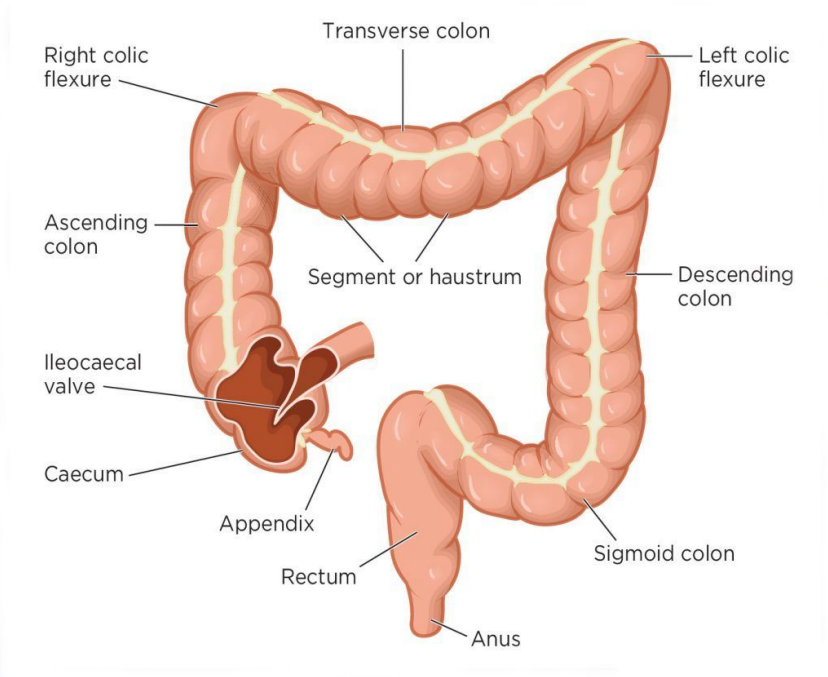


Figure 1.5: Illustration of the colon anatomy, including the haustra [31]

After the algorithm will detect these markers and record their position, an Aurora electromagnetic (EM) sensor will be used to verify the accuracy of these measurements.

This method of localizing the distal end helps override issues with deformation or atypical colons. In scenarios where returning to a given point is needed, knowing the distance in meters is not as useful as knowing the position of the endoscope in terms of the anatomy of the specific colon the procedure is performed on. More so, it allows the endoscopist to return to a point of interest more quickly and efficiently.

1.3.1 Approach

The solution explored in this study focuses on both tracking the endoscope throughout a section of the colon (both straight and slightly curved) and positioning areas of interest along the trajectory. These areas of interest are designated points on the haustral folds. Figure 1.6 illustrates relaxed and constricted colons, in which the lowest part of the haustral fold marks a dark spot on the colon. These spots served as inspiration for the technical approach, which will be discussed in detail in chapter 3.

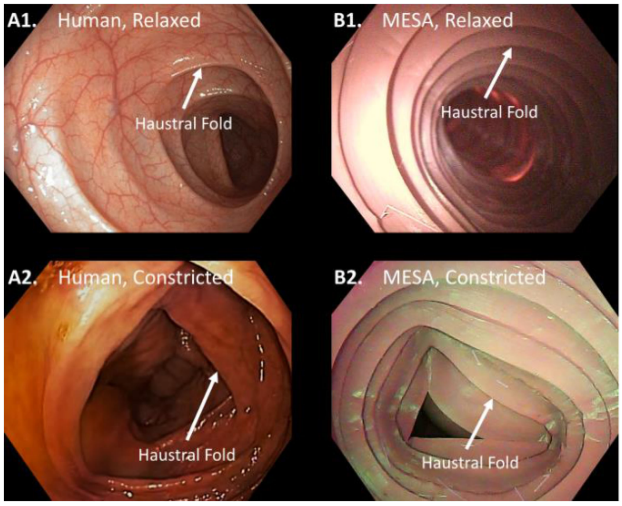


Figure 1.6: Relaxed and constricted colon views [32]

1.4 Overview of the report

This report is structured into seven chapters. After chapter 1, which indicates the background of the research and the research question, chapter 2, is dedicated to exploring the literature that stands at the basis of this project. Chapter 3, details the approach mentioned in 1.3.1. Chapter 4, discusses the software and hardware tools that made the practical side of the project possible. The final experiments and their results are described in chapter 5. These results are discussed in chapter 6. Finally, chapter 7, draws conclusions regarding the project as a whole. A list of abbreviations is appended at the end of the document.

2 Literature review

Before starting a discussion of current trends in endoscope localization, it is best that a few terms are defined. Observing the position of an agent throughout its movement and recording the sequential location points for processing is called *tracking*. *Mapping* refers to reconstructing an environment map by using the acquired environment information through the position and attitude information of the device [33].

2.1 Endoscopic tracking

Electromagnetic localization methods have been used quite often in endoscopy [34]. Electromagnetic endoscopes use the electromagnetic signal caused by driving sine signals through electrical coils. A more detailed description of the algorithm can be found in Peng et al. [35].

While some electromagnetic solutions are currently in use clinically, they still present some limitations, such as insufficient accuracy, excessive complexity, and lack of user-friendliness. More inaccuracies can be caused by metals present in the operating environment.

Generally, image-based localization seems to remain the most used option when it comes to endoscope tracking. The other methods are generally more expensive, or present difficulties in multiple sensor fusion. When using an X-ray method, for example [36], there is the danger of excessive radiation suffered by the patient. For the ultrasound localization [37], there is a need for a water medium, which complicates the operation further.

2.2 SLAM

One way of using visual elements to compute the location of a robotic device is by using Simultaneous Localization and Mapping (SLAM). As the name suggests, its goal is to concurrently build and use a map of local features. The device that has to be located uses its sensors to observe features, which are then used in locating the device on a map, although an absolute position is not possible to be determined. To make up for this shortcoming, SLAM uses statistical modeling that takes into account the odometry of the device to be located. Thus, most of the inconsistency between where the features are predicted to be and where it is based on the sensor readings is removed [38]. However, this is the case for binocular algorithms or algorithms which incorporate other sensors as well. Monocular SLAM lacks the depth information which would make this possible and will exhibit drift in time.

Table 2.1, based on information collected by Xie et al. [33], describes the main different types of SLAM currently available. By analyzing the advantages and disadvantages of each type of SLAM, it is easier to note which is the most suitable for a specific purpose.

	Category	Description	Advantages	Disadvantages
Image acquisition	Monocular [39]	Uses one camera	cheap; small size; low power usage; easy to calibrate	scale and motion estimates drift in time
	Binocular [40]	Uses two cameras	no scale drift	more expensive than the monocular; pretty large;
	RGB-D [41]	Uses an RGB-D camera, which includes a depth sensor	less computation needed; no scale drift	large power consumption; large size; quite expensive
Front-end registration	Feature-based ex: ORB-SLAM [42]	Extracts features from the images collected by sensors and registers these features to achieve the union of features	low computational complexity; sufficient for navigation	the tracking fails if the texture is sparse (because feature points can't be extracted)
	Dense-based ex: LSD-SLAM [43]	Point-to-point registration of dense point clouds though ICP (Iterative Closest Point)	uses information of the whole image; stable if the texture is sparse	significant amount of computation; difficult to navigate if the inter-frame displacement is too large
State estimation	Filter-based ex: Recursive Bayesian estimation [44]	May use as frameworks: Kalman filters (KF), or Extended Kalman Filters (EKF); Particle filters (PF); Expectation maximisation (EM)	can mostly handle both linear and non-linear situations; can handle process and measurement noise	computational complexity is high; limited update efficiency; may produce inconsistent estimates due to errors introduced during the linearization process [44]
	Smoothing-based (graph optimisation) [45]	Used to evaluate the position of the device and construct the map through multiple position information points	in Karlsson et al. [45], it is shown to create large maps in near linear time	shows some sensitivity to drift

Table 2.1: SLAM techniques comparison

As it is mentioned by Xie et al. [33], the most suitable SLAM algorithm for inside the human intestinal environment is a dense-based monocular variation. After comparing Scale-Invariant Feature Transform (SIFT), Binary Robust Invariant Scalable Keypoints (BRISK), ORB, and Fast Retina Keypoint (FREAK) in terms of feature extraction time, correct matching rate, and attitude estimation accuracy under rotation and scaling transformations, the ORB binary feature operator has proven to be the best solution. The ORB-SLAM had fast feature extraction, a high correct matching rate, and a small estimation error [33].

2.3 ORB-SLAM3

2.3.1 General aspects

To understand the way ORB-SLAM works and how it is used in colonoscopy, it is needed to build on the foundation set in subsection 2.2.

As mentioned prior, SLAM is the problem of concurrently creating a model of an environment and calculating the trajectory of the agent that explores the said environment. First introduced at the IEEE Robotics and Automation Conference in 1986, SLAM has a long history of being continuously improved, with new algorithms derived from the original concept.

ORB-SLAM is one of these algorithms based on ORB features. It is a sparse, feature-based SLAM method, which allows an easier transition from images to geometry while providing robustness to partial occlusion [46]. A disadvantage is that, over time, small drifts will start to occur due to errors in motion estimation and feature extraction, and the accumulation of these small errors in each frame will cause a large drift by the end of the trajectory. As the last step of the map reconstruction, ORB-SLAM uses bundle adjustment.

Bundle adjustment minimizes the reprojection error between the locations of the observed and, respectively, the predicted image points. Mathematically, this is expressed as the sum of squares of a large number of nonlinear, real-valued functions. The minimization is then obtained by using nonlinear least-squares algorithms [47], as can be seen in equation 2.1.

$$\min_{\mathbf{a}_j, \mathbf{b}_i} \sum_{i=1}^{n} \sum_{j=1}^{m} v_{ij}\, d(\mathbf{Q}(\mathbf{a}_j, \mathbf{b}_i), \mathbf{x}_{ij})^2, \tag{2.1}$$

where n is a number of 3D points, m is the number of views corresponding to them, and x_{ij} is the projection of the i^{th} point on image j. Each point i is parameterized by a vector b_i, while a_j is the parameter vector for each frame j. If point i is visible on image j, the binary variable v_{ij} will be equal to 1. Otherwise, it will be 0. $Q(a_j, b_i)$ represents the predicted projection of i on j. Thus, bundle adjustment takes into account all points and camera parameters when calculating the minimum of the total reprojection error.

Levenberg–Marquardt is one such algorithm widely used due to both an ease of implementation and an efficient convergence from a wide range of initial guesses as produced by its damping strategy [48]. In ORB-SLAM, this algorithm is contained within the General Graph Optimization (g2o) framework [49].

The ORB features used by this type of SLAM algorithm refer to "Oriented FAST and rotated BRIEF" [50], where Features from Accelerated Segment Test (FAST) is a feature detector and BRIEF, a binary descriptor. Compared to other feature types, ORB is more efficient, resistant to image noise, rotation invariant, and multi-scale.

ORB-SLAM also incorporates a bag-of-words description of the keyframes for place recognition [51], which uses a vocabulary tree to speed up correspondences for geometrical verification.

Finally, the global map and loop closure are based on the covisibility graph of Strasdat et al. [52].

With the general aspects out of the way, it is time to discuss the three parts of ORB-SLAM: tracking, local mapping, and loop closing pictured in figure 2.1.

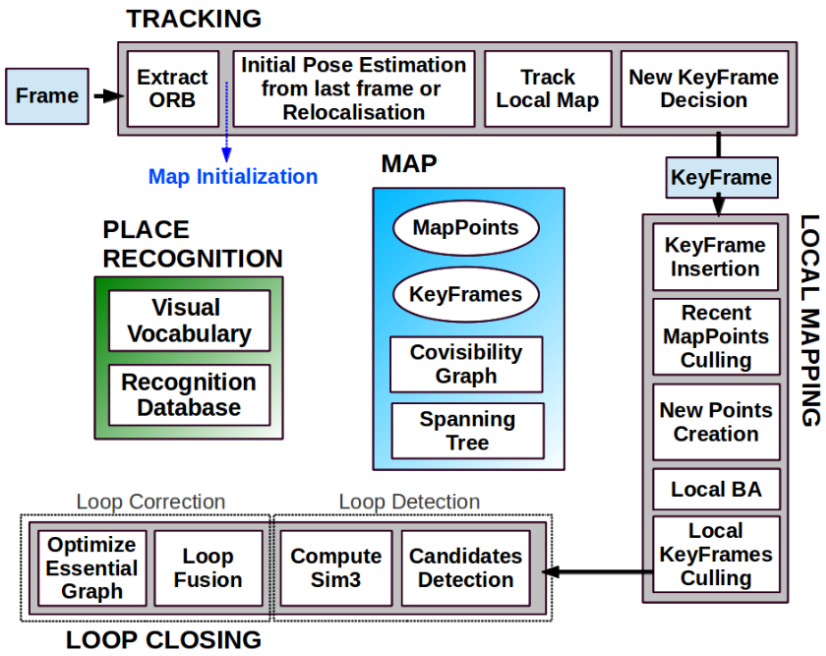


Figure 2.1: ORB-SLAM algorithm diagram, showing the Tracking, Local Mapping, and Loop closing threads [43]

2.3.2 Map initializaton

The first step to working with this algorithm is initializing the map since depth cannot be estimated from just one image. This is done based on the estimated pose between the first two frames. ORB-SLAM computes two geometric models in parallel, one assuming

the scene is planar and a fundamental matrix which works better with non-planar ones. The initial map is only created once one of the two models results is an acceptable configuration.

