

Universitat Politècnica de Catalunya
Facultat d'Informàtica de Barcelona (FIB)



UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Diagnostic imaging for tracheobronchomalacia patients

Miquel Prieto Moliné

Computing Specialization

Bachelor's Degree Project Director: Joan Climent Vilaró

GEP Tutor: Paola Lorenza Pinto

March 19th, 2022

Abstract

The computer vision field is gaining, every year, more and more importance across the whole world, from being used in order to create self-driving cars, to helping medics accomplishing difficult and arduous tasks. One of these tasks is to detect if a patient suffers from tracheobronchomalacia, a condition which causes the walls of the trachea and bronchi to collapse when exhaling.

This project aims to give an estimation on the degree of aperture of the airways for people suffering from tracheobronchomalacia, and healthy users, and in so giving a diagnosis for the disease using existing or new segmentation algorithms, and features gathered from the various frames of the patients' bronchoscopies.

We will use various segmentation algorithms based on regional, edge and binarization segmentation, together with information gathered during the process, in order to calculate and evaluate the evolution of the stenosis degree of the trachea.

Resumen

El campo de la visión por computador va adquiriendo cada año más y más importancia en todo el mundo, desde su uso para crear coches sin piloto, hasta ayudar a profesionales médicos en la realización de tareas difíciles y arduas. Una de estas tareas es detectar si un paciente padece de traqueobroncomalacia, una enfermedad que hace que las paredes de la tráquea y los bronquios se colapsen al exhalar aire.

Este proyecto pretende dar una estimación sobre el grado de apertura de las vías respiratorias de las personas que padecen traqueobroncomalacia, y de los usuarios sanos, y así dar un diagnóstico para la enfermedad utilizando algoritmos de segmentación existentes o creando de nuevos, y usando las características recogidas de los distintos fotogramas de las broncoscopias de los pacientes.

Utilizaremos varios algoritmos de segmentación basados en la segmentación regional, de bordes y de binarizado, junto con la información recogida durante el proceso, para calcular y evaluar la evolución del grado de estenosis de la tráquea.

Resum

El camp de la visió per computador va adquirint cada any més i més importància en tot el món, des del seu ús per a crear cotxes no tripulats, fins a ajudar a professionals mèdics en la realització de tasques difícils i dures. Una d'aquestes tasques consisteix en detectar si un pacient pateix de traqueobroncomalacia, una malaltia que fa que les parets de la tràquea i els bronquis es col·lapsin al exhalar.

Aquest projecte pretén donar una estimació en el nivell de la obertura de les vies respiratòries de la persona que pateix traqueobroncomalacia, i dels pacients sans, i així donar un diagnòstic per a la malaltia utilitzant algorismes de segmentació existents o creant-ne de nous, i usant les característiques obtingudes dels diversos fotogrames de les broncoscòpies dels pacients.

S'usaran varis algorismes de segmentació basats en la segmentació per regions, per bores i la segmentació per binarització, juntament amb la informació obtinguda durant el procés, per calcular i avaluar la evolució del grau de estenosi de la tràquea.

CONTENTS

Contextualization and scope	7
1.1 Context	7
1.1.1 Introduction	7
1.1.2 Problem to be resolved	8
1.1.3 Stakeholders	9
1.2 Justification	10
1.2.1 Previous studies.....	10
1.2.2 Why this project?.....	10
1.3 Scope.....	11
1.3.1 Objectives and sub-objectives.....	11
1.3.2 Obstacles and risks.....	12
1.4 Methodology and rigor.....	12
1.4.1 Methodology	12
1.4.2 Development and monitoring tools	13
2 Project Planning.....	14
2.1 Task definition.....	14
2.1.1 Summary table and estimations	16
2.2 Resources used	18
2.2.1 Human resources.....	18
2.2.2 Software resources	18
2.2.3 Hardware resources.....	18
2.3 Gantt chart	19
2.4 Risk management: alternative plans and obstacles.....	20
2.5 Laws and regulation	20
3 Budget and sustainability	21
3.1 Budget	21
3.1.1 Direct costs	21
3.1.2 Indirect costs.....	23
3.1.3 Unexpected costs.....	24
3.1.4 Total Budget.....	25
3.2 Management and budgetary control.....	25
3.3 Cost changes	26
3.4 Sustainability.....	26
3.4.1 Self-assessment	26

CONTEXTUALIZATION AND SCOPE

3.4.2	Economic dimension	27
3.4.3	Environmental dimension	28
3.4.4	Social dimension	29
4	Development	31
4.1	Introduction	31
4.2	Gathering of frames	32
4.3	Treatment of frames	32
4.3.1	Cropping and preprocessing	33
4.3.2	Segmentation algorithms.....	35
4.3.3	Calculating the final blob	44
4.3.4	Gathering information from the final blob.....	45
4.3.5	Computation of the results	46
4.4	Creating the results video.....	48
5	Experiments and performance	49
5.1	Binarization based segmentation algorithms and best features	49
5.1.1	Segmentation comparison.....	49
5.1.2	Chunks of videos and performance	53
5.2	Region based segmentation algorithm	59
5.2.1	Feature comparison	59
5.2.2	Chunks of videos and performance	62
5.3	Edge based segmentation algorithm	66
5.3.1	Segmentation results	66
5.3.2	Chunks of videos and performance	69
5.4	Mix bag segmentation	73
5.4.1	Chunks of videos and performance	73
6	Conclusions and future work	78
6.1	Conclusions	78
6.2	Future work	78
	Bibliography	79
	Appendix	82
	Get Frames script	82
	Main Program script.....	83
	Get Video script	108

CONTEXTUALIZATION AND SCOPE

1.1 CONTEXT

This Bachelor Thesis Degree is delimited inside the scope of the Faculty of Informatics of Barcelona, specialization in Computing, and it is directed by Joan Climent Vilaró.

1.1.1 Introduction

Nowadays, computer vision has a big impact in healthcare applications due to the progress made using image processing and pattern recognition algorithms during the past several decades. Thanks to this, computer vision is being used to assist medical professionals in making better decisions regarding the treatment of patients [1].

Computer vision is a field of artificial intelligence that enables computers and systems to drive meaningful information from digital images, videos and other visual inputs – and take actions or make recommendations based on that information, we could say computer vision enables computers to think based on what they “see”, and it has been around for about 60 years. It began in 1959 when neurophysiologists showed a cat an array of images, attempting to correlate a response in its brain. They discovered that it responded first to hard edges or lines, and scientifically, this meant that image processing starts with simple shapes like straight edges.

At about the same time, the first computer image scanning technology was developed, enabling computers to digitize and acquire images. Another milestone was reached in 1963 when computers were able to transform two-dimensional images into three-dimensional forms. In the 1960s, AI emerged as an academic field of study, and it also marked the beginning of the AI quest to solve the human vision problem. 1974 saw the introduction of optical character recognition (OCR) technology, which could recognize text printed in any font or typeface. Similarly, intelligent character recognition (ICR) could decipher hand-written text using neural networks. Since then, OCR and ICR have found their way into document and invoice processing, vehicle plate recognition, mobile payments, machine translation and other common applications [2].

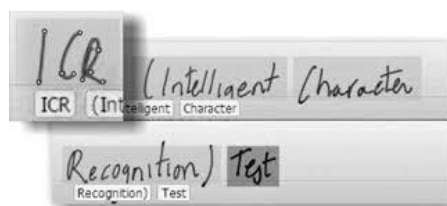


Figure 1: Example of usage of intelligent character recognition on human writing. Source: Google

Computer vision has kept on growing until today, giving a helping hand to humanity by solving a vast number of problems and making arduous and difficult tasks easier.

This project aims to find a way of giving a positive diagnosis, given a bronchoscopy (a video of the airways of a patient), for patients suffering tracheobronchomalacia (TBM), a condition caused by a weakness on the walls of the trachea and bronchi that makes it difficult to breathe due to the collapsing of the airway when the patient breathes out. This condition can be caused by congenital conditions or by trauma (due to prolonged intubation or tracheal surgery for example), and it can be presented either at birth or during adulthood, causing coughs, shortness in breath and and/or recurrent tracheal and bronchi infections [3].

There are studies done to give an endoscopic estimation of the degree of stenosis (narrowing of the trachea due to an injury or a congenital anomaly) in the central airway [4], but there are none about giving a diagnosis for tracheobronchomalacia, where the stenosis of the airway varies depending if the patient is breathing in or out. To give such a diagnosis, this project aims to use the knowledge acquired during my years as a student of computer science at FIB to find such a capable algorithm, mainly using computer vision.

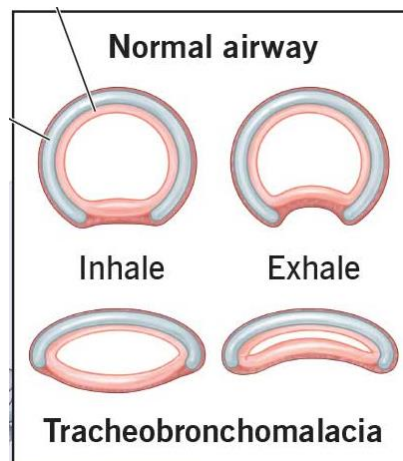


Figure 2: Comparison between a normal airway (up), and a tracheobronchomalacia patient. Source: Cleveland Clinic

1.1.2 Problem to be resolved

As exposed previously, there are studies done to give an estimation on the degree of stenosis in the central airway using manually selected frames from a bronchoscopy, others that allow to measure the size of the diameter of the airway tracks using triangulation principles with a new type of bronchoscope [5], but it obviously requires the usage of a new and dedicated bronchoscope, and even then, it would only prove useful to measure airway dimensions that doesn't vary (or vary very little) in time, or to measure lesion sizes.

Due to the fact that tracheobronchomalacia makes the walls of the trachea and bronchi so weak, we have to take in account the degree of aperture left once the patient breathes out and compare it with when he/she breathes in, so we can evaluate the area left and give a positive or negative diagnosis for this disease.

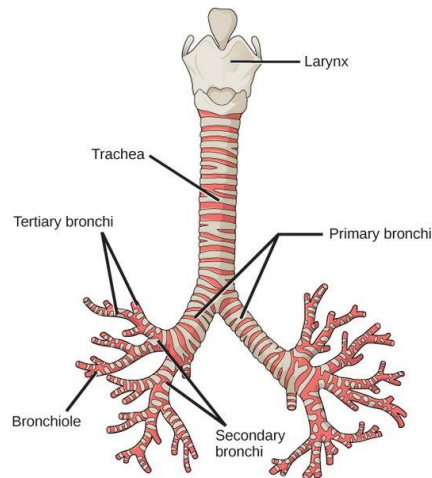


Figure 3: Image showing the trachea and the bronchi. Source: pediaa.com

1.1.3 Stakeholders

In this section, we are going to discuss the different actors involved. This project has four main involved parties, which can be grouped in two different classes depending on the interaction and benefits they have with the project itself.

First of all, there are the directly involved stakeholders with the project, which are:

- **Project director**

The project director Joan Climent Vilaró, expert on computer vision, image processing, pattern recognition, mathematical morphology and embedded systems, whose idea was the base for this project. He is in charge of leading the developer in case of any difficulty as well as advising him.

- **Developer**

The developer Miquel Prieto Moliné, is in charge of the planning, the documentation, information search, development of the algorithms, performance and analysis of the experiments done throughout the project and the solution of problems that may arise during the development, as well as being in charge of fulfilling the terms established during this project.

In second hand, we have the indirectly involved stakeholders, which are:

- **Benefited users**

Even though this project doesn't have the intention to create a product on this own, and its scope is limited to the patient samples given by the hospital (there are a lot of bureaucracies and limitations regarding the gathering of video samples and any other kind of material referring to hospital patients). Investigating this matter and offering a possible prototype could prove useful for other researches in order to make a plausible solution, which in return would prove useful to medical professionals and patients.

1.2 JUSTIFICATION

1.2.1 Previous studies

As pointed out in point 1.1.2, there are multiple studies done throughout the years that try to evaluate the stenosis degree or to quantify the degree of obstruction of the central airway.

In the previously mentioned study of "*Objective Endoscopic Measurements of Central Airway Stenosis: A Pilot Study*" [4], researchers found a way to give a SI (stenosis index), quasi in real time, using an image-based computational system, developed by them, that calculates de CSA (cross-section area) of the airway track. Even though this system is used to estimate the general SI of patients that have a different ailment than tracheobronchomalacia, we take inspiration in their usage of lumen and tracheal ring regions, that vary due to breathing, to estimate the SI between exhaling and inhaling, and thus giving a possible diagnosis for tracheobronchomalacia.

Other studies like "*New method for quantitative assessment of airway caliber*" [5] as indicated before and "*Quantifying Central Airway Obstruction during Therapeutic Bronchoscopy*" [6], give accurate diameter estimations of the airways system by using dedicated hardware developed expressively to commit such task (and with medical professionals' intervention). Given that this project only has information gathered given a bronchoscopy of a patient suffering from tracheobronchomalacia and a bronchoscopy of healthy person of a patient, both of these studies prove no use to us, nonetheless, as the study referred in the previous paragraph, we take inspiration in the usage of the airway lumen area to calculate the opening area of the airways.

1.2.2 Why this project?

As stipulated before, all the previously referred studies give a way to calculate the stenosis index or give accurate estimations of the diameter of the airway using dedicated and expensive hardware or software, some needing the intervention of medical professionals. Due to the previously mentioned fact that we only have two bronchoscopy videos of a patient suffering from tracheobronchomalacia and a video of a healthy patient, both filmed with normal endoscopes, and that this project

aims to give a diagnosis for the disease based on these videos, a different approach than the one showed in the previous studies is needed.

1.3 SCOPE

1.3.1 Objectives and sub-objectives

As mentioned, during the justification, the main objective of this project is to give a diagnosis for tracheobronchomalacia using computer vision algorithms given a video of a bronchoscopy.

To accomplish this, the project has the following main and secondary objectives:

Main Objectives

There are a few principal objectives

- Obtain each frame of the video and segmentate in two parts so the area of aperture is separated from the walls of the airway.
 - Try different features (redness of the image, saturation, value ...) until finding the best settings applicable for the segmentation.
 - Use different segmentation algorithms: binarization, region-based segmentation or contour-based segmentation.
- Try to use light intensity in order to detect drastic changes in the airway stenosis.
- Compare which of the both previous objectives are better or try to merge them in order to improve diagnosis.
- Take in account previous frames of the video (time) in order to correctly segmentate and assure diagnosis correctness.
- Gather the information of the previous objectives in order to give the most accurate possible diagnosis.

Secondary Objectives

These are the secondary, but not less important, objectives:

- Implement all the aspects of the algorithm in the most efficient way possible.
- Find information about segmentation and tracking techniques to improve any aspect of the program.

- Being able to detect frames that don't provide any valid information for the computation of the diagnosis.

1.3.2 Obstacles and risks

Some obstacles may appear during the development of this project, alongside with some risks.

- As mentioned in the previous point, during the bronchoscopy there may be some occlusion caused by humidity of the airways, causing blurriness or making those frames unusable.
- Erratic movement of the endoscope, not showing a clear image of the airway tracks and approaching the walls of the trachea and bronchi too much.
- Time consuming due to the image processing algorithms used. The algorithms used have to be as efficient as possible in order to not take too long processing the frames of the bronchoscopies.
- Managing time in order to make it to the deadlines. There are tight deadlines for the project, and we have to take in account that I, as the developer, work a normal job 43 hours a week, plus I'm enrolled in two other subjects during this quarter. So, in order to meet the deadlines, a good planning is needed to finish the project.

1.4 METHODOLOGY AND RIGOR

In this section, we are going to look at the methodology and the set of tools that will be used during the development of the project.

1.4.1 Methodology

In order to meet the deadlines, we a combination of Kanban and Agile work methodology is applied, where the development of the project will be sub-divided in small tasks (cards). During each sprint of the development, a set of tasks are focused on, and then if needed, a meeting with the director takes place to get feedback on the work done and, if the situation arise, the addition of new possible tasks to the Kanban board.

The Kanban board will be divided in 3 different sections, the first one is going to be dedicated to the "To Do" tasks, where all the tasks pending to be developed for each sprint are going to be posted. Once work starts in one of the tasks, it will be

moved to the “Development” section, where all the tasks going under development are posted. And finally, every time a task is completed, it will be posted in the “Done” section.

1.4.2 Development and monitoring tools

All the programming of this project is done using MATLAB®, with libraries destined to image processing and computer vision.

MATLAB is a programming platform designed specifically for engineers and scientists to analyze and design systems and products, fairly used during the computers engineering studies at FIB. The heart of MATLAB is the MATLAB language, a matrix-based language allowing the most natural expression of computational mathematics [7].

In order to keep track of all the tasks to be done, Trello is used as a Kanban system, and for source control and version tracking of the code, GitHub.

2 PROJECT PLANNING

This chapter deals with the project's planning describing the tasks performed, through and action plan, to meet the deadlines. It also presents the resources used and analyses the possible obstacles that could modify its planning.

This project began during the third week of February of 2022 and the developer will present it in June 2022. With an estimation of 28 hours per week, (10 hours in total from Monday to Friday, and 18 during the weekends), keeping in mind that the distribution of hours during the week fluctuate due to the other 2 subjects that the developer is attending to, and his full-time job, the total estimated time is of (giving 18 weeks of work) $28h * 18weeks = 504$ hours. Later in *section 2.1.1* of the document, we will see a more detailed explanation of the time estimations.

The planification of the project has suffered some changes since its beginning, not so much on the total amount of hours spent on it, but the number of hours spent on different parts of the development and the timings of each and every tasks.

2.1 TASK DEFINITION

In this section, all the tasks carried out along the project are described.