The map initialization is done by following a five-step algorithm:

1. The user sets a match threshold in the calibration file. Based on this value, the algorithm will attempt to find a number of feature correspondences between the current frame (F_c) and the previous frame, considered reference (F_r). The process is repeated until a good enough set of matches $x_c \Longleftrightarrow x_r$ (where x_i is a point in frame F_i) is obtained.

2. To obtain the best configuration, two geometric models are computed in parallel, with the same prefixed number of iterations for homogeneity. Equation 2.2 shows the relation between the point in the current frame and the one in the reference frame, where H_{cr} is a homography, x_c is a point in frame F_c and x_r is a point in frame F_r. The homography is computed with normalized Direct Linear Transform (DLT) for 4 points.

$$\mathbf{x}_c = \mathbf{H}_{cr}\mathbf{x}_r \tag{2.2}$$

   At the same time is calculated the fundamental matrix, F_{cr} in equation 2.3, where x_c and x_r have the same meanings as before. For this model are needed 8 points, in order to fulfill the requirements for the eight-point algorithm.

$$\mathbf{x}_c^T \mathbf{F}_{cr} \mathbf{x}_r = 0, \tag{2.3}$$

   Each iteration ends with a score S_M for each of the two models, calculated as shown in equation 2.4:

$$S_M = \sum_i (\rho_M(d_{cr}^2(\mathbf{x}_c^i, \mathbf{x}_r^i, M)) + \rho_M(d_{rc}^2(\mathbf{x}_c^i, \mathbf{x}_r^i, M)), \tag{2.4}$$

   where d_{cr}^2 and d_{rc}^2 are the symmetric transfer errors from one frame to the other, T_M is the outlier rejection threshold (equal to 5.99 for homography and 3.84 for fundamental matrix assuming a standard deviation of 1 pixel in the measurement error), and $\rho_M(d^2)$ is calculated as follows:

$$\rho_M(d^2) = \begin{cases} \Gamma - d^2 & \text{if } d^2 \ T_M \\ 0 & \text{if } d^2 \geq T_M \end{cases}$$

   The geometric models with the highest scores are kept unless no model can be computed due to a lack of inliers, in which case the process is restarted.

3. The appropriate model is selected: a homography if the scene is planar or has low parallax, or a fundamental matrix if the scene is non-planar. There are situations, however, in which it is not as straightforward to decide on the best approach. The

equation shown in 2.5 is used if that happens to be the case. If the R_H is greater than 0.45, the homography will suffice.

$$R_H = \frac{S_H}{S_H + S_F} \tag{2.5}$$

4. Following the selection of a model is the retrieval of motion hypotheses.

   Upon choosing a homography model when R_H 0.45, the 8 motion hypotheses are extracted as described in the method of Faugeras and Lustman [53] unless the parallax is low. Within the context of ORB-SLAM, the authors suggest a direct triangulation of the eight solutions and checking if there is a motion hypothesis with significantly more points with parallax from both cameras and with low reprojection errors compared to the others. A lack of a clear winner leads to repeating the map initialization process from the beginning.

   If $R_H \geq 0.45$, the fundamental matrix is converted to an essential matrix as shown in equation 2.6

   $$\mathbf{E}_{rc} = \mathbf{K}^T \mathbf{F}_{rc} \mathbf{K}, \tag{2.6}$$

   where K is the calibration matrix. The four motion hypotheses are them retrived with the singular value decomposition, triangulated and the best solution is selected.

5. Lastly, bundle adjustment is performed on the best hypothesis. The result is the initial map.

2.3.3 Tracking

During tracking, ORB-SLAM attempts to find pose differences between each frame and the one before it as well as decides if a new keyframe can be created.

ORB is extracted constantly throughout the use of the algorithm as FAST corners at 8 scale levels with a scale factor of 1.2 (although these parameters can be changed from the calibration file). For a homogeneous distribution, each scale level is split into a grid with the expectation that each cell will contain at least 5 FAST corners, which are then described using the ORB descriptor.

These extracted ORB initialize the pose of the current frame given the fact that the camera has constant velocity. The map points detected in the previous frame are then looked for in the new one. In case there aren't enough matches, the search area for each of them is increased. These newfound matches, if sufficient, then optimize the pose.

In case the tracking is lost, the frame is converted to bag-of-words and its position is searched for in the global map by finding ORB matches between the current frame and the one before, then optimizing the pose of the new frame with motion-only bundle adjustment.

After obtaining this estimation of the camera pose and an initial set of feature matches, the map can be projected into the frame so that ORB-SLAM can search for more map

point correspondences. The local map, considered a set of keyframes, will use the one keyframe with the highest number of correspondencies as reference. Each point in this map is then projected onto the new image frame to check:

1. if it belongs to the image;
2. if the cosine of the angle between the points' viewing direction n and the current viewing ray v is higher than the product v ;
3. if the distance from the point to the camera center is within the scale invariance region of the map point.

These points are then compared to the rest of the unmatched ORB features in the current frame to match the point with the best contender.

Lastly, the algorithm needs to decide if the current frame needs to be a new keyframe. While, for robustness purposes, it is desirable to spawn as many keyframes as possible, all keyframe candidates must fulfill four conditions:

1. Since the last relocalization, the system must have passed through at least 20 frames;
2. Local mapping must be either idle or at least 20 frames must have passed from the last keyframe;
3. At least 50 points must be tracked by the current frame;
4. The current frame must track less than 90% of the points found in the reference keyframe.

2.3.4 Local Mapping

The algorithm updates the covisibility graph for every new keyframe by creating a node for it and updating edges resulting from the shared map points with other keyframes. This new keyframe is then represented using bag-of-words.

All detected map points must be both found in more than 25% of the frames in which they are predicted to appear, and the keyframe they spanned from must be seen from three other keyframes. In case these two conditions are not met, the points are culled (where culling is the process of removing part of a set while maintaining the amount of information as high as possible.) To create new map points, matches for previously unmatched ORB are searched in other keyframes. Based on positive depth, parallax, re-projection error, and scale consistency, the new pairs can either be accepted or rejected.

Local Bundle Adjustment (LBA) is used to optimize the currently processed keyframe and all connected map points and keyframes. Outliers are discarded. Furthermore, redundant keyframes (those whose 90% of points can be found in at least 3 other keyframes in the same or finer scale) are detected and deleted to keep the bundle adjustment computation complexity low.

2.3.5 Loop Closing

Throughout the running of the algorithm, every time a keyframe is added to the trajectory, the loop closing thread verifies if the conditions for loop closing are met.

If a loop closure is detected, the loop is aligned to the map by means of the adjustment of a similarity transformation. Duplicate points are fused, edges are upgraded, and one more optimization is performed.

Since this project uses straight trajectories for testing, this step of the ORB-SLAM3 algorithm is not used.

2.3.6 ORB-SLAM versions

The algorithm described in 2.3 is that of ORB-SLAM, the first version of ORB-SLAM published. Since the publishing of the original paper [43] in 2015, two more versions have appeared. It was considered that the best understanding of the algorithm can be gathered by first studying the original paper, the differences between versions discussed thereafter.

ORB-SLAM2 [54] has the added benefit of also taking stereo images or depth maps as inputs, which can be used to represent some of the ORB points as stereo points consisting of the coordinates in the left image and the vertical position in the rectified right image. This option can be very helpful for optimization algorithms.

ORB-SLAM3 [55] brings another very important advantage. Improved recall allows the ORB-SLAM3 to relocalize itself after periods of poor visual information. Specifically because of this characteristic, ORB-SLAM3 is the best suited for this project: colons very often have sections lacking in distinctive features and a robust relocalization algorithm is vital.

2.4 Blob detection

To simulate the haustra separately from the surface texture of the colon, black dots were placed at equidistant intervals throughout the endoscopic path. To perform the blob detection, the following parameters have been taken into consideration:

1. *Color*: lower values (0) of this interval lead to the detection of dark-colored blobs, while higher values (255) find the lighter-colored ones. In this case, the chosen values lie between 0 and 100;

2. *Area*: Useful to eliminate a significant portion of undesirable blobs in case the goal is to find some of a given size. For this project were chosen blobs between 500 and 2000 pixels.

3. *Circularity*: Focuses on how close the blob is to a geometric circle. Considering that the blobs used in this project are circular, but deformed due to the camera's point of view, the circularity was chosen as 0.3

4. *Convexity*: Obtained by dividing the area of the blob by the area of its convex hull, it illustrates how complete the blob shape is. The minimum value chosen for this algorithm is 0.3

5. *Inertia*: Referring to the elongation of a shape, the inertia was chosen as 0.05 (very elongated ellipse).

2.5 ROS

Finally, ROS (Robot Operating System) was used to bring everything together. ROS is a 'meta-operating system" that supports message passing between different processes across a network (Inter-Process Communication = IPC) IPC is needed when multiple different systems have to communicate for a common goal, when there is a need for modularity, or to connect systems written in different programming languages. To give an example from the current project, it allows the blob detection algorithm (written in Python) to communicate with the camera (a sensor that constantly feeds the blob detection information), to the SLAM algorithm (written in C++, which also makes use of the camera), and finally to the values communicated by the Aurora NDI (a second sensor with continuous feed.)

On a filesystem level, ROS uses a few particular types of files, such as packages, the main modular unit of organizing software, manifests (.xml files containing metadata about a package), repositories (groups of packages sharing a version control system (VCS), message and service types that describe the message and respectively service descriptions.

On a computation level, a ROS-user would encounter nodes (runtime processes), a master that allows these notes to communicate, topics (the messages routed by a node to another), bags (the format for saving and replaying ROS message data). Nodes can be either Publishers (in case they send a topic) or Subscribers (which receive that topic), as shown in figure 2.2.

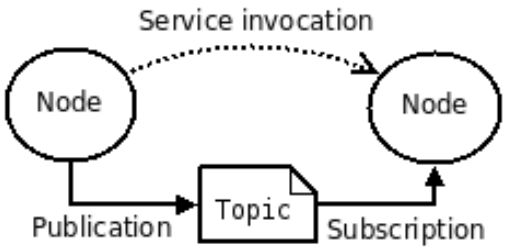


Figure 2.2: ROS basic functionality diagram [56]

Rviz is the 3D visualizer designed for the ROS framework used for visualization.

3 Approach

As was discussed in the first chapter , the research question can be split into two sub-questions, one taking into account just the accuracy of the endoscope trajectory on a phantom with various textures, while the second one focuses on the positions of the haustra delimiters in relation to trajectory.

The project described in this report, while not having direct usability in a clinical context, provides proof of concept for the proposed algorithm. A clinical workflow is simulated in a simplified manner, by using a phantom to act as the patient's colon and the endoscopic camera similar to those that might be used in endoscopy.

Figure 3.1 illustrates the approach. After calibrating the camera, the ORB-SLAM module receives the video feed from the endoscopic camera as input and returns the trajectory. The EM sensor, placed on the tip of the camera, acts as a validation agent. Concurrently, the dark blots symbolizing the lower part of the haustra are identified, then the centroid for each of them is calculated and converted to world coordinates. Together, the trajectory of the endoscope and the localization of haustra along the trajectory create a schematic representation of the patient's colon, allowing for easier identification of specific points the clinician might need to return to.