While the first two main tasks of this project have suffered no changes, the development task has suffered some. In this section I'll state all the main tasks done throughout this project and its subtasks.

T1. Study of concepts and research

Before starting the development of this project, it is necessary to gather and become familiar with basic concepts relatives to the subject, referring to the medical and technological terminology. Alongside with this, constant research about possible problems that may appear or information needed to improve and continue the development is needed.

T2. Project Planning

This task refers to the content covered by the GEP course. It can be split into four sub-tasks:

1. **Context and scope of the project**
2. **Project planning**
3. **Budget and sustainability**
4. **Final project definition**

System configuration

Before starting the development, it is crucial to configure all the necessary tools correctly. This task is included inside the study of concepts and research task.

In order to develop this project, it is necessary to have a stable version of MATLAB installed, alongside with some add-ons (e.g., Computer Vision toolbox and Image Processing Toolbox) that may prove useful during the development.

For the source control, as noted during the *Contextualization and scope* chapter, it is necessary to set up Git.

T3. Development

This is the most important task of the project and as stated before it's the only one that has suffered changes since its beginning. It includes the development of the program, gathering of the results and documentation. This phase can be split in the following sub-tasks:

1. **Treatment of the patients' videos.** The gathering off all the frames of the different videos and their storage.
2. **Obtaining the best features and usage of light.** Try different image features in order to obtain the best combination for the purpose at hand. It includes the usage of light due to the fact that every aspect of the frames treated bases it's segmentation on a different usage of its light, like using the redness of an image (it involves the usage of the red channel of the image and its intensity).
3. **Preprocessing.** Preprocess each frame in order to segmentate it later as better as possible.
4. **Segmentation algorithms.** Find the best segmentation algorithm in order to separate the aperture area from the walls of the airway.
5. **Postprocessing and information gathering.** Process the segmented frame in order to gather as much useful information from it as possible.
6. **Previous frames.** Use the results obtained in previous frames in order to assure correctness in the data gathering of new frames and a consistency.
7. **Give a diagnosis.** Give a diagnosis based in all the gathered information during the past sub-tasks.
8. **Testing.** Time destined to test the implementation and evaluate the results obtained during the development of the project. Given that this part of the project takes more than 5 hours, it has its own sub-task assign to it.

The dependencies between the tasks are the following. First of all, the treatment of the patients' videos is needed before doing anything else, after that, in order to segmentate correctly, the obtention of the best features is needed. In third place, in order to segmentate each frame as easy as possible, the preprocessing is needed, that's why the third sub-task has priority over the fourth one. The fifth sub-task can only be done once the segmentation is done, and the sixth sub-tasks, previous frames, can be done alongside the postprocessing in order to keep a consistency during the information gathering. The seventh sub-task, give a diagnosis, can only be done once all the previous ones are completed. And finally, the testing of the different implementations takes place alongside all of the previous tasks throughout the project.

T.4 Final Stage

In this task, the final documentation of all the previous stages will be structured and properly written, and the preparation for the defense will begin. Given the fact that development has had various phases where big chunks of the code were modified or redone, the documentation would be constantly changing, so I've modified the total amount of hours needed to 100 in order to cover all the final documentation of the project.

T5. Meetings/e-mails

This final task consists of all the meetings and e-mails sent with the project director.

2.1.1 Summary table and estimations

ID	Task	Time (hours)	Dependencies	Resources
T1	Study of concepts and research	75		Pc
T2	Project Planning	80		
T2.1	Context and scope of the project	25	T1*	Pc, Microsoft Word
T2.2	Project planning	15	T2.1	PC, Microsoft Word, Trello, Monday.com
T2.3	Budget and sustainability	20	T2.2	Pc, Microsoft Word
T2.4	Final project definition	20	T2.1, T2.2, T2.3	Pc, Microsoft Word, Monday.com
T3	Development	200	T1*	
T3.1	Treatment of the patients' videos	15		Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.2	Obtaining the best features and usage of light	20	T3.1	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.3	Preprocessing	15	T3.2	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word

PROJECT PLANNING

T3.4	Segmentation algorithms	50	T3.3	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.5	Postprocessing and information gathering	50	T3.4	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.6	Previous frames	25	T3.5*	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.7	Give a diagnosis	15	T3.6	Pc, MATLAB, Patients' videos, GIT, Trello, Microsoft Word
T3.8	Testing	10	T3.1*, T3.2*, T3.3*, T3.4*, T3.5*, T3.6*, T3.7*	Pc, MATLAB, Patients' videos
T4	Final Stage	100	T2*, T3*	Pc, Microsoft Word
T5	Meetings/e-mails	20		Project Director
Total		475		

Table 1: Summary table with time estimations. Source: Own creation

*: The task that creates the dependency doesn't have to end in order to begin the task that depends on it.

Taking in account the original time estimated during the introduction, we have a total amount of about 30 extra hours in case we need them.

2.2 RESOURCES USED

In this section of the project, the different types of resources used are exposed. These resources can be grouped in three different types.

2.2.1 Human resources

There are three main human resources in this project. In first place, we have the developer of this project, whose duty (as exposed in the *stakeholders'* section of this project) is to do the planning, documentation, information search, development analysis and experimenting of the project. In second hand, we have the project director, who is in charge of leading and guiding the developer. And finally, we have the patients which bronchoscopies are used to develop this project.

2.2.2 Software resources

Multiple software resources are used during this project.

- MATLAB, for the development
- GIT, for source control
- Microsoft Word, for documentation
- Trello, for the Kanban board
- Monday.com, for the Gantt chart

2.2.3 Hardware resources

A computer with the following aspects was used during the development of this project: i5-11600K 3.90GHz, NVIDIA GeForce RTX 3060Ti, 32GB of RAM, 1TB SSD, 250GB SSD, 1TB HDD.

2.3 GANTT CHART

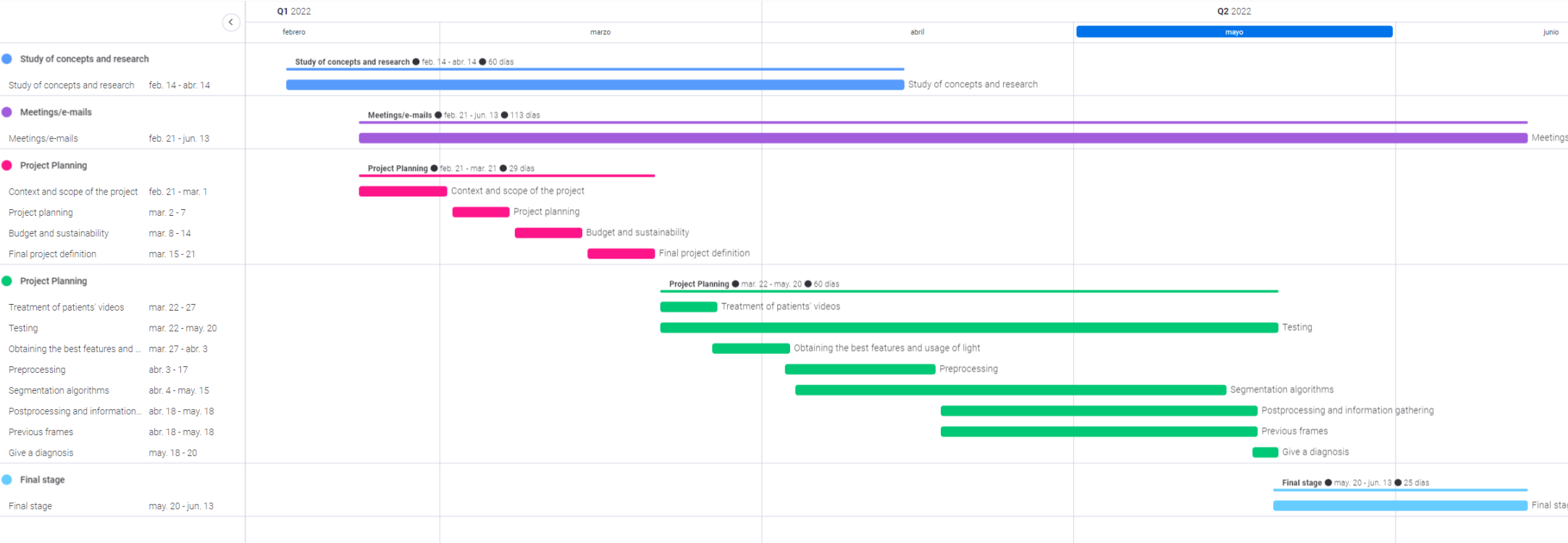


Chart 1. Gantt chart of the project's planning. Source: Own creation

2.4 RISK MANAGEMENT: ALTERNATIVE PLANS AND OBSTACLES

During the development, some obstacles or deviations that can potentially alter the initial planning may occur. These risks and obstacles have already been introduced in the *Obstacles and risks* section; in this section we simply present how they may be solved in order to progress.