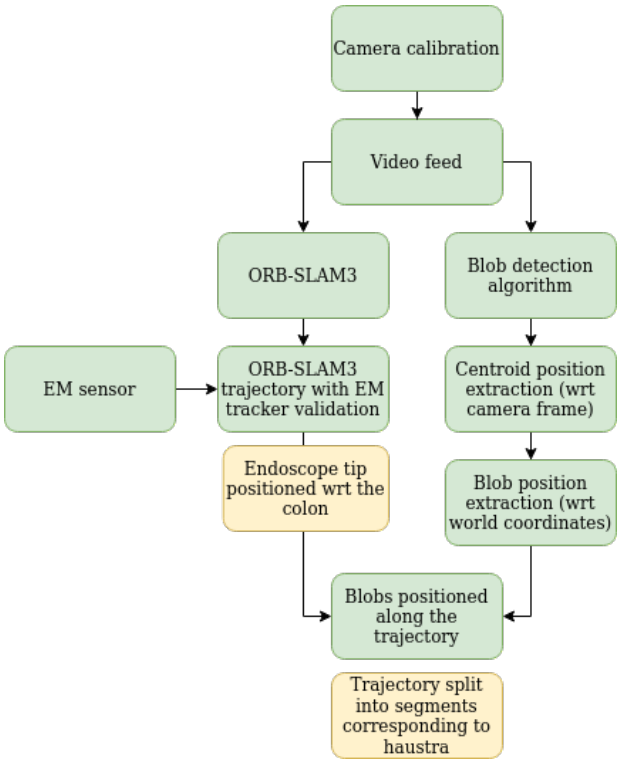
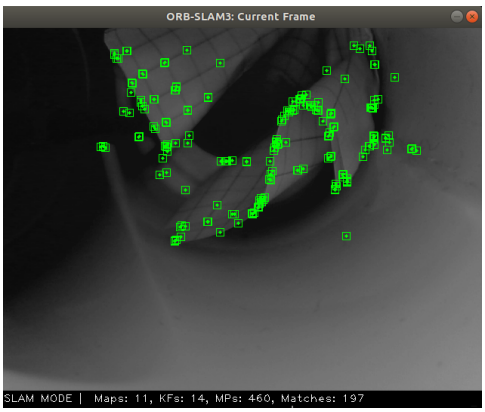


Figure 3.1: Schematic of proposed approach

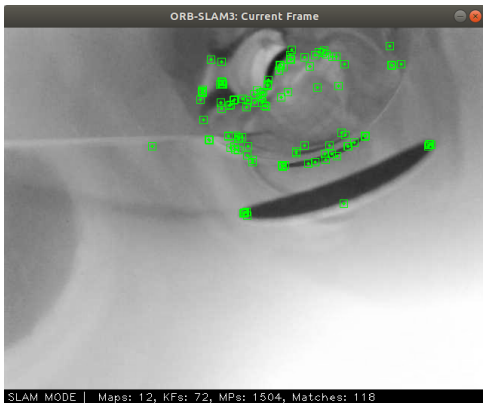
3.1 Endoscope localization with ORB-SLAM3

Compared to other SLAM algorithms, ORB-SLAM has faster feature extraction while using a relatively low computational complexity, a high accuracy of point matching, and small estimation error, as detailed in section 2.2. After an underwhelming trial with ORB-SLAM2, which had trouble finding sufficient features for efficient tracking, ORB-SLAM3 became the algorithm of choice following the explanation given in section 2.3.6. For the sake of comparing the ORB-SLAM trajectory with the EM sensor values, a ROS wrapper is used to transmit both the image and the positional data to the computer.

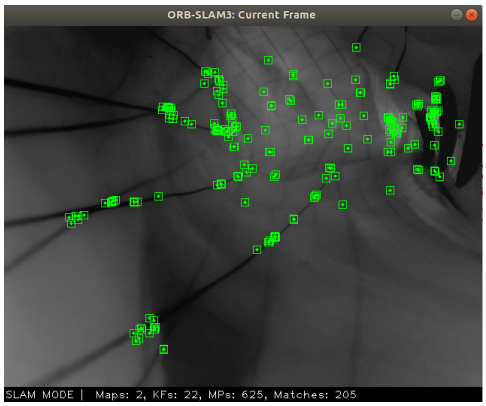
Figure 3.2 illustrates the way ORB-SLAM finds landmarks in the phantom without any other features, when using the blobs, and when using both blobs and striations.



(a) ORB-SLAM detection in the bare phantom



(b) ORB-SLAM detection in the phantom, with blobs



(c) ORB-SLAM detection in the phantom, with blobs and striations

Figure 3.2: ORB-SLAM detection in 3 cases

3.2 Haustral fold detection

Localizing the distal end with the help of haustra helps override issues with deformation or atypical colons.

Knowing the distance in meters is not as relevant in a clinical context as knowing the position of the endoscope in terms of the anatomy of the specific colon the surgery is performed on. This is especially important in circumstances where returning to a given spot is required for either reinspection or surgical operations. A precise localization enables the endoscopist to return to a point of interest with greater speed and efficiency. The localization happening during the operation can be correlated with information obtained prior, such as CT scans or MRI volumetric data. This way, the localization of the camera can be performed accurately based on existing data.

In this project, the haustral folds are represented by colored dots, symbolising the dark areas visible on haustral folds during colon constriction. The markers are placed equidistantly along the trajectory, thus creating 2 cm long segments between each pair of markers.

The black dots symbolizing the folds are present in all experiments, whereas the texture (lines) surrounding them changes. Blob detection was used for identifying the positions of these markers, as it is sufficiently different from the representation of the other texture to allow for a more fair observation.

3.2.1 Step for detecting haustra

The image received from the camera is denoised by passing it through a blur function, after which it is converted from BGR (Blue Green Red) to HSV. The previously found HSV values are used as a threshold, to obtain the image mask, which is then dilated and eroded to remove minor imperfections. The OpenCV function SimpleBlobDetector is used on this preprocessed image and detects the desired blobs from the image. The function's parameters dictate which blobs in the image are considered good candidates.

Given the parameters described in section 2.4, the detector is created, followed by the reversing of the original mask. The key points marking each blob are found by using this reverse mask and then drawn on the image to ease visual inspection. The detection is based on both the color of the blobs (black, chosen to stand out from the white of the phantom) and their shape.

The blob detection would not be useful if the blobs' position could not be extracted. This is achieved in a few steps:

1. finding the centroid of each detected blob, obtaining a set of cartesian coordinates with respect to the camera frame;
2. transforming the coordinates of the centroid to pixel values;
3. calculating the bearing angle for each blob using the previously obtained pixel values and the camera parameters extracted from the calibration file;

4. calculating the world coordinates of the centroid.

The positions of the markers are calculated and placed along the SLAM trajectory to create a better map of each colon. Before attaching these positions to the trajectory, each marker needs to be detected and its position extracted with respect to the camera frame. These cartesian coordinates are transformed to pixel values, then used for calculating the bearing angle. Finally, this angle is used for extracting the world coordinates.

3.2.2 Finding the centroid

Finding the centroid of a shape refers to obtaining the average of all the points in the shape. In this case, as the shapes are made of pixels, the centroid is the weighted average of all pixels making up the shape. Equation 3.1 shows the formulas needed to obtain the centroid.

$$C_x = \frac{M_{10}}{M_{00}}; C_y = \frac{M_{01}}{M_{00}} \tag{3.1}$$

where C_x represents the x coordinate and C_y, the y coordinate of the centroid. Values represented by M_{ij} represent raw image moments, which are weighted averages of the pixels' intensities. The moments can be extracted via an OpenCV function ("cv2.moments"), which takes a user-specified threshold which isolates the desired shapes (blobs) from the rest of the image. Equation 3.2 illustrates the process of obtaining moments from the rasterized image (which is an array of intensity values of pixels obtained from the original image), where x and y represent pixel coordinates.

$$\mathtt{M}_{ji} = \sum_{x,y} \left(\mathtt{array}(x,y) \cdot x^j \cdot y^i \right) \tag{3.2}$$

3.2.3 Field of view and bearing angles

The coordinates of the blob centroids with respect to the camera frame are two-dimensional and, in order to make the algorithm applicable in real-life scenarios, these two dimensions have to be transformed into their 3D counterparts.

Figure 3.3 illustrates the relation between the point in the image (with coordinates (u,v)) with its real-life analogue, $P(X,Y,Z)$ for a pinhole camera, which is the type used in this study. The principal point, of coordinates c_x, c_y is obtained from the camera calibration file.

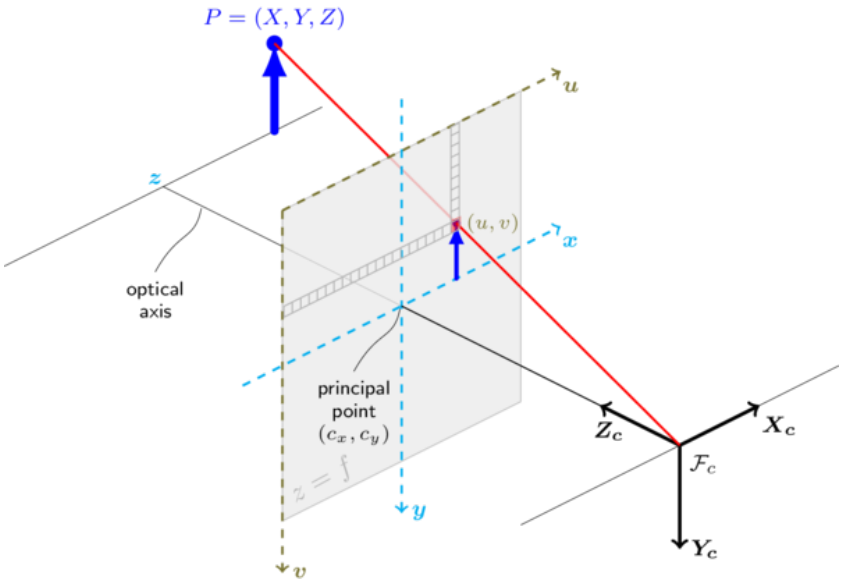


Figure 3.3: Pinhole camera model [57]

The perspective transformation from world coordinates to image coordinates can be seen in equation 3.3.

$$s\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{3.3}$$

where, other than the notations explained previously s is the scale; the first matrix on the right side is the matrix of intrinsic camera parameters; the second matrix on the right side is the joint rotation-translation matrix; and f_x, f_y, the focal lengths expressed in pixel units, found in the camera calibration file. Equation 3.4 illustrates the simplified calculation.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{3.4}$$

Figure 3.4 illustrates the relationships between all values needed to obtain the bearing angle for each blob position. The notations keep the same meanings as previously mentioned. The angle illustrated with yellow represents half of the wide field of view, which encompasses the entire frame. The angle colored with pink is θ, the bearing angle from the principal axis to the blob position.

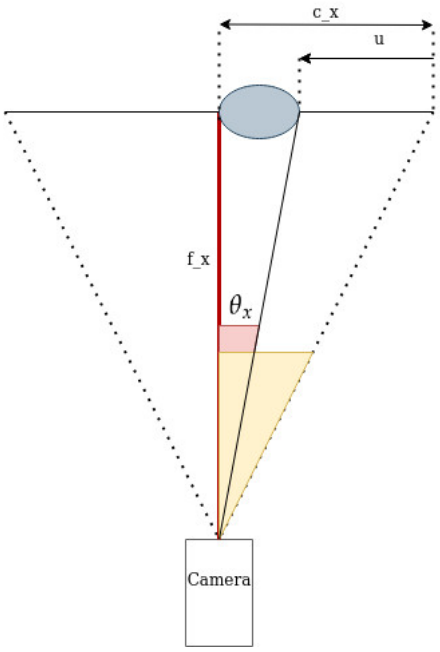


Figure 3.4: Field of view

The formulas to obtain θ_x and, respectively, θ_y are described in equation 3.5.

$$\begin{aligned} \theta_x &= \arctan(\frac{c_x - u}{f_x}) \\ \theta_y &= \arctan(\frac{c_y - v}{f_y}) \end{aligned} \tag{3.5}$$

Revisiting the steps mentioned in section , it is clear that the next step in finding the position of the haustra against the trajectory, is finding the centroid of the detected blobs. This is done by simply following the formulas provided in section . The resulting cartesian coordinates are expressed with respect to the camera frame and need to be transformed to pixel values.

The camera frame is structured as to have the point $(0,0)$ in the center, while the top-right corner has the value of $(1,1)$, and the bottom-left, $(-1,-1)$. To transform the centroid coordinates to pixels, we use the formulas from equation 3.6.

$$\begin{aligned} p_x &= \frac{1}{2}\text{width} + (\frac{1}{2}\text{width} * \text{centroid}_x) \\ p_y &= \frac{1}{2}\text{height} + (\frac{1}{2}\text{height} * \text{centroid}_y), \end{aligned} \tag{3.6}$$

where p_x is the location of the blob center in the image plane, in the horizontal axis, p_y is the center location in the vertical axis, width is the width of the image in pixels and height, the height in pixels. The width and the height can be extracted from the calibration file (see figure 4.5).

These pixel values p_x and p_y can be used in the equation 3.5, as u and, respectively, v. The c_x and c_y in the equation are the pixel coordinates of the principal point, and their values can be taken from the camera calibration file. The focal lengths, f_x and f_y, can also be found in the calibration file.