- **Occlusion and erratic movement during the bronchoscopy.** In case that some frames of the videos present occlusion of the airway or erratic movement of the endoscope, two possible solutions can be put in practice. One solution would be to manually select a chunk of the video that doesn't present any kind of occlusion or erratic movement (this wouldn't take more than 1 hour because we would simply need to modify the frames used for the diagnosis) in order to be used as base for the diagnosis. The other solution would be to automatically discard the frame being treated; in case it doesn't give us any usable information.
- **Time consuming image processing algorithms.** In case that the image processing algorithms take too long, some hours have been granted to the planning of the development task, in order to meet the deadlines.
- **Not having enough time to make it to the deadlines.** This can probably happen due to the fact that this planning was done before starting the project, so a possible future rework, in a more advanced state of the project, could help us fix this problem. If even then, we lack of enough hours due to the extra work load that the developer has, the weekly hours destined to the project should be increased. And if even then it can't be done, maybe another lecture turn should be scheduled.

2.5 LAWS AND REGULATION

The main purpose of this project is to tell if it's possible to give a diagnosis for tracheobronchomalacia based solemnly on videos of bronchoscopies. Due to the fact that the patients' videos used during the realization of this project were given by a hospital with complete anonymity, and no names nor faces of any patient or any other type of private information is being shared, there is no infraction of any law or regulation taken into place.

3 BUDGET AND SUSTAINABILITY

This chapter deals with the budget and sustainability of the project. It includes a description of the material and human costs, and an analysis of the budgeted changes that the project may undergo due to development obstacles.

3.1 BUDGET

This section contains an estimate of the budget needed to develop this project. The budget can be divided into two different groups: direct and indirect costs.

3.1.1 Direct costs

We can refer to direct costs as all the expenses and resources incurred directly in order to make this project. This will be divided into three types: software, hardware and human.

Referring to the amortization, it has been taken into account that the project has a duration of approximately four months (roughly 120 days).

Software

The great majority of software used during this project needs a paid license, even though some products like MATLAB were used using an educational license provided by the university, an annual license price is used for this analysis.

The next table shows the costs of the software.

Product	Price	Lifetime	Amortization/day	Amortization
Windows 10	14.90€	3 years	0.01€/day	1.63€
MATLAB	800€	1 year	2.19€/day	263.01€
Microsoft 365	59.08€	3 years	0.05€/day	6.47€
GIT	0 €	-	-	-
Trello	0 €	-	-	-
Monday.com	0 €	-	-	-
Total	873.98 €	-	-	271.11€

Table 2. Software resources cost

Hardware

The amortization takes into account an approximate duration of 455 hours (approximately 18,96 working days) as already said during *chapter 2*, and the lifespan of the machine used during this project is about 5 years, so we have 1825 days of life (1 year has 365 days approximately), .

The next table shows the hardware costs.

Product	Price	Lifetime	Amortization/day	Amortization
PC Desktop	2150€	5 years	1.17€/day	22,34€
Total	2150€	-	-	22,34€

Table 3. Hardware resources cost

The price of the PC Desktop is from a year ago, when the developer of this project built it, the prices could vary from nowadays.

Human resources

Given the Gantt diagram previously showed, a role can be defined for each task:

- Project manager: Study of concepts, meetings with the director, project planning and final stage.
- Software developer: Development of the application.

In the next table, an estimation of the hours and tasks assigned to each role is displayed.

ID	Task	Time (hours)	Dependencies	Role
T1	Study of concepts and research	75		Project Manager
T2	Project Planning	80		Project Manager
T2.1	Context and scope of the project	25	T1*	Project Manager
T2.2	Project planning	15	T2.1	Project Manager
T2.3	Budget and sustainability	20	T2.2	Project Manager
T2.4	Final project definition	20	T2.1, T2.2, T2.3	Project Manager
T3	Development	200	T1*	Software Developer
T3.1	Treatment of the patients' videos	15		Software Developer
T3.2	Obtaining the best features and usage of light	20	T3.1	Software Developer
T3.3	Preprocessing	15	T3.2	Software Developer
T3.4	Segmentation algorithms	50	T3.1	Software Developer

BUDGET AND SUSTAINABILITY

T3.5	Postprocessing and information gathering	50	T3.3*, T3.4*	Software Developer
T3.6	Previous frames	25	T3.3, T3.4	Software Developer
T3.7	Give a diagnosis	15	T3.6	Software Developer
T3.8	Testing	10	T2*, T3*	Project manager
T4	Final Stage	100		Project director, Project Manager
T5	Meetings/e-mails	20		Project Director
Total		475		

Table 4. Tasks assigned to each role

In the following table, we expose the salary and number of hours worked for each role.

To determine the salary for each role, we used the website *Tusalario.es* [8], approximating the experience of the persons fulfilling the roles based on the experience that a university teacher and a student finishing the degree would have. Then the cost of Social Security is added (gross income multiplied by 1.3).

Role	Gross salary (40h/week)	Gross income (€/h)	Gross Income + SS (€/h)	Total hours	Estimated Cost (€)
Project director	3319€/month	20.74€/h	26.97€/h	20	539.4€
Project manager	2724€/month	17.025€/h	22.13€/h	275	6085.75€
Software developer	1852€/month	11.57€/h	14.83€/h	200	2966€
Total					9591.15€

Table 5. Human resources cost

3.1.2 Indirect costs

In this section we will see all the costs that we didn't take into account during the previous section, like internet and water costs, power consumption and rent.

Power consumption

The actual cost of the kWh is 0,369999 €/kWh [9]. Taking into account that the PC consumes about 300 watts, and counting the hours needed for the development of this project using the computer (455 hours), we are left with: $0.3\text{kW} * 455 \text{ h} * 0,369999 \text{ €/kWh} = 50.50 \text{ €}$.

Internet cost

The internet costs 33.99€/month. Taking into account that the project lasts four months and the average hours worked per day are 3.8h (455 hours / 120 days), we have an internet cost of: $4\text{months} * 33.99\text{€/month} * (3.8\text{h}/24\text{h}) = 21.527\text{€}$.

Water cost

I pay around 27€/month. Using the previously shown formula and using the same working hours we are left with the following numbers: $4\text{months} * 27.5\text{€/month} * (3.8\text{h}/24\text{h}) = 17.412\text{€}$

Rent cost

I will develop this project mainly at my flat, in Barcelona. The rental cost is of 860€/month, given that I live with another person, my monthly rent is 430€ per month. Given that this project takes four months, using the formulas shown previously, the total proportional cost is of: $4\text{months} * 430\text{€/month} * (3.8\text{h}/24\text{h}) = 272.33\text{€}$.

Generic cost of the project

The following tables summarizes all the indirect costs of the project.

Concept	Estimated Cost (€)
Power consumption	50.50€
Internet	21.52€
Water	17.41€
Rent	272.33€
Total	361.76€

Table 6. Indirect costs

3.1.3 Unexpected costs

In case of any deviation during the planning of the project, a small portion of the budget will be allocated, so we can face this setback. Due to the fact that the vast majority of difficulties during the project may come from the development, the allocation of budget results in the following.

BUDGET AND SUSTAINABILITY

Role	Gross income (€/h)	Gross Income + SS (€/h)	Total hours	Estimated Cost (€)
Project director	20.74€/h	26.97€/h	10	269.7€
Project manager	17.025€/h	22.13€/h	10	221.3€
Software developer	11.57€/h	14.83€/h	50	741.5€
Total				1323.5€

Table 7. Unexpected costs

3.1.4 Total Budget

The following table shows the total budget of the project with a contingency of 10% in case of any unforeseen events.

Concept	Estimated cost (€)
Direct costs	
Software	271.11€
Hardware	22,34€
Human resources	9591.15€
Indirect costs	361.76€
Unexpected costs	1323.5€
Subtotal	11569.86€
Contingency (10%)	1156.99€
Total	12726.85€

Table 8. Total budget of the project

3.2 MANAGEMENT AND BUDGETARY CONTROL

Due to the fact that time is short and there may come some obstacles during the development of this project, the original time estimations may not be fulfilled, so a means to define a model for controlling budget deviations is needed.

Given the different deviations, based on the fluctuation on the approximated hours of consumption and cost of resources and labor-force that may be present during this project, we will use the following formulas to calculate them:

- **Resource pricing deviation:** $(\text{estimated cost} - \text{real cost}) * \text{real consumption}$
- **Resource consumption deviation:** $(\text{estimated consumption} - \text{hours consumed}) * \text{real cost}$
- **Labor pricing deviation:** $(\text{estimated cost} - \text{real cost}) * \text{real consumed hours}$
- **Labor consumption deviation:** $(\text{estimated hours consumed} - \text{real consumed hours}) * \text{real cost}$
- **Total labor deviation:** $\text{total estimated labor cost} - \text{total real labor cost}$

- **Total resource deviation:** total estimated resource cost – total real resource cost

In case of any deviation during the realization of the project, all will be calculated during its final stage.