With the newly obtained θ_x and θ_y values, the distance from the camera to each blob is calculated by using the formulas in equation 3.7. $dist_{blobx}$ is the real-life equivalent to the c_x from image 3.4. More precisely, $dist_{blobx}$ is the distance from the side of the phantom to each blob, which is the radius of the phantom, 1.5 cm (along the x-axis illustrated in figure 3.5). $dist_{bloby}$ is the distance between two consecutive blobs, 2 cm. Since the blob positions do not change along the z-axis, but each blob needs 3 dimensions for an accurate representation, the values for z are obtained by averaging the z-values extracted by the EM sensor, assuming that the endoscope and the EM sensor connected to it are sufficiently close to the bottom of the tube, where the blobs are, that this value will be sufficiently accurate.

$$\begin{aligned} distance_x &= \frac{dist_{blobx}}{\theta_x} \\ distance_y &= \frac{dist_{bloby}}{\theta_y} \end{aligned} \tag{3.7}$$

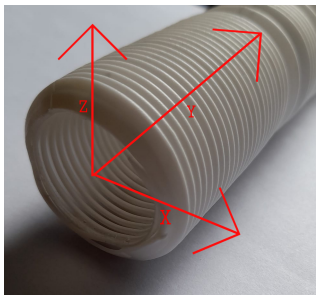


Figure 3.5: Coordinate axes respective to the phantom

When applying this algorithm to a clinical environment, the black markers will, of course, be replaced with the actual haustral folds. This changes the approach slightly. Instead of the blobs, the algorithm will focus on detecting the folds themselves. Since they have a relatively circular shape, even when constricted, it will be possible to calculate the centroid of the shape, which will no longer be placed on the "floor" of the colon, but an imaginary point in the air. This removes the necessity of calculating the distance from the edges of the colon to the point of interest, as this distance is just the radius of

the colon itself. The distance between each two folds (which is the value on the 'y' axis) can be approximated during the endoscopy itself, as they differ from patient to patient.

4 Implementation

To be able to execute the proposed experiments, it is needed to prepare the hardware and software tools used in the preparation of the algorithm and those needed for the experimentation portion of the project.

4.1 Hardware

4.1.1 Endoscopic camera

The endoscopic camera used for experiments is a Depstech USB endoscope camera with a 5 m cable, illustrated in figure 4.1. To improve visibility in the closed phantom, it is equipped with a ring of 6 LED lights around the head. As it is very flexible, it can mold itself easily around the curves of the phantom. It has a resolution of 640 x 480.

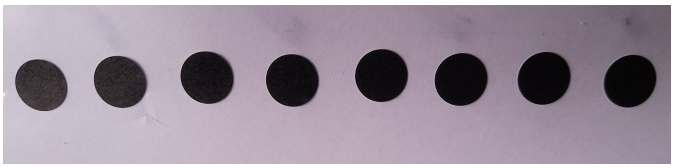
Figure 4.1: Depstech USB camera [58]

4.1.2 Phantom

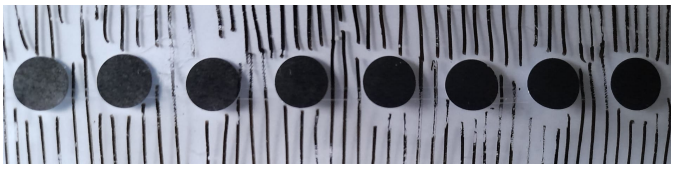
Both experiments will be conducted using a flexible, striated plastic tube with an inner diameter of 3 cm, used as a colon phantom. Figure 4.2 shows this tube in one of the bent states used during the experiments. Inside this tube are placed transparent plastic sheets with different textures.

Figure 4.3 shows the four types of texture used in this study. While the black dots simulate the dark spots created by haustra along the colon, the presence (or absence) of lines represents the texture of the colon wall, between two consecutive haustra. A lack of lines simulates the case in which the texture is not visible at all; the short, long, or hashed lines are meant to emulate the blood vessels visible inside the colon.

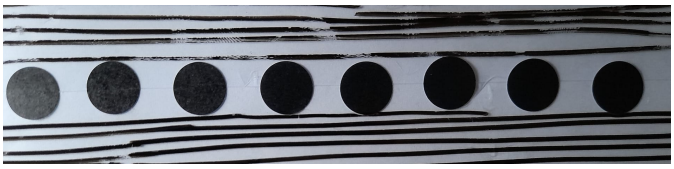
Figure 4.2: Tube acting as phantom



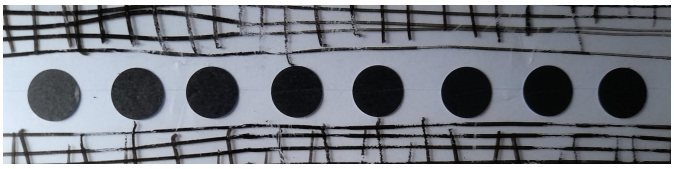
(a) Just black dots



(b) Short striations



(c) Long striations



(d) Long striations

Figure 4.3: Texture types used in the experiments

4.1.3 Computer

For all the development and testing needs of the project, a Ubuntu 18.04 PC with an Intel Core i7-10750H with 16GB DDR4 3200Mhz Memory and an Intel Iris XE Graphics G7 80UE graphics card will be used.

4.1.4 AURORA NDI

Aurora systems are the most widely used electromagnetic tracking systems in surgical navigation [59], due to their real-time tracking, precise positioning, and miniaturized sensors easy to attach to various surgical tools. The experiments were performed with a 6 degree of freedom (DOF) sensor attached to the endoscopic camera, as shown in figure 4.4. The NDI Aurora EM tracker operates at 40 Hz.

Figure 4.4: Endoscope and Aurora sensor

4.2 Software

4.2.1 Camera calibration

As the endoscopic camera is a pinhole model, it presents some deformation that needs to be rectified. Because of this, a static camera calibration was performed before starting any work with the algorithms. This was done with the help of the camera_calibration package, part of the ROS stack image_pipeline. It allows easy calibration of the monocular camera and a simple calibration checkerboard as the target. Afterward, all the parameters have to be introduced (such as the number of squares in the checkerboard, the size of each square, and directing the input towards the endoscopic camera). A window then appears on the computer screen, recording all the movements made by the checkerboard along the X, Y, and Skew lines until sufficient movements have been made along all of them. After pressing the calibration button on the screen, a calibration file is generated.

The calibration file contains the intrinsic and extrinsic parameters of the endoscope camera, which is used to normalize the frames used as input.

The generated calibration file is, however, in a different format than what the ORB-SLAM3 algorithm requires, which is used by the OpenCV library. Where the obtained calibration file provides four matrices (a 3x3 camera matrix, a 1x5 matrix with distortion

coefficients, a 3x3 rectification matrix, and, finally, a 3x4 projection matrix), the needed format, shown in figure 4.5 requires separate parameters for each value.

```
# Camera calibration and distortion parameters (OpenCV)
Camera.fx: 696.8738499999999
Camera.fy: 698.619824
Camera.cx: 349.066805
Camera.cy: 269.595258

Camera.k1: 0.119415
Camera.k2: -0.5023989999999999
Camera.p1: 0.00117
Camera.p2: 0.002633

Camera.width: 640
Camera.height: 480
```

Figure 4.5: Required calibration file format

When converting the original format into the second one, the *camera.fx*, *camera.fy*, *camera.cx*, *camera.cy* parameters can be found in the camera matrix as $\begin{bmatrix} fx & 0 & cx & 0 & fy & cy & 0 & 0 & 1 \end{bmatrix}$. The distortion coefficients can be extracted from the distortion coefficients matrix: $\begin{bmatrix} k1 & k2 & p1 & p2 & k3 \end{bmatrix}$, where *k1, k2, k3* represent the radial distortion and *p1, p2*, the tangential distortion.

4.2.2 ORB-SLAM3

A detailed description of the ORB-SLAM3 algorithm is not needed in this section, as it was provided in chapter 2. However, it will be mentioned that the ORB-SLAM3 algorithm was supplemented with a ROS wrapper written mostly in C++ that allows the publishing of extracted information through ROS nodes, information that is vital for experimentation.

Using the procedure mentioned in subsection 4.2.1, a calibration file is created to rectify the image received from the camera. This is done using the camera_calibration package from ROS with a checkerboard calibration sheet. The image will be used as input twofold: for the ORB-SLAM algorithm and the blob detector.

The algorithm itself can be seen as a schematic in figure 4.6. Receiving the image feed from /camera/image_raw, the /orb_slam3_mono node applies the ORB-SLAM3 algorithm. /tf is the transformation used to bring the SLAM points to a world coordinate system. Finally, /orb_slam3_ros/trajectory_server_orb_slam3 records the positions of these points.

4.2.3 Blob detection

A script is used to find the best values for the blobs in the HSV (Hue Saturation Value) color space, yielding the values (255, 165, 88) for outside of the phantom and (255, 255, 80) for inside.

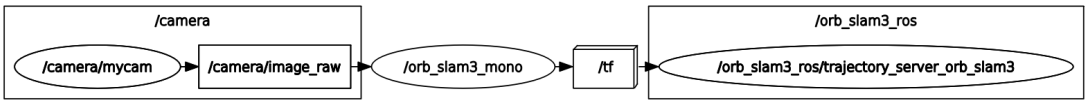


Figure 4.6: ROS schematic showing the connections between different blocks of the algorithm

The camera node (/camera) reads the image input from the endoscopic camera and publishes it to the /camera/image_raw). At the same time, the /blob_find node subscribes to the same topic and, using OpenCV, identifies the blobs within the image and further publishes the position of the blob in the camera frame and the video feed containing green circles drawn around the identified blobs.

This process can be seen as a schematic in figure 4.7. To be noted that /blob_get_pos is the topic containing the position of each detected blob, whereas /rostopic_11054_1645622339960 represents the recording of each experiment as a rosbag file. Rosbag files document every information obtained by each node for the duration of the recording for further processing. The image shown in figure 4.7 has been processed to show only the blob detection part of the algorithm. In reality, the blob detection is only performed in parallel to SLAM (although the SLAM algorithm can be run by itself). The change was made to better illustrate this specific portion of the algorithm.

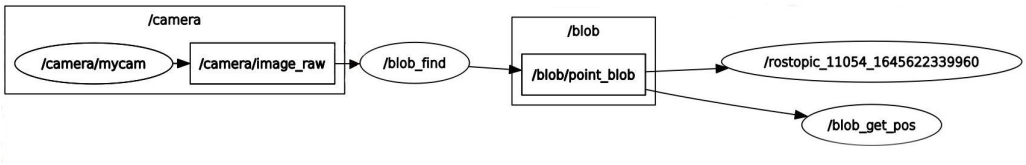


Figure 4.7: ROS schematic showing the connections between different blocks of the blob detection algorithm

4.2.4 Combining ORB-SLAM3, blob detection, and EM tracking

The blob detection section of the algorithm was written in Python, with ROS capabilities. The most important libraries used are OpenCV (a cross-platform library of programming functions mostly aimed at real-time computer vision), NumPy (a collection of mathematic functions), SciPy (scientific and technical computing, more advanced than Numpy), Imutils (basic image processing functions), RosPy (a Python client library for ROS, allowing programmers to communicate with ROS-specific functions), and, finally, libuvc_camera (for the endoscope camera feed).

Both ORB-SLAM3 and its ROS wrapper are written in C++ and they remain, largely, the same way as they can be found on Github.

The way all the nodes coming from the different modules (ORB-SLAM, blob detection, EM tracking) are organized in terms of each other can be seen in figure 4.8. While the SLAM and blob detection portions of the algorithm have been previously explained, this schematic allows a holistic view of the entire process. The new element seen in this figure is the Aurora EM tracker section. Two libraries are used to set up the EM sensor: Plus app 2.8 ('server side') for use on the computer, and the ros_igtl_bridge package for ROS ('client side'), which receives information (in this case, position data) from the PlusApp.

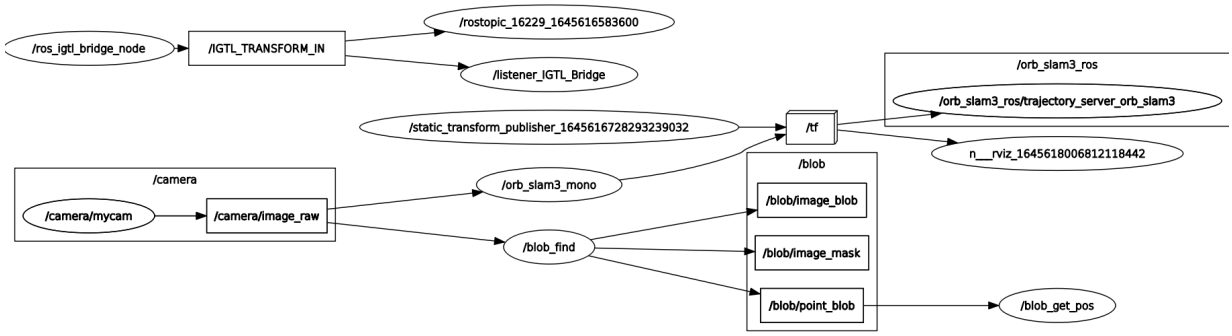


Figure 4.8: All nodes used in this project and the relationships between them

5 Experimental Validation

This part of the report describes the experimental design and setup which will aim to answer the research question, as well as compiles the results obtained following the two experiments. Afterward, the results will be interpreted in preparation for a discussion and extraction of conclusions.

5.1 Experimental Design

The first experiment focuses on testing the accuracy of the end-effector position as obtained by the ORB-SLAM3 algorithm and validated using the Aurora EM tracker.

In order to determine the accuracy of the measurements, the trajectories calculated by the EM tracker and the ORB-SLAM are compared using two scripts: one which will measure the absolute trajectory errors between the two trajectories, and one which will focus on the relative pose error.

The results shed light on which of the textures yields the lowest error. The root-mean-square (RMS) error will be compared to the average speed for each measurement, to see if there is any correlation between the speed of the endoscope and the errors yielded. If the difference is error between two types of texture is negligible, whereas the speed of one is significantly higher, this would lead to a clearer idea of what type of texture is the best suited for endoscopy.

The second experiment will add the element of blob detection. After feeding the endoscope through one of the side gaps of the phantom, the endoscope will continue to advance for a length of around 15 cm (value which represents around 10% of the total length of a human colon) and while doing so it will detect the phantom features (lines, black dots) and map out the trajectory. When a black dot is detected, the position is recorded and the process continues. As before, the trajectory created by the blobs will be compared to the one made by the EM sensor in the form of a RMS error.

A flexible camera equipped with LED lights will act as an endoscope as it is sufficient for the scope of the project. The Aurora EM sensor will be attached to this camera and follow the same trajectory that is calculated by the ORB-SLAM.

5.2 Experimental Setup

For using the EM tracker, the Aurora NDI system has to first be turned on, the sensor has to be plugged in, then the PlusServer Launcher on the computer acting as a server can be connected to the sensor. This will yield an IP address, that has to be introduced in the launch file on the client-side for a complete connection.

Since the Aurora system works on the basis of electromagnetism, it is advised to avoid using any metal objects in its area of influence (a square with an edge of 0.5 m, where the Aurora control unit is in the center).

A photograph illustrating the experimental setup can be seen in figure 5.1.

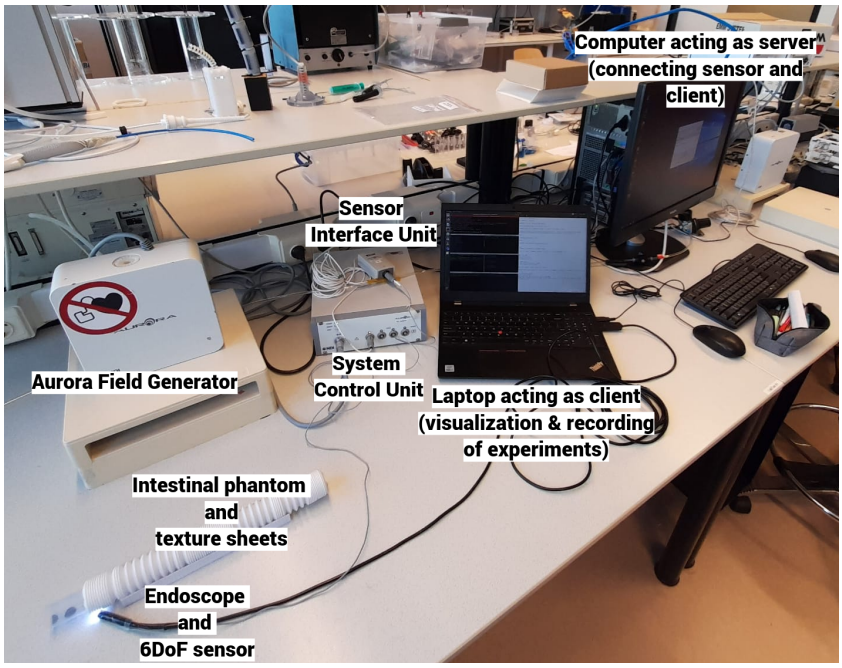


Figure 5.1: Experimental setup

5.3 Experiment protocol

Before performing any analysis on the trajectories, the timestamps of both files are manually inspected to ensure they have the same starting and ending timestamp. This has to be performed as the SLAM algorithms need a few seconds to initialize, depending on the level of detail in each landscape it starts, the time in which no measurements are made. Another reason for this analysis is that the EM tracker performs measurements continuously throughout the experiments, without being turned off between recordings. The starting timestamps on a corresponding pair of files have been chosen to have an offset of no more than 0.1 seconds between themselves.

To be able to perform any error calculations on these trajectories, the timestamps in each pair of files are used to associate each trajectory point, which is performed via singular value decomposition. The aligned trajectories are then compared.

5.4 Data analysis

The main metric used to measure the accuracy of the ORB-SLAM3 trajectory is the RMS error, although other metrics are used as reference as well: the mean, median, minimum and maximum errors, as well as the standard deviation. All these errors respresent absolute trajectory errors (ATE), where ATE refers to measuring the difference between

pairs of points from the true and the estimated trajectory. These errors will be extracted for each of the textures mentioned in section 4.1.2.

During the experiment recording phase, the measurements made by the EM tracker and SLAM, respectively, have been exported to text files following a specific format: *timestamp tx ty tz qx qy qz qw*, where the *timestamp* refers to the time at which a specific measurement was performed, the tx, ty, tz values represent the position of the camera in cartesian coordinates and the qx, qy, qz, qw are quaternions representing its orientation.

The first set of measurements performed referred to the absolute trajectory error (ATE) of each pair of files. The EM-measured file is used as ground truth and the SLAM-measured one as an estimate. A maximum time difference is chosen to help identify the best pairs of equivalent timestamps, in order to synchronize the sampled EM and SLAM estimated data. In this case, two timestamps considered equivalent have a maximum of 0.042 seconds. Another important value is the scale of each set of measurements. While the EM measurements have a consistent scale, it is not the case for SLAM. Since the particular variation of ORB-SLAM3 used in this project is monocular, at the beginning of each experiment the scale is randomly chosen, hence there is a need to correct this in post-processing. The main metric illustrating the differences between the trajectories is the root-mean-square error, following the algorithm described in equation 5.1. The mean, median, minimum, maximum errors, and standard deviation are also extracted.

$$\begin{aligned} \text{align}_{\text{error}} &= \text{EM}_{\text{measurement}} - \text{alignedSLAM}_{\text{measurement}} \\ \text{trans}_{\text{error}} &= \sqrt{\sum \text{align}_{\text{error}}{}^2} \\ \text{RMSE} &= \sqrt{\frac{\text{trans}_{\text{error}}{}^2}{len(\text{trans}_{\text{error}})}}, \end{aligned} \tag{5.1}$$

where $\text{EM}_{\text{measurement}}$ represents each position measured by the EM sensor; $\text{alignedSLAM}_{\text{measurement}}$ is the value of each position identified by the SLAM algorithm, aligned to the timestamps of the EM file; $\text{align}_{\text{error}}$ is the alignment error obtained from the measurement files (EM and SLAM); $\text{trans}_{\text{error}}$ is the translational error; $len(\text{trans}_{\text{error}})$ is the length of the translational error vector, or, more precisely, the number of compared pairs of points; finally, RMSE is the root-mean-square error.

The other type of metric used is the relative pose error (RPE), which measures the relative motion between pairs of values at specific timestamps. After aligning the timestamps again, the errors in relative motion are computed, by means of a moving window of 1 s. Each pose in the trajectory to be verified is associated with a later pose according to this window size. This is useful to estimate the drift in trajectories, an important factor when using monocular visual algorithms.

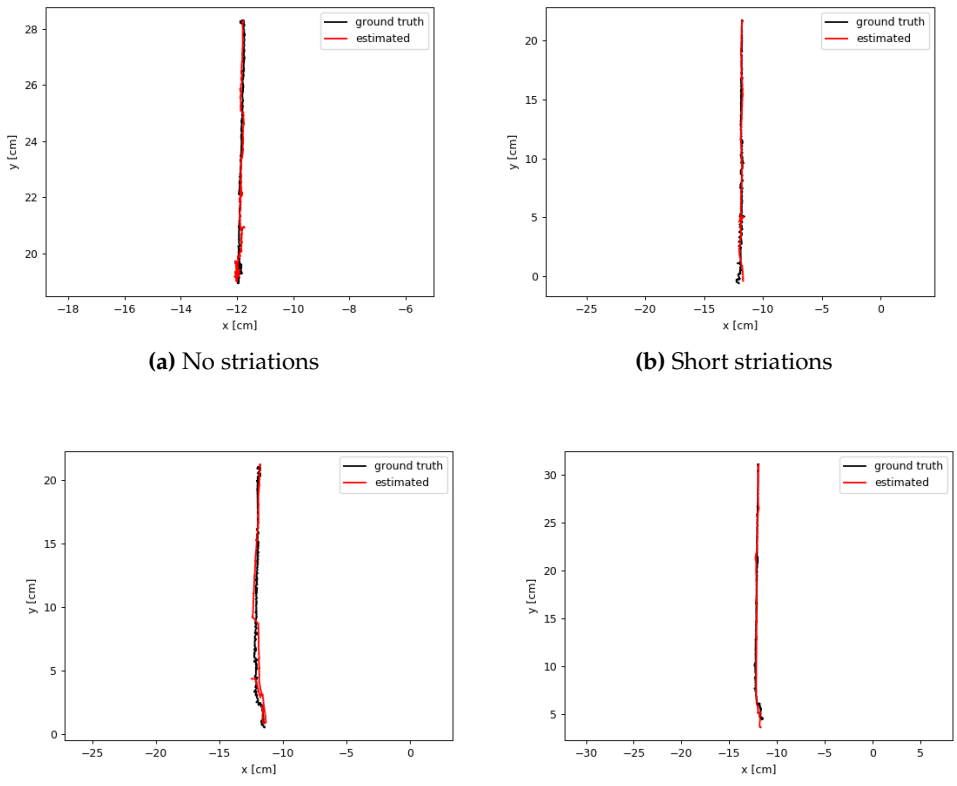
6 Results and discussion

The results of the two experiments will be analyzed in this chapter. These experiments have been performed by pushing the endoscope through a plastic, ribbed tube, which is shaped to form both straight and bent trajectories.

6.1 Experiment 1, localization

6.1.1 Experiment 1, straight trajectory

In figure 6.1 are illustrated the four pairs of trajectories on the straight tube. The experiment made with just the blobs as texture yielded the poorest results, with a trajectory of only a little over 8 cm. This is due to the ORB-SLAM3 algorithm not having enough features to continue to track the endoscope further. The other types of textures provided decent results for the entire length of the desired trajectory.



(a) No striations **(b)** Short striations

(c) Long striations **(d)** Hashed striations

Figure 6.1: Trajectory comparison between ground truth and estimated positions

To be noted that 'poor results' refer, in this case, to the unusably short trajectory that could be tracked with the SLAM algorithm. Further in the chapter, the errors on each trajectory will be taken into account as well.

The absolute trajectory errors calculated for these four scenarios can be seen in figure 6.2. The highest set of errors can be seen in the case of the long striations, which go along the length of the tube, whereas the smallest one is in the case with no striations. However, as mentioned before, the case with no striations didn't have a sufficiently long working trajectory.

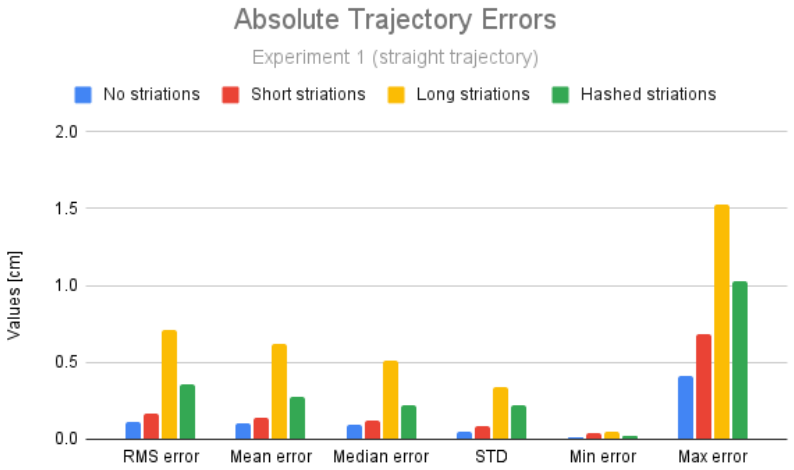


Figure 6.2: Absolute trajectory error for straight trajectory

The errors relative to each texture have no apparent correlation, as neither the environment with the least amount of texture (no striations) or with the most texture (hashed striations) have the highest errors. This is initially surprising, as it was expected that the errors would increase with the amount of texture (the more features can be detected, the more opportunities for errors there are, with the trade-off of a longer trajectory length and a higher velocity).