3.3 COST CHANGES

Given the previous mentioned changes, 20 hours have been added to the final stage of the project, with a total cost of 442.6 € (22.13€/h of the project manager * 20 hours), which is covered by the allocated budget that was destined in case of any setbacks during the development. The total amount of budget destined to unexpected costs was 1323.5 €, with the 442.6 € added cost there's left a margin of 882.9 €.

3.4 SUSTAINABILITY

3.4.1 Self-assessment

During the realization of this project a survey to self-assess our knowledge based on sustainability was done.

Before starting GEP, when I was thinking about being sustainable in a project, I was always thinking about taking into account the effects and repercussions that a project may take onto the environment, and trying to make everything as eco-friendly as possible. I totally ignored the social and economic spectrum. Due to this, I can see that I had a wrong vision of sustainability in a project, and that I lacked to understand its true meaning.

Even though having done the pool and the previous sections of this project, I don't see how I could make it more sustainable, environmentally talking, given that all I have to do is based using the computer, and in the process, I have to use a great deal of energy. So, I can't see how to reduce its impact in any way other than by reducing the total number of hours needed to complete the project.

In the social aspect, I've never really given a thought about the impact a project may have in society, more than being polite and helpful with other parties that may be involved in its realization. And even though there wasn't any intention, I can see how the completion of this project could have a positive impact in society. I will detail more this subject in the society section.

Referring to the economic aspect, as described before, I've never given a thought about it, and I have to say that in perspective, it's a very important point take into account during any development.

In conclusion, after taking the pool and making the budget and sustainability chapter of this project, I can see that I had a very short minded vision of what sustainability really is.

3.4.2 Economic dimension

Regarding Project Development: Reflection on the cost you have estimated for the completion of the project

The estimation of the cost needed for the completion of the project is already documented in the *budget* section of the document, taking into account all the resources necessary to develop it.

In my opinion, the budget for the project is reasonable and could be carried out in real life due to the fact that the medical field is a very important one, and any improvement on it affects the whole world.

Regarding Project Development: Have you quantified the cost (human resources and materials) of the realization of the project? Which decisions have been made in order to reduce its cost? Have you quantified its savings?

Yes, this cost has been quantified during the direct costs section of the document. No decisions have been made in order to reduce the cost, due to the fact that they cannot be reduced, on the other hand, they could be increased if more time were needed to finish the project. I have not quantified any savings because I have not reduced the cost.

Regarding Project Development: Has the planned budgeted been brought into line with the final budget?

Yes, as stated during the cost changes section, the extra cost of the project is widely covered by the budget destined to unexpected setbacks.

Regarding Useful Life: How are currently solved economic issues (costs...) related to the problem that you want to address (state of the art)?

I don't really see any way of reducing more the cost of this project given that it's based on the analysis of a given video produced by a basic endoscope. Maybe using some type of A.I. based library able to detect the airway area could limit the development time, and consequently the total amount of hours needed.

How will your solution improve economic issues (costs ...) with respect other existing solutions?

The tracheobronchomalacia disease is usually diagnosed by eye, through a bronchoscopy, as explained before. If this project succeeds in its objective, an implementation in real life could save time from medical professionals, as a program could detect the disease instead of them, saving time and resources.

Regarding Useful Life: Which is the cost that you expect the project will have during its life course? Could it be reduced?

This is already explained during the *budget* section of this chapter. It couldn't be reduced due to the fact that nearly all the hours have been used.

Regarding Risks: There could be a scenario that would harm the viability of the project?

I don't think so, no scenario comes to my mind that would impact on the economic viability of the project. Maybe if any license expired or the water, electrical bill, or the rent would increase.

3.4.3 Environmental dimension

Regarding Project Development: Have you estimated the environmental impact of the project?

I have not, the only impact this project has on the environment is the electrical consumption. Given that nearly every aspect of this project is done with a computer there will be a high electric power usage.

Regarding Project Development: Have you quantified the environmental impact of the realization? Which measures have you adapted in order to reduce its impact? Have quantified this reduction?

I have quantified it during the indirect costs section of this project, where an estimation of the total amount of power is needed in order for its realization ($0.3\text{kW} * 455 \text{ h} = 136.5 \text{ kwh}$). No measures have been adapted in order to reduce the impact due to the fact that I can't spend less time using the computer for the realization of the project, and so I have not quantified its reduction.

Regarding Project Development: Given the possibility of redoing the project, would you be able to reduce the consumption of resources.

I don't think so, because as previously mentioned the established resources are the needed ones for its realization.

Regarding Project Development: Did you plan to minimize its impact, for example, by reusing resources?

As already said before, given that the biggest part of this project is the implementation in MATLAB, the only way to reduce its impact would be using a library able to detect the trachea and bronchi airway dimensions based on a frame, if it existed.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)?

As stated before, by eye using the video frames obtained by the bronchoscopy.

How will your solution improve the environment with respect other existing solutions?

It would not, given that the usage of a machine able to run the program to give the diagnosis is needed.

Regarding Useful Life: Which resources would you estimate that will be used throughout the useful life of the project? Which will be the environmental impact of this resources?

I already estimated them during the indirect costs section of the document, and they are the power consumption and water supply resources. The environmental impact would be as the same as the impact of any other person working from

home, consuming electrical power to use the computer, and water in order to drink and clean oneself.

Regarding Useful Life: Would the project allow the usage of other resources? Will the project better or worsen the global ecological footprint?

Due to the fact that the project needs a computer for a user to use it, the project would worsen the global ecological footprint.

Regarding risks: There could be a scenario where the ecological footprint of the project would increase?

No, because the power and water consumption would stay the same no matter what.

3.4.4 Social dimension

Regarding Project Development: What do you think you will achieve -in terms of personal growth- from doing this project?

First of all, it will allow me to obtain my degree in computers which will help me in the labor world. And secondly, it will allow me to correctly develop real projects outside university.

Regarding Project Development. The realization of this project has supposed any significant reflection to a personal or professional level, etc. of any of the persons involved?

The realization of this project has taken a toll into my personal level due to the fact that, as previously stated, I as the developer of the project, have a full-time job and I'm enrolled in other two subjects at the same time whiles doing the TFG. So, my stress levels have increased drastically and my quality of life has diminished a little thanks to it.

Regarding Useful Life: How is currently solved the problem that you want to address (state of the art)?

As stated previously, giving a diagnosis, by eye, based on the video feed of a bronchoscopy.

How will your solution improve the quality of life (social dimension) with respect other existing solutions?

Giving a correct diagnosis for the tracheobronchomalacia disease would help the medical professionals in their assessment of a diagnosis, that in return would help the patients suffering from it as well.

Regarding Useful Life: Is there a real need for the project?

Given the fact that every year, more and more advancements in medicine are made, and the intervention of technology is continuously increasing in the field. A means to help a medical professional to give a diagnosis for a disease it's always going to help society and thus, there is a real need for it.

Regarding Useful Life: Who will benefit from using the project? Is there a collective that could be adversely affected by the project? In what measure?

As previously mentioned, yes, with more time and future improving and trial of the project, patients and medical practitioners would be benefited. I can't think of any collective being aggravated by this project.

In which measure does the project fix the initial proposed problem?

Depending on the bronchoscopy given of a patient and the usage of the program, the problem that is proposed can be accomplished given a steady and occlusion-free sample.

Regarding Risks: Is there any scenario that could make the project harmful for any population group?

As already stated before, there is not any group that could be harmed by this project, due to the fact that the objective of this is to help the population.

Could there be any kind of dependence created by the project, that would leave the users in a debilitating state?

No, because the project aims to assist the user, not relieve it from its duties.

4 DEVELOPMENT

This chapter deals with all the development of the program, explaining all the methods and algorithms used in order to analyze every frame from the bronchoscopies.

4.1 INTRODUCTION

Computer vision allows us to gather meaningful data from images, as already stated during the first chapter, in order to automatize tasks that we can achieve by using our own eyes. For example, in the case of self-driving cars, computer visions allow cars to detect and analyze objects, signals and persons (in real time) in order to know how to drive without going off course or having an accident, basically it works as the eyes of the car.

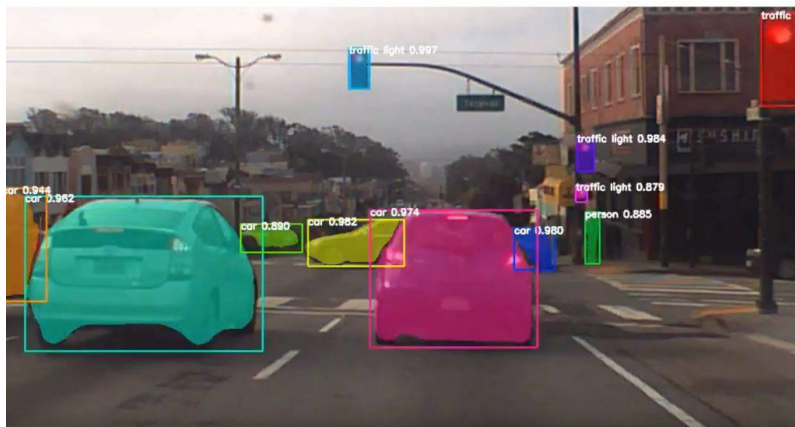


Figure 4. Object detection of a self-driving car. Source: *Unbundling The Autonomous Vehicle*, *cbinsights*

In the case of this project, computer vision is applied in order to gather data from the various frames of a patient's bronchoscopy, calculating various parameters as the diameter of aperture of the trachea or the area, in order to give a diagnosis and indicate the degree of stenosis of the trachea.

4.2 GATHERING OF FRAMES

There are three main scripts that define this program, the first one which consists of the gathering of frames from a video source in order to treat them later during processing, is the one that will be explained during this section and it's called *getFrames.mlx*.

First of all, the script asks the user to type in the path of the video, once the path is stored, the script saves all the frames of the video into a folder with the name of the video, removing said folder previously in case it already exists, at the MATLAB's environment path where the script was executed. Every frame is stored as a .jpg with the number of the frame as label.

For example, in the case of having a video called sick1.mpg of a patient, simply by introducing its path the script would create a folder named sick1, where the it was executed, filled with all the frames of the video labeled by their number in the occurrence of the video.

4.3 TREATMENT OF FRAMES

This section of the project exposes the second and most important script, the *mainProgram.mlx*, which is responsible for the gathering of all the information from the frames, its evaluation and final diagnosis, following the desired segmentation algorithms chosen by the user.

The script works in the following way:

- Having used the *getFrames.mlx* script in first hand, the script will ask the user for the name of the folder containing the frames that need to be analyzed.
- Ask the type of segmentation algorithm that is desired for the task at hand and the color channel or color space (more abouts color channels will be explained during this chapter) that is going to be used depending on the algorithm chosen.
- The script asks if the user wants to preprocess the frames before applying the segmentation algorithms.
- A folder named after the segmentation algorithm used is created in order to save all the treated frames.
- A range of frames to be analyzed is asked to the user, this is due to the fact that many frames from the bronchoscopies could have long sections with a lot of noise on the image due to erratic or quick movements of the bronchoscope or the lenses getting occluded by liquid, so it's better to treat

a chunk of video that has a clear image and a limited movement of the bronchoscope.

- The segmentation of each frame and storage of the results takes place.
- The script treats the results, and gives a diagnosis, displayed in a message box, based on them.

The different parts of the script are explained in detail in the different subsections below.

4.3.1 Cropping and preprocessing

Before any kind of segmentation algorithm is applied to the video frames, a cropping of the image takes place in order to remove the black borders and not useful data from the image (white text).

Due to the fact that all the videos that have been provided by the hospital have the same resolution, the cropping options for the image are hardcoded for all the frames.

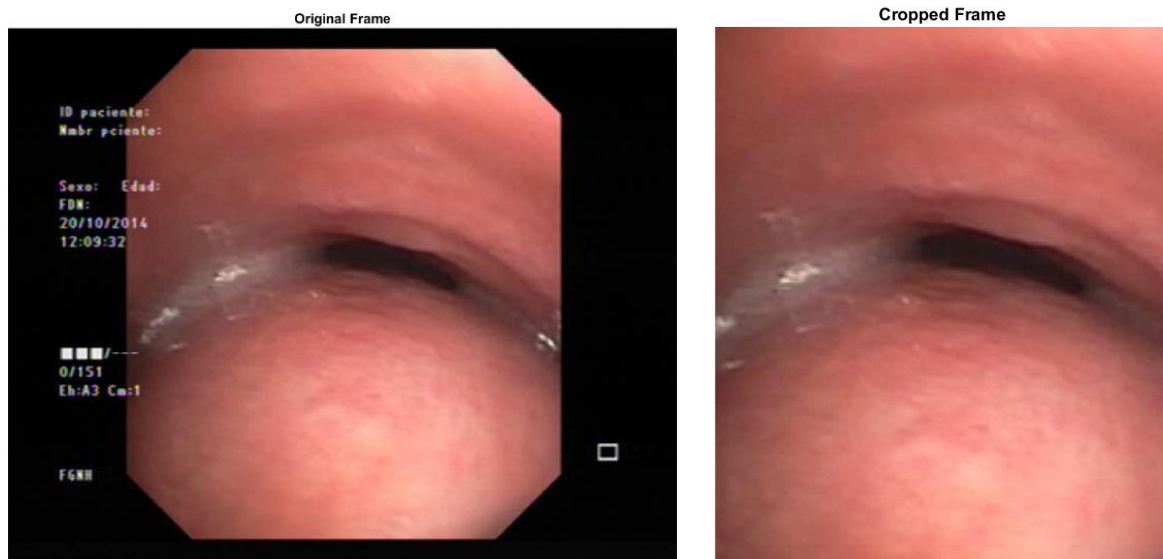


Figure 5. Original patient frame and cropped frame. Source: Own creation

Once the frame is cropped, the image is segmented into its three-color channels and a grayscale version of the all the image is saved. By default, an image is stored as a 3-dimensional matrix containing the light intensity for each Red, Green and Blue channels with values in a range between 0 and 255 (being 0 the lowest intensity of light and 255 the maximum).

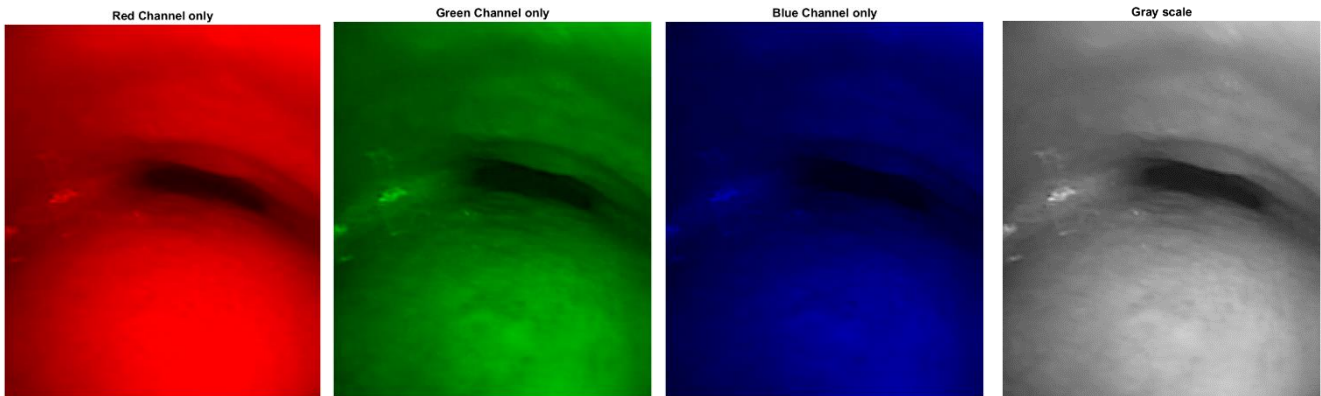


Figure 6. RGB channels and gray scale. Source: Own creation

Finally, once the desired channel is obtained, its preprocessing begins (if selected during the execution of the script).

The preprocessing of the image consists of a series of morphological operation of the image in order to get more consistent levels across the whole matrix, getting as a result an image that looks a little bit faded, where little light highlights, due to the camera light reflecting against the liquid in the trachea, blends with its surroundings.

First of all, a morphological erosion of the image is done using a disk-shaped structuring element. Then, using image reconstruction of the eroded frame as a marker, together with the original frame as a mask, an opened by reconstruction frame is obtained. Once the opened frame is obtained, its dilation followed by a reconstruction of the complement of the dilated frame together with the complement of the opened by reconstruction frame allows us to obtain the desired effect on the frame to preprocess.

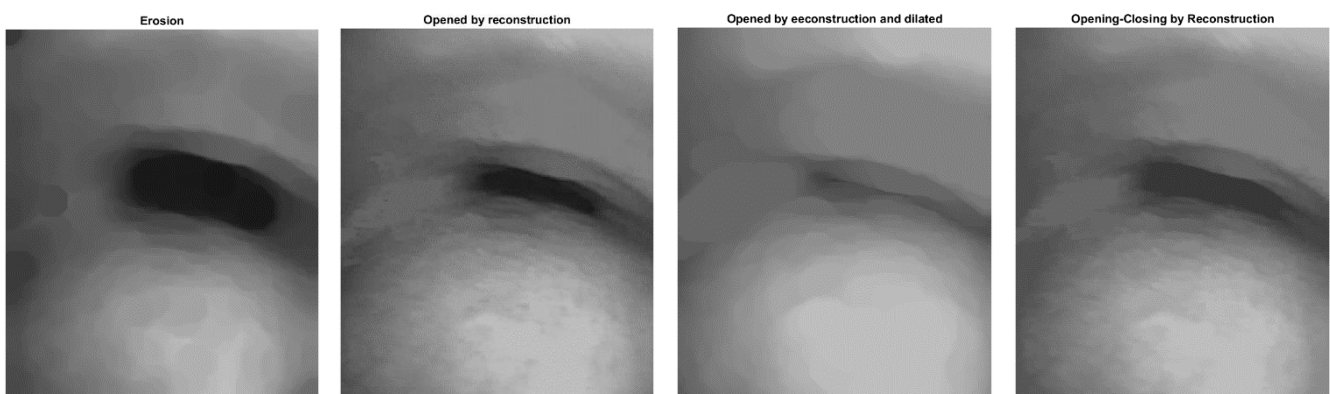


Figure 7. Erosion, Open by Reconstruction, Dilation and Open/Close by Reconstruction of the frame being treated. Source: Own creation

4.3.2 Segmentation algorithms

Segmentation consists on dividing an image in regions with similar characteristics.