Observing the number of feature pairs noted in table 6.1, the environment with long striations has by far the highest number of feature pairs, which explains the increased error values. The environments with no striations and short striations have higher numbers of features than the one with hashed striations, despite the latter having, objectively, more texture. This is because the number of feature pairs in the former two skews high due to the first frames, which have an abundance of features. This initial frame with a lot of texture is needed for SLAM initialization.

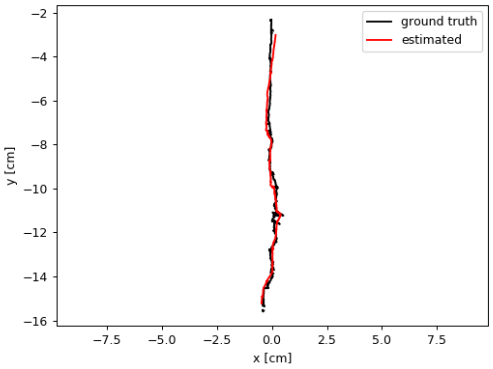
Texture	Feature pairs	RMS error [cm]	Time [s]	Length [cm]	Velocity [cm/s]
No striations	684	0.114307	22	8	0.36
Short striations	830	0.165619	30	22	0.73
Long striations	1358	0.709183	48	20	0.41
Hashed striations	639	0.353284	22	25	1.14

Table 6.1: Comparison of all experiment 1 measurements on a straight trajectory

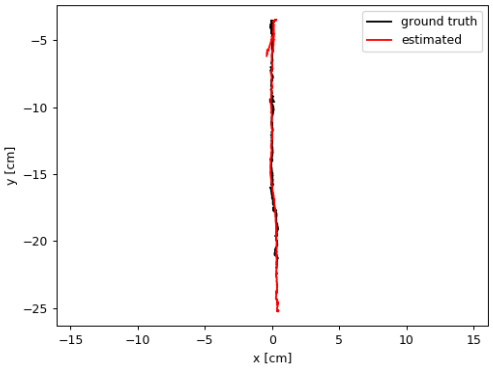
Better indicators of accuracy for a specific texture are the RMS error and the velocity, both of which increase proportionally with the number of feature (no striations < short striations < hashed striations), with the exception of the environment with long striations. After a visual analysis of all measurements discussed thus far, it was concluded that the reason why the long striation environment is such an outlier is most probably due to human error, as the movement is slower than in the other texture environments, despite a high number of detected features. To be noted that the velocity is the mean velocity over the trajectory. As such, it encompasses both smooth and jerky motions which happened during each specific measurement.

6.1.2 Experiment 1, bent trajectory

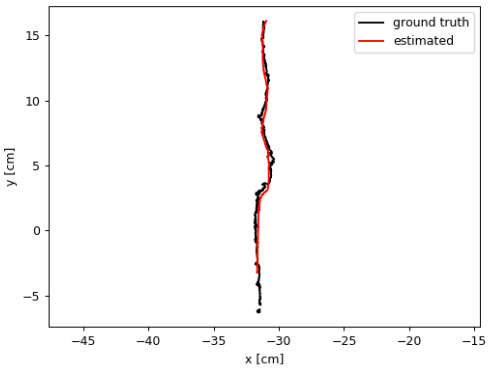
Figure 6.3 shows trajectories for the three cases with striations. The scenario with no striations wasn't attempted anymore as it didn't perform well enough (losing track after a short time) in the previous trial. The angle to which the tube was bent for each of these measurements is between 15°and 45°. This section of the experimentation is also where the endoscope had to be pulled, rather than pushed (in the case of the trajectory with hashed striations), as the creases created in the plastic sheet proved to difficult to navigate otherwise. Figure 4.2 illustrates an example of the bent tube used for this section of experiments.



(a) Short striations



(b) Long striations



(c) Hashed striations

Figure 6.3: Trajectory comparison between ground truth and estimated positions

Figure 6.4 illustrates the absolute trajectory errors for measurements made on bent trajectories. Unlike the trial on straight trajectories, here the environment with hashed striations presents the highest errors. By taking into account the number of features from table 6.2, it can be noted that the number of feature pairs for the hashed striations is significantly higher than for the other two, while having a similar trajectory length and velocity to the environment with long striations.

When looking at the trajectories themselves, illustrated in figure 6.3, it becomes clear that the hashed striations scenario has the most accentuated curve, which both increased the number of features to be detected (due to an increased number of angles caused by the plastic sheet folds), thus increasing the error, and slowed the process down (as the endoscope camera would, at times, have a difficult time advancing through the narrower space).

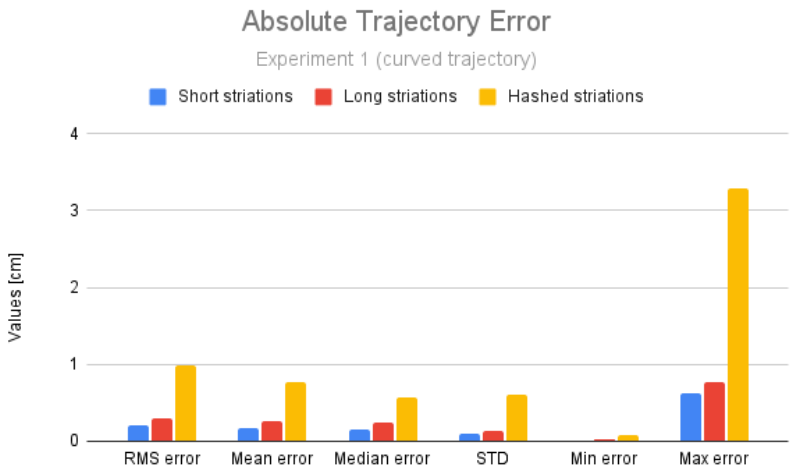


Figure 6.4: Absolute trajectory error for bent trajectory

Texture	Feature pairs	RMS error [cm]	Time [s]	Length [cm]	Velocity [cm/s]
Short striations	869	0.206289	28	14	0.50
Long striations	640	0.302173	49	20	0.41
Hashed striations	1370	0.978403	46	20	0.43

Table 6.2: Comparison of all experiment 1 measurements on a bent trajectory

The long striation scenario has an insignificant curve, with an almost straight trajectory. The phantom itself was bent similarly to the experiment with the short striation, however the curve was along the z-axis, thus not showing on the graph. A screencapture illustrating the curve can be seen in figure 6.5. The curves in the scenarios with long and hashed striations are both more pronounced than the one with short striations, thus decreasing the speed considerably. The short striation scenario also has a shorter trajectory, imposed by loss of texture, due to large dark areas caused by the bending.

Of course, all these bent trajectories presented issues. Due to the relatively rigid nature of the plastic sheets used, the creases formed as result of bending the tube increased the difficulty of endoscopic movement, by creating very small spaces through which the endoscope had trouble navigating smoothly and harsh shadows or bright reflection areas obscuring the camera vision. These issues repeated throughout multiple sets of measurements. In a clinical environment, with softer tissues and less contrast, these issues would be at least partially neutralized. This is why the most telling results for this method are the ones obtained from the straight trajectories.

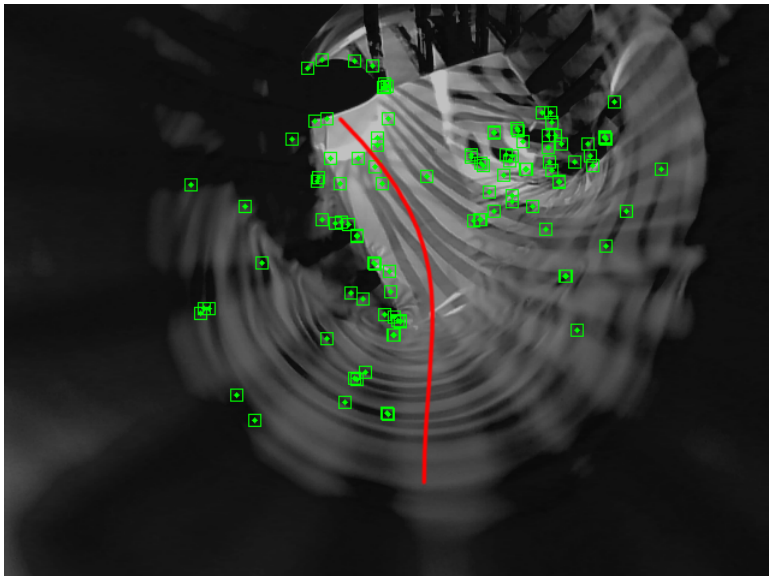
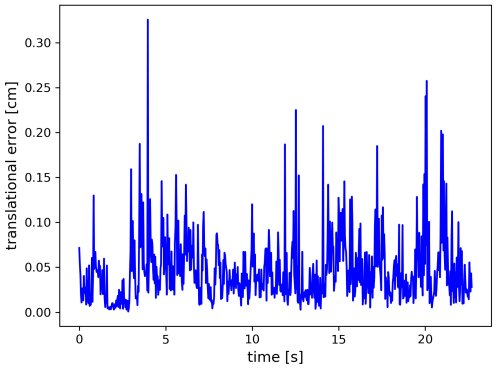


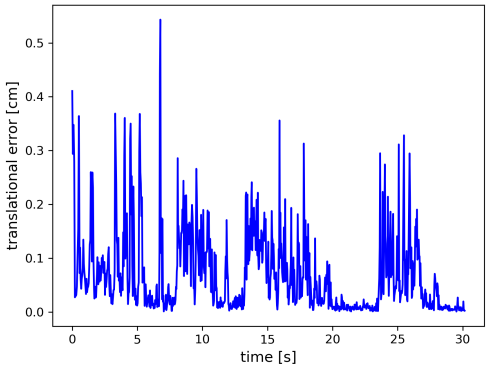
Figure 6.5: Curvature in environment with long striations

6.1.3 Experiment 1, relative pose drift

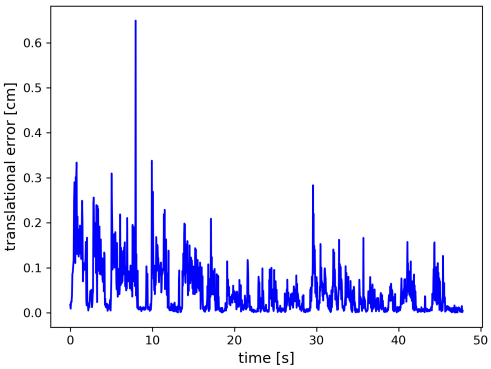
Figures 6.6 and 6.7 illustrate the drift errors for the measurements presented above, for straight and bent trajectories respectively. More specifically, it shows the amount of drift for each timepoint rather than a cumulative drift over time. This drift is calculated in cm per second. To mostly remains below 0.5 cm/s, although there is a case where it spikes to almost 0.8 cm/s, in the case of the straight trajectory with hashed striations, figure 6.6d. On all trajectories, but more clearly on the bent ones, the sudden spikes coincide with abrupt movement, mostly caused by the endoscopic camera tripping over edges while inside the phantom. Most notably, in figure 6.7a, showing the drift for the bent trajectory on the environment with short striations, the two spikes towards the right of the graph represent a moment in which the endoscope camera passed over a large crease, which obscured the remainder of the trajectory, causing loss of vision soon after.