Three kinds of segmentation algorithms have been used in this project in order to segment the opening of the trachea from the rest of the image:

- **Binarization based segmentation**

Consists on reducing the number of grey levels of an image into two levels (foreground and background). The segmentation algorithms of this kind tested during this project are the following ones:

- An own implementation of segmentation based on image histogram binarization.
- Segmentation based on adaptive binarization.
- Otsu binarization.
- Niblack's Binarization.
- Sauvola's Binarization.
- Mean value Binarization.
- Kapur's Binarization.

- **Region based segmentation**

Consists on dividing the image into different regions based on similarities between values of adjacent pixels. The segmentation algorithm of this kind tested during this project is the following one:

- K-means segmentation

- **Edge based segmentation**

Consists on dividing the image into different regions based on differences between values of adjacent pixels. The segmentation algorithm of this kind tested during this project is the following one:

- Watershed segmentation

Finally, an implementation that takes into account the algorithms that yield better results, from each of the previous segmentation families, has been made in order to get the most accurate diagnosis possible. This algorithm is called mixed bag segmentation.

Each of the previous algorithms is presented below:

1. Own binarization based segmentation implementation

This algorithm uses the histogram of the frame in order to binarize the image. First of all, the histogram of the image is calculated in order to gather the number of pixels that fall within the same range and the location of the ranges, passing per parameter the number of bins which is desired for the histogram to have. For example, per default the number of bins in the call for a channel of an image is of 256 bins, due to the fact that grayscale ranges from 0 to 255.

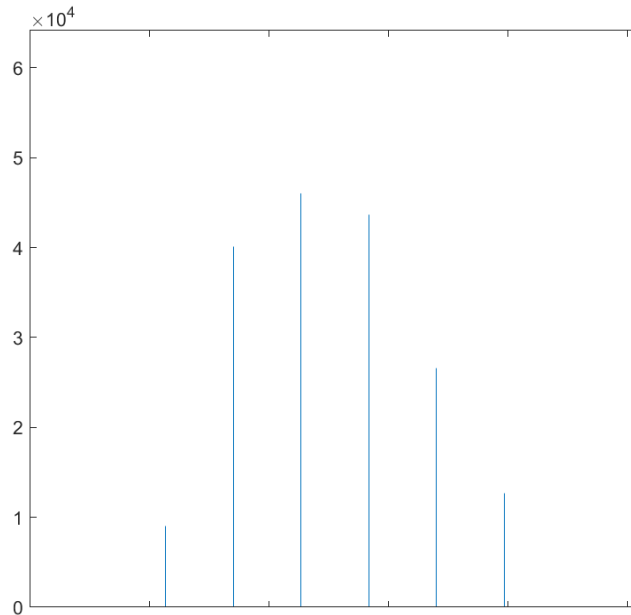


Figure 8. Histogram with a total of 10 bins. Source: Own creation

Then, local maximas of the results gathered from the histogram are searched. Once the first local maxima from the histogram is found, the threshold used to binarize the frame is found by calculating the difference in distance between the previous bin to the local maxima and the bin at the local maxima taking into account the percentage of total amount of pixels per each bin.

For example, in the case that the first local maxima has is located at the bin location 100 of the [0,255] spectrum, and the previous bin is at location 50, if the number of pixels that fall within the range of the first bin is of 2500 pixels and the number of pixels that fall within the range of the second bin is of 10000, then the threshold would be set at $50 + \text{binDiff} * \text{percentageOfPixelDiff}$, where $\text{binDiff} = 100 - 50$ and $\text{percentageOfPixelDiff} = (100 - (10000 - 2500) / 100)$.

In case of there not being a local maxima, the algorithm calculates the differences in number of pixels between each pair of consecutive bins and gathers the pair which have the greatest difference. Then applies the same formula in order to obtain the binarization threshold.

If the frame wasn't previously preprocessed, a 12.5 percentage of the threshold is subtracted for fine tuning due to overestimation of the threshold.

Finally, once the threshold is obtained, in case that the previous evaluated frame had a valid threshold, the new threshold is calculated applying the percentage of difference between both the previously mentioned threshold and the one calculated during this call to the algorithm. Then it simply binarizes the image evaluating to true all the pixels that had a value lower or equal to the threshold, and evaluating to false the other ones.

2. Adaptive binarization

This algorithm computes a locally adaptive threshold choosing a threshold based on the local mean intensity in the neighborhood of each pixel [10], given a sensitivity. The higher the sensitivity value, the more pixels the algorithm will consider as foreground, which would risk including some background pixels in it.

3. Otsu's binarization

This algorithm returns the threshold that minimizes the intra-class intensity variance, or what is the same, returns the thresholds that maximizes the inter-class variance by stepping through every possible threshold. Simply put, this algorithm finds the threshold value where the sum of foreground and background pixels spread is at its minimum. [11]

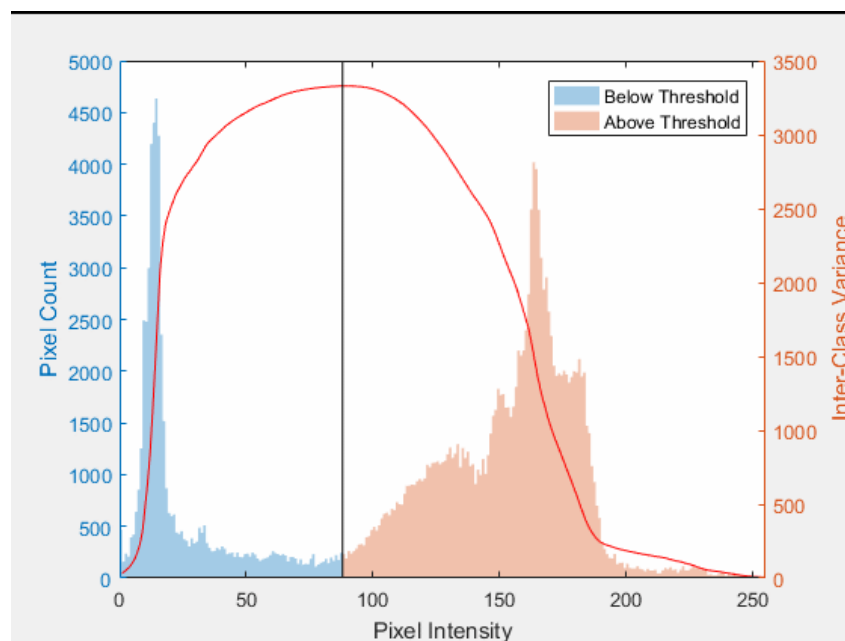


Figure 9. Otsu thresholding. Source: Wikipedia

4. Niblack's binarization

This algorithm returns the segmented frame by performing local thresholding using a sliding window of M-N neighborhood, setting each local threshold as the result of the following formula:

$$t = \mu - 0.2\sigma$$

Where μ is the mean of the sliding window and σ the standard deviation.

In the case of this project, the sliding window is set to 0.6 times the height of the frame by 0.6 times the length.

5. Sauvola's binarization

This algorithm is a variation of the Niblack's binarization algorithm. Its formula is the following one:

$$t = \mu(1 + k(\sigma/r - 1))$$

Where μ is the mean of the sliding window, σ the standard deviation, k it's a constant initiated per default at 0.34 and r is the maximum deviation from all the sliding windows.

In the case of this project, the sliding window is the same as the previously mentioned in the Niblack's binarization.

6. Mean value binarization

This algorithm basically calculates the threshold as the result of the mean value of all values passed as the input matrix.

7. Kapur's binarization

This algorithm classifies the input image into multiple classes by comparing the entropy of the histogram, the higher the entropy value, the more homogeneous classes. The entropy of a given image indicates the compactness and separateness among distinctive classes. By default, the entropy of an image is equal to $-\sum(p \cdot \log_2(p))$, where p contains the normalized histogram counts.