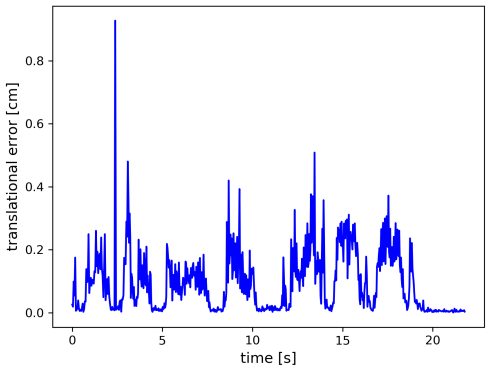
(a) Straight trajectory, no striations



(b) Straight trajectory, short striations

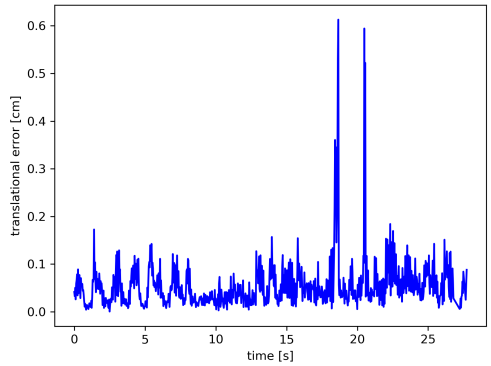


(c) Straight trajectory, long striations

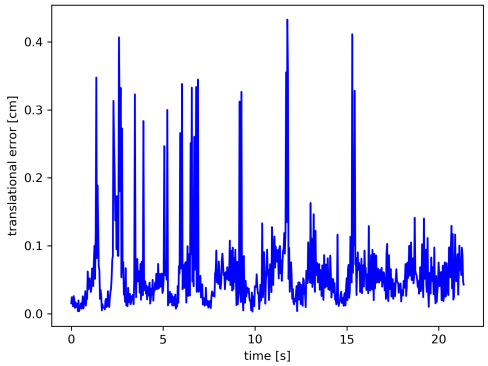


(d) Straight trajectory, hashed trajectory

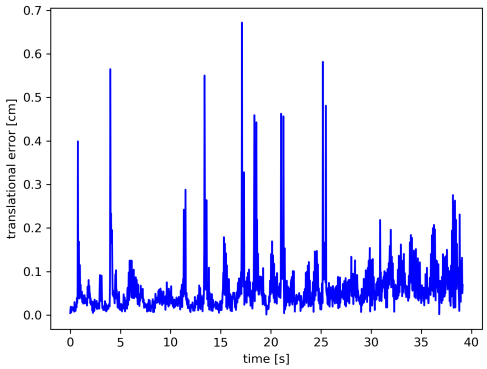
Figure 6.6: Trajectory comparison between ground truth and estimated positions, straight trajectories



(a) Bent trajectory, short striations



(b) Bent trajectory, long striations



(c) Bent trajectory, hashed striations

Figure 6.7: Trajectory comparison between ground truth and estimated positions, bent trajectories

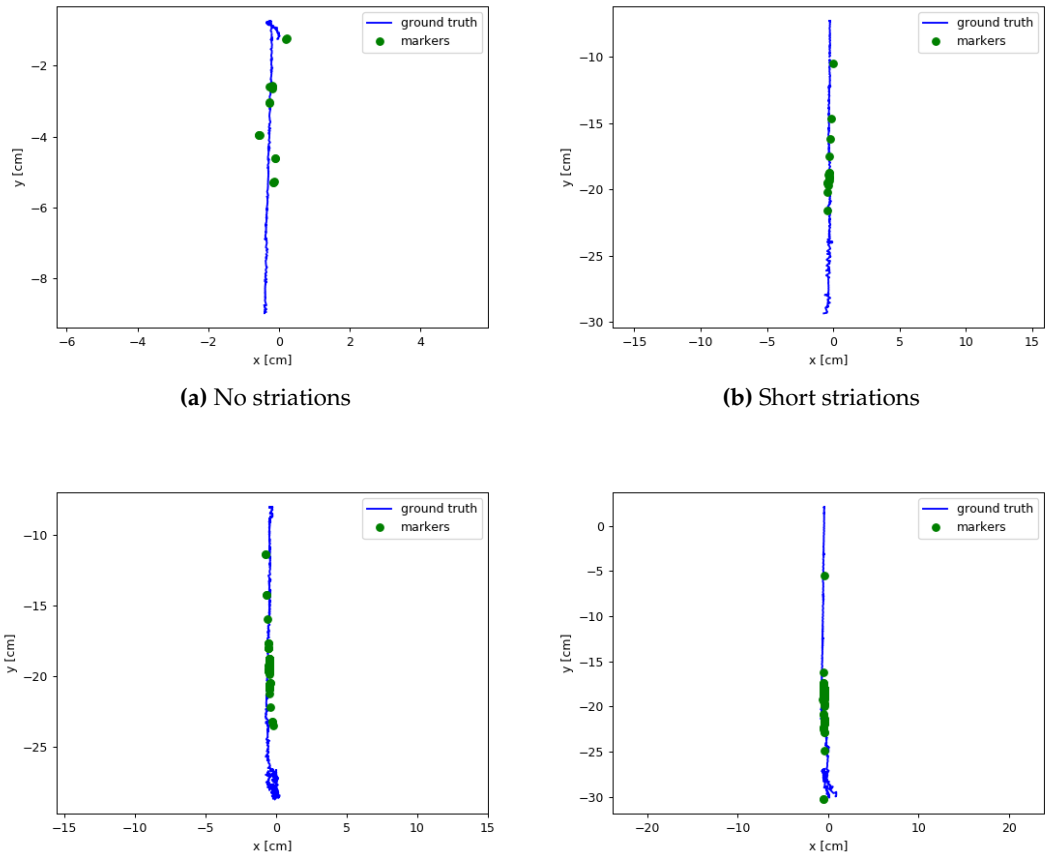
6.2 Experiment 2, haustral folds localization

The main purpose behind the detection and positioning of haustral folds along the trajectory is the creation of a personalized colon map, which the clinicians can use for a more accurate localization of the endoscope with respect to the patient's anatomy.

During the second experiment, it was noticed that the blob detection element of the algorithm was a lot faster to initialize than the ORB-SLAM3 algorithm, and good results obtained from this method could allow for a faster operation time.

However, the results are not as robust as desired. While the markers themselves are correctly identified in most cases, positioning them provided unsatisfactory results. Subfigures 6.8a and 6.8b have the most clear identified markers, with the other two cases, seen in subfigures 6.8c and 6.8d, illustrating large areas of near-constant blob detections. In none of the cases, however, do the positions of the identified blobs coin-

cide with their real positions (as the distance between two subsequent markers is not 2 cm). This is most probably caused by the repeated detection of the same marker as the endoscope progresses through the trajectory.



(a) No striations

(b) Short striations

(c) Long striations

(d) Hashed striations

Figure 6.8: Trajectory comparison between ground truth and estimated blob positions

In figure 6.9 can be seen the error values obtained when comparing the trajectory formed by the marker detection and the ground truth (the trajectory created by the EM-tracker). It can be noted that, unlike the results of the first experiment, the error differences between textures are less prominent and less reliable, without yielding a noticeable trend.

As it currently stands, the spaces between blobs can give clues relating to the speed of the endoscope on sections of the trajectory, where areas with sparse markers were covered in a longer amount of time, while those with a high density of detections correspond to higher velocities. This can also yield some potentially useful conclusions, as the clinicians can estimate their position given the higher or lower identified velocity.

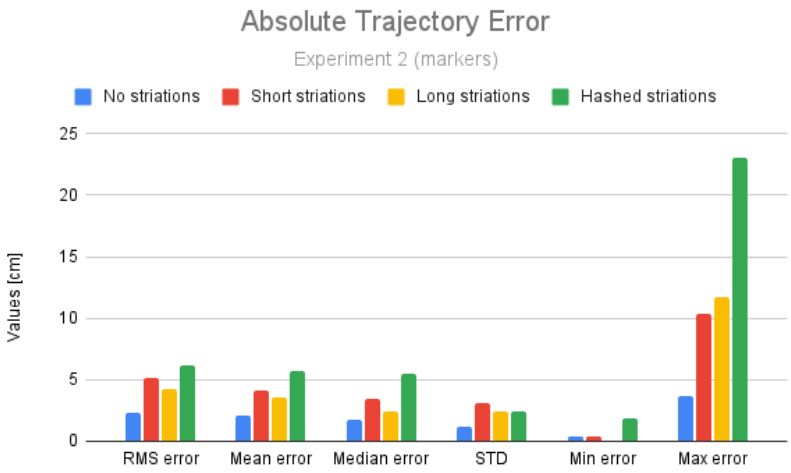


Figure 6.9: Absolute trajectory error for blob positioning

However, without a clear way of quantifying this result, the usefulness of this method as it stands is debatable at best.

6.3 Discussion

The ORB-SLAM3 algorithm performed quite well in three of the four texture situations. While the RMS error of 0.11 cm for the scenario without striations is quite good, the tracking fails after only 8 cm. The other three scenarios have an error of 0.16 cm for the case with short striations, perpendicular to the trajectory path; 0.71 cm for long striations along the trajectory path; and 0.35 for hashed striations. When testing the same textures on a bent trajectory, the error remained below 1 cm, with 0.97 cm for hashed striations, followed by 0.3 cm for long striations and 0.2 for short striations, all managing to reach trajectory lengths of at least 14 cm. Positioning the haustra along the designated trajectory is partly successful, managing to identify the desired indicators in the vast majority of cases, but failing to position them correctly and independently from each other.

7 Conclusions

7.1 Result summary

This project focused on obtaining a robust method of tracking an endoscope through colons of various textures and localizing it with respect to haustra. The algorithm combines the ORB-SLAM3 algorithm for general tracking with blob detection for finding the positions of simplified representations of colon folds along said trajectory. The accuracy of the tracking was tested using an Aurora EM tracker connected to the tip of the endoscope. The experimental results showed that the ORB-SLAM3 algorithm performed reasonably well in three of the four texture situations, with decent results obtained in the case of bent trajectory scenarios. Positioning the haustra along the designated trajectory is largely unsuccessful, managing to identify the spots, but not to position them in a way that can yield useful results.

Unlike endoscope tracking solutions currently in use, the methods explored in this project focus more on abnormal colons, personalizing the tracking to each patient. Their application is also relatively cheap, small, and easy to use as they require just a monocular camera, rather than one of the more expensive options (stereo, RGB-D). ORB-SLAM has proven to yield robust results on both straight and bent trajectories, with the caveat that it takes a few seconds to initialize and sometimes loses track due to lighting conditions or severe lack of texture. Thanks to its relocalization capabilities, however, the second issue can mostly be prevented.

7.2 Limitations

The second part of the project, referring to the positioning of the haustral folds, poses the biggest concerns as it stands. The poor positioning results do not provide useful enough information for this method to have a real-life application as it stands. It does show promise, as it is very fast, and even easier to implement than the SLAM algorithm.

7.3 Future work

A few changes are needed to make the haustral detection and localization more robust and more applicable in a clinical environment. First of all, the issue with the repeated detection of the same marker should be resolved, perhaps by including a search window in the middle of the frame, outside of each the markers are ignored, or by establishing a potential speed of insertion for the endoscope, and keeping just one positional measurement for each time unit. Secondly, the testing environment used for this project can be replaced by a more realistic version, including actual folds instead of the circular markers that have been used so far. Thirdly, to this improved environment can be added lesion and tumor replicas, which can be identified by using machine learning methods.

7.4 Uses

While the project described in this report does not have direct usability in a clinical context, it provides an exploration of the concept for the proposed algorithm. Both the localization and the haustral folds positioning are tested in a simplified colon environment, with an endoscopic camera similar to the one used in clinical practice. The more this concept is developed, as suggested in the future work section, the closer it becomes to a real, clinical application, in which each patient's colon anatomy is used to create personalized maps, useful for further procedures.

A Abbreviations

- **ATE** - Absolute Trajectory Error
- **BGR** - Blue Green Red
- **BRISK** - Binary Robust Invariant Scalable Keypoints
- **DOF** - Degree of Freedom
- **EE** - End-Effector
- **EKF** - Extended Kalman Filter
- **EM** - Electro-Magnetic (when referring to the sensor)
- **EM** - Expectation Maximisation (when referring to the algorithm)
- **FAST** - Features from accelerated segment test
- **FREAK** - Fast Retina Keypoint
- **g2o** - General Graph Optimization
- **GI** - Gastro-intestinal
- **HSV** - Hue Saturation Value
- **ICP** - Iterative Closest Point
- **IEEE** - Institute of Electrical and Electronics Engineers
- **IPC** - Inter-Process Communication
- **KF** - Kalman Filter
- **LBA** - Local Bundle Adjustment
- **LED** - Light Emitting Diode
- **LSD** - Large-Scale Direct monocular
- **MIS** - Minimally Invasive Surgery
- **NDI** - Northern Digital Inc.
- **OA** - Open Appendectomy
- **ORB** - Oriented FAST and Rotated BRIEF
- **OS** - Open Surgery

- **PF** - Particle Filter
- **reloc** - Relocalization
- **RGB-D** - Red Green Blue Depth
- **RMS** - Root Mean Square
- **ROS** - Robot Operating System
- **RPE** - Relative Pose Error
- **SIFT** - Scale-Invariant Feature Transform
- **SLAM** - Simultaneous Localization and Mapping
- **STD** - Standard deviation
- **VCS** - Version Control System

Bibliography

[1] K. Semm, "Endoscopy Appendectomy," 1983.

[2] N. J. Switzer, R. S. Gill, and S. Karmali, "The Evolution of the Appendectomy: From Open to Laparoscopic to Single Incision," *Scientifica*, vol. 2012, pp. 1–5, 2012.

[3] N. J. Soper, P. T. Stockmann, D. L. Dunnegan, and S. W. Ashley, "Laparoscopic Cholecystectomy The New 'Gold Standard'?," *Archives of Surgery*, vol. 127, no. 8, pp. 917–923, 1992.

[4] C. D. Smith, C. J. Weber, and J. R. Amerson, "Laparoscopic adrenalectomy: New gold standard," *World Journal of Surgery*, vol. 23, no. 4, pp. 389–396, 1999.

[5] "Types of minimally invasive surgery (robotic, endoscopic, laparoscopic): Johns hopkins medicine in baltimore, md." https://www.hopkinsmedicine.org/minimally_invasive_robotic_surgery/types.html. Accessed: 2022-04-06.

[6] "Exploring surgery options: open vs. minimally invasive | Beaumont Health." https://www.beaumont.org/health-wellness/blogs/exploring-surgery-options-open-vs-minimally-invasive. Accessed: 2022-04-06.

[7] K. H. Fuchs, "Minimally invasive surgery," *Endoscopy*, vol. 34, no. 2, pp. 154–159, 2002.

[8] B. Mantoğlu, B. Karip, M. Mestan, Y. İşcan, B. Ağca, H. Altun, and K. Memişoğlu, "Should appendectomy be performed laparoscopically? Clinical prospective randomized trial," *Turkish Journal of Surgery*, vol. 31, no. 4, pp. 224–228, 2015.

[9] R. Hargest, "Five thousand years of minimal access surgery: 3000BC to 1850: early instruments for viewing body cavities," *Journal of the Royal Society of Medicine*, vol. 113, no. 12, pp. 491–496, 2020.

[10] S. J. Spaner, B. M. Sc, and G. L. Warnock, "Endoscopy, Laparoscopy,," vol. 7, no. 6, pp. 369–373, 1997.

[11] "ENT rigid and flexible endoscopes - Happersberger Otopront." https://www.otopront.de/en/products/microscopy-and-optical-systems/rigid-and-flexible-endoscopes.html. Accessed: 2022-04-06.

[12] R. Wientjes, H. J. Noordmans, J. A. d. Eijk, and H. v. d. Brink, "Automated Objective Routine Examination of Optical Quality of Rigid Endoscopes in a Clinical Setting," *PLoS ONE*, vol. 8, no. 3, pp. 4–11, 2013.

[13] M. Liedlgruber and A. Uhl, "A Summary of Research Targeted at Computer-Aided Decision Support in Endoscopy of the Gastrointestinal Tract," *Department of Computer Sciences, University of Salzburg, Austria.*, vol. 1, no. April 2011, p. 61, 2011.

[14] R. Shumway and J. D. Broussard, "Maintenance of gastrointestinal endoscopes," *Clinical Techniques in Small Animal Practice*, vol. 18, pp. 254–261, 11 2003.

[15] A. Loeve, P. Breedveld, and J. Dankelman, "Scopes too flexible and too stiff," *IEEE Pulse*, vol. 1, no. 3, pp. 26–41, 2010.

[16] C. C. Tseng, T. Y. Hsiao, and W. C. Hsu, "Comparison of rigid and flexible endoscopy for removing esophageal foreign bodies in an emergency," *Journal of the Formosan Medical Association*, vol. 115, pp. 639–644, 8 2016.

[17] D. Ferrari, A. Aiolfi, G. Bonitta, C. G. Riva, E. Rausa, S. Siboni, F. Toti, and L. Bonavina, "Flexible versus rigid endoscopy in the management of esophageal foreign body impaction: Systematic review and meta-analysis," *World Journal of Emergency Surgery*, vol. 13, no. 1, pp. 1–9, 2018.

[18] H. C. Cheung, K. F. Siu, and J. Wong, "A Comparison of flexible and rigid endoscopy in evaluating esophageal cancer patients for surgery," *World Journal of Surgery 1988 12:1*, vol. 12, pp. 117–121, 2 1988.

[19] G. Unfried, F. Wieser, A. Albrecht, A. Kaider, and F. Nagele, "Flexible versus rigid endoscopes for outpatient hysteroscopy: A prospective randomized clinical trial," *Human Reproduction*, vol. 16, no. 1, pp. 168–171, 2001.

[20] J. Roth and S. Constantini, "Combined rigid and flexible endoscopy for tumors in the posterior third ventricle," *Journal of Neurosurgery*, vol. 122, pp. 1341–1346, 6 2015.

[21] Y. X. Mak, A. Lanciano, S. Stramigioli, and M. Abayazid, "Development of Haptic Approaches for a Head-Controlled Soft Robotic Endoscope," *Proceedings of the IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics*, vol. 2020-Novem, pp. 1216–1222, 2020.

[22] N. Omidbakhsh, S. Manohar, R. Vu, and K. Nowruzi, "Flexible gastrointestinal endoscope processing challenges, current issues and future perspectives," *Journal of Hospital Infection*, vol. 110, pp. 133–138, 4 2021.

[23] N. Kurniawan and M. Keuchel, "Flexible Gastro-intestinal Endoscopy — Clinical Challenges and Technical Achievements," *Computational and Structural Biotechnology Journal*, vol. 15, pp. 168–179, 1 2017.

[24] J. Lee, A. Voytovich, W. Pennoyer, K. Thurston, and R. A. Kozol, "Accuracy of colon tumor localization: Computed tomography scanning as a complement to colonoscopy," *World J Gastrointest Surg.*, vol. 2(1), pp. 22–25, Jan 2010.

[25] M. N. Zikusoka and J. H. Kwon, "Colonoscopy and Flexible Sigmoidoscopy in Colorectal Cancer Screening and Surveillance," *Early Diagnosis and Treatment of Cancer Series: Colorectal Cancer*, pp. 83–92, 1 2011.

[26] F. A. Farraye, "How to use chromoendoscopy and high definition white light endoscopy - YouTube." https://www.youtube.com/watch?v=h5saIczgCOM&t=684s, note = Accessed: 2022-04-06.

[27] G. Lichtenstein, "Choice of Approaches for the Optimal Performance of Chromoendoscopy - YouTube." https://www.youtube.com/watch?v=xEmxSX5893k. Accessed: 2022-04-06.

[28] "Interview with clinical experts." personal discussion with drs. Milou van Riswijk. Day of meeting: 2021-08-25.

[29] "Congenital malformations of the gastrointestinal tract." https://www.aboutkidshealth.ca/Article?contentid=1770&language=English. Accessed: 2022-04-06.

[30] "Endoscopy nurses technicians: Keys to colonoscopy success." https://www.colowrap.com/blog/eaocrc-0. Accessed: 2022-04-06.

[31] "Gastrointestinal tract 5: the anatomy and functions of the large intestine | nursing times." https://www.nursingtimes.net/clinical-archive/gastroenterology/gastrointestinal-tract-5-anatomy-functions-large-intestine-23-09-2019/. Accessed: 2022-04-06.

[32] G. A. Formosa, J. M. Prendergast, J. Peng, D. Kirkpatrick, and M. E. Rentschler, "A modular endoscopy simulation apparatus (mesa) for robotic medical device sensing and control validation," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4054–4061, 2018.

[33] C. Xie, T. Yao, J. Wang, and Q. Liu, "Endoscope localization and gastrointestinal feature map construction based on monocular SLAM technology," *Journal of Infection and Public Health*, vol. 13, pp. 1314–1321, 9 2020.

[34] M. Wang, Q. Shi, S. Song, C. Hu, and M. Q.-H. Meng, "A Novel Relative Position Estimation Method for Capsule Robot Moving in Gastrointestinal Tract," *Sensors (Basel, Switzerland)*, vol. 19, 6 2019.

[35] X. Peng, T. Xie, Y. Hao, and P. Chen, "Wearable Electromagnetic Apparatus for Endoscope Localization:," *https://doi.org/10.1016/j.jala.2010.09.001*, vol. 16, pp. 366–370, 10 2011.

[36] S. Hayashi, M. Takenaka, M. Hosono, and T. Nishida, "Radiation exposure during image-guided endoscopic procedures: the next quality indicator for endoscopic retrograde cholangiopancreatography," *World Journal of Clinical Cases*, vol. 6, no. 16, p. 1087, 2018.

[37] B. F. Cox, F. Stewart, H. Lay, G. Cummins, I. P. Newton, M. P. Y. Desmulliez, R. J. C. Steele, I. Näthke, and S. Cochran, "Perspective on Capsule Endoscopy Ultrasound capsule endoscopy: sounding out the future," *Ann Transl Med*, vol. 5, no. 9, p. 201, 2017.

[38] S. D. Pendleton, H. Andersen, X. Du, X. Shen, M. Meghjani, Y. H. Eng, D. Rus, and M. H. Ang, "Perception, planning, control, and coordination for autonomous vehicles," *Machines*, vol. 5, no. 1, pp. 1–54, 2017.

[39] H. Strasdat, J. M. Montiel, and A. J. Davison, "Real-time monocular SLAM: Why filter?," *Proceedings - IEEE International Conference on Robotics and Automation*, pp. 2657–2664, 2010.

[40] H. Yin, Z. Ma, M. Zhong, K. Wu, Y. Wei, J. Guo, and B. Huang, "SLAM-based self-calibration of a binocular stereo vision rig in real-time," *Sensors (Switzerland)*, vol. 20, no. 3, 2020.

[41] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the RGB-D SLAM system," *Proceedings - IEEE International Conference on Robotics and Automation*, vol. 3, no. c, pp. 1691–1696, 2012.

[42] J. Engel, T. Schöps, and D. Cremers, "LSD-SLAM: Large-Scale Direct Monocular SLAM," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8690 LNCS, no. PART 2, pp. 834–849, 2014.

[43] R. Mur-Artal, J. M. Montiel, and J. D. Tardos, "ORB-SLAM: A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[44] S. Huang and G. Dissanayake, "Convergence and consistency analysis for extended Kalman filter based SLAM," *IEEE Transactions on Robotics*, vol. 23, no. 5, pp. 1036–1049, 2007.

[45] A. Karlsson, J. Bjärkefur, J. Rydell, and C. Grönwall, "Smoothing-Based Submap Merging in Large Area SLAM," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 6688 LNCS, pp. 134–145, 2011.

[46] P. H. Torr and A. Zisserman, "Feature based methods for structure and motion estimation," in *International workshop on vision algorithms*, pp. 278–294, Springer, 1999.

[47] B. Triggs, P. F. McLauchlan, R. I. Hartley, and A. W. Fitzgibbon, "Bundle adjustment—a modern synthesis," in *International workshop on vision algorithms*, pp. 298–372, Springer, 1999.

[48] K. Levenberg, "A method for the solution of certain non-linear problems in least squares," *Quarterly of applied mathematics*, vol. 2, no. 2, pp. 164–168, 1944.

[49] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard, "g 2 o: A general framework for graph optimization," in *2011 IEEE International Conference on Robotics and Automation*, pp. 3607–3613, IEEE, 2011.

[50] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International conference on computer vision*, pp. 2564–2571, Ieee, 2011.

[51] D. Galvez-López and J. D. Tardos, "Bags of binary words for fast place recognition in image sequences," *IEEE Transactions on Robotics*, vol. 28, no. 5, pp. 1188–1197, 2012.

[52] H. Strasdat, J. Montiel, and A. J. Davison, "Scale drift-aware large scale monocular slam," *Robotics: Science and Systems VI*, vol. 2, no. 3, p. 7, 2010.

[53] O. Faugeras and F. Lustman, "Motion and structure from motion in a piecewise planar environment," *International Journal of Pattern Recognition and Artificial Intelligence - IJPRAI*, vol. 02, 09 1988.

[54] R. Mur-Artal and J. D. Tardos, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Transactions on Robotics*, vol. 33, p. 1255–1262, Oct 2017.

[55] C. Campos, R. Elvira, J. J. G. Rodriguez, J. M. Montiel, and J. D. Tardos, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Transactions on Robotics*, vol. 37, p. 1874–1890, Dec 2021.

[56] "Ros basic concepts." http://wiki.ros.org/ROS/Concepts. Accessed: 2022-04-06.

[57] "Pinhole camera model." https://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html. Accessed: 2022-04-25.

[58] "Endoscope camera depstech cmos hd usb endoscope waterproof test comparison." https://test-vergleiche.com/en/product/depstech/endoscope-camera-depstech-waterproof/. Accessed: 2022-04-06.

[59] G. Andria, F. Attivissimo, A. Di Nisio, A. M. L. Lanzolla, and M. A. Ragolia, "Assessment of position repeatability error in an electromagnetic tracking system for surgical navigation," *Sensors*, vol. 20, no. 4, 2020.