Assuming that $[th_1, th_2, \dots, th_n]$ represent the threshold combinations which divide the input image into various classes. Then Kapur's method can be defined as:

$$H(th_1, th_2, \dots, th_n) = H_0 + H_1 + \dots + H_n \quad (1)$$

Where:

$$H_0 = - \sum_{j=0}^{th_1-1} \frac{p_j}{\omega_0} \ln \frac{p_j}{\omega_0}, \quad \omega_0 = \sum_{j=0}^{th_1-1} p_j$$

$$H_1 = - \sum_{j=th_1}^{th_2-1} \frac{p_j}{\omega_1} \ln \frac{p_j}{\omega_1}, \quad \omega_1 = \sum_{j=th_1}^{th_2-1} p_j$$

$$H_n = - \sum_{j=th_n}^{L-1} \frac{p_j}{\omega_n} \ln \frac{p_j}{\omega_n}, \quad \omega_n = \sum_{j=th_n}^{L-1} p_j$$

H_0, H_1, \dots, H_n denote the entropy of distinct classes and $\omega_0, \omega_1, \dots, \omega_n$ are the probabilities of each class.

In order to obtain the optimal threshold values, the threshold that maximizes the entropy of the equation (1) is returned. [12]

8. K-means segmentation

K-means works in the following way:

1. Randomly initialized k cluster centers, c_1, \dots, c_k
2. Given each cluster centers, determine points in each cluster:
 - a. For each point p , find the closest c_i . Put p into cluster i
3. Given points in each cluster, c_i set to be the mean of points in cluster i .
4. If c_i have changed, step 2 is repeated.

This algorithm can use as input any kind of multidimensional matrix, in the case of this project, two types of inputs have been used for testing, RGB and $L^*a^*b^*$ color space. Where $L^*a^*b^*$ color space is another kind of representation of color in an image, where the L^* dimension represents the luminosity of the image, the a^* dimension indicates where each pixel falls along the red-green axis and the b^* dimension indicates where the color falls along the blue-yellow axis. This color space has been used for testing in order to see how the k-means segmentation would work using only the color representation of each pixel on the image without taking into account its brightness (a^*b^* dimensions only).

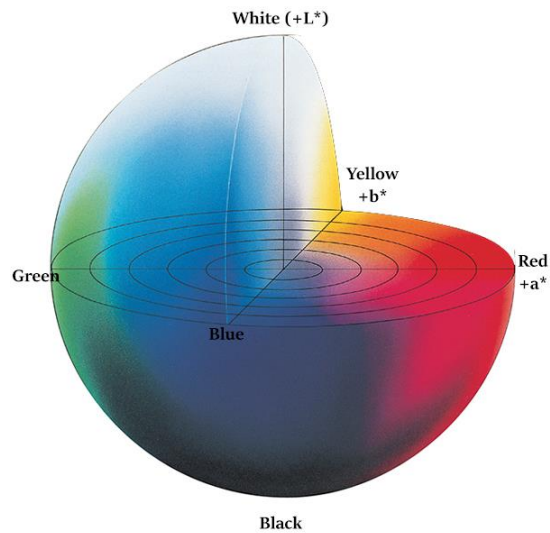


Figure 10. $L^*a^*b^*$ color space representation. Source: Google

The implementation of k-means segmentation for the RGB color space is the following one:

1. Apply k-means to the input frame
2. Find the cluster where the intensity of each channel it's at its lowest.
3. Return the mask with said cluster.

There is an additional flag called *flagOuterMaks* in the implementation of the function that allows to return the frame segmentation using the cluster with higher values of light intensity for each channel of the input frame.

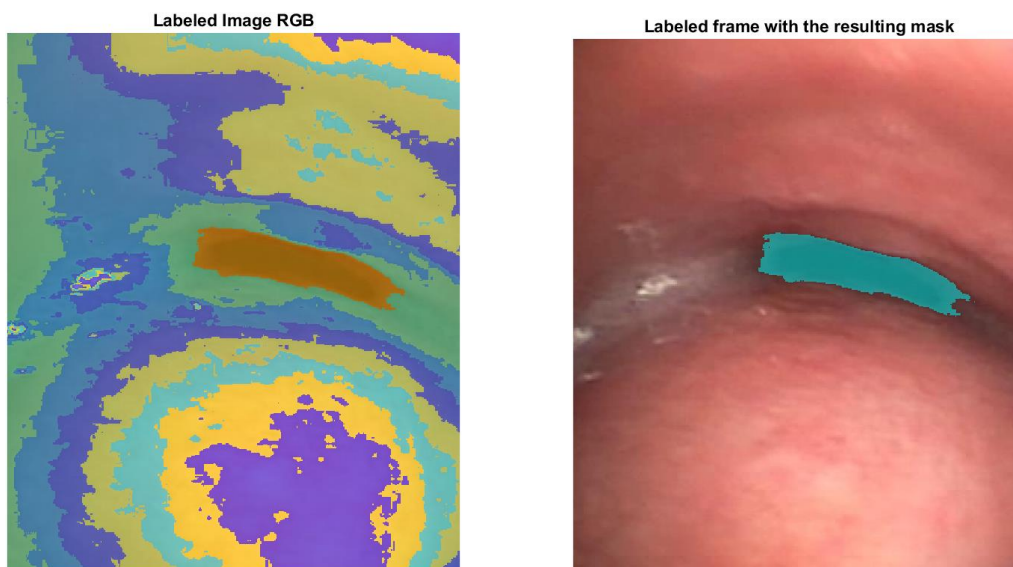


Figure 11. Labeled RGB image with the k clusters and final mask. Source: Own creation

On the other hand, the implementation of k-means segmentation based on the L^*a^*b color space is the following one:

1. Separate the L^* channel from the L^*a^*b color space input frame.
2. Apply k-means only to the a^*b^* channels
3. Use the L^* channel in order to get the cluster with the minimum value of light.
4. Return the mask of the cluster returned in point 3.

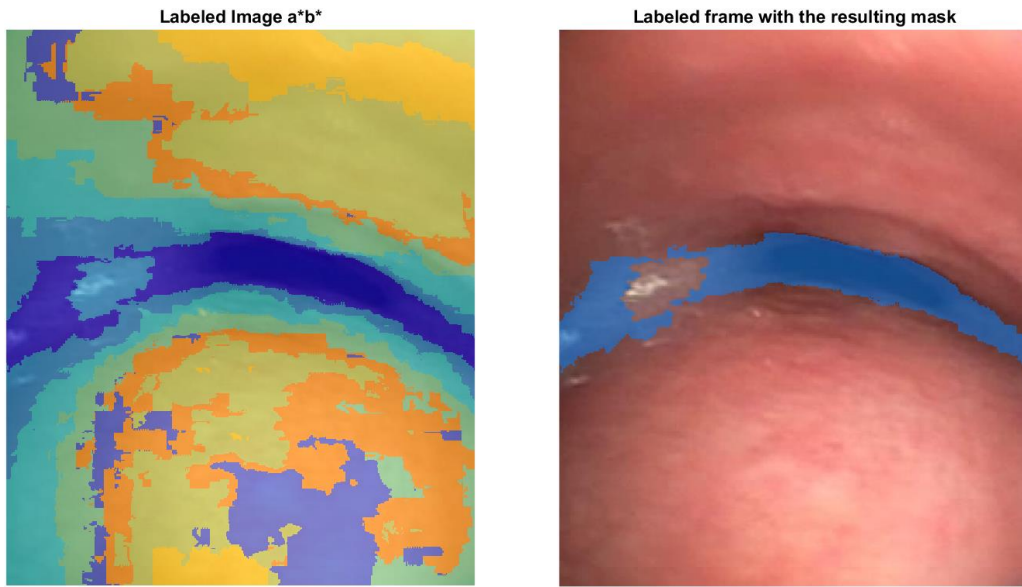


Figure 12. K-means labeled a^*b^* image and final resulting mask. Source: Own creation

9. Watershed segmentation

An intuitive description of what watershed segmentation consists of would be the following one:

1. An image is represented as a plastic relieve
2. The plastic is punctured in some markers
3. The plastic is submerged into water progressively
4. The water entering through every hole on the plastic is labeled
5. Once different labeled waters touch, a wall is built between them separating one from the other
6. The algorithms keep submerging the plastic until everything is flooded
7. Once the algorithm ends the walls represent the lines of the watershed and the image is partitioned in regions

The implementation of this algorithm, in this project, uses the k-means segmentation algorithm previously mentioned, on the RGB color space, in order to get sturdy markers for the inner-most and outer-most regions, with a k value of 2 and 14 respectively, in order to get markers based on color value. Once the markers are gathered, the regional maximums and minimums are added in order to get even more markers for the watershed, so we get an even more accurate result.

Then, using the morphological closing and erosion, followed by the elimination of small connected components, the markers are cleaned up from rough edges and noise.

Outer-most markers and inner-most markers imposed on frame



Figure 13. Inner-most and outer-most markers imposed on frame. Source: Own creation

Once the markers are finally obtained, the image gradient of the preprocessed gray level of the frame is calculated in order to get the edges that will work as ridge lines for watershed. Then the markers are imposed on the image gradient getting the value of the regional minima on the region of each marker using, in order to prepare the frame for the watershed.

Finally, once the watershed is done, a mask is applied to the image filtering by the label of the inner-most marker obtained from the call to k-means, so the result is the desired segmented.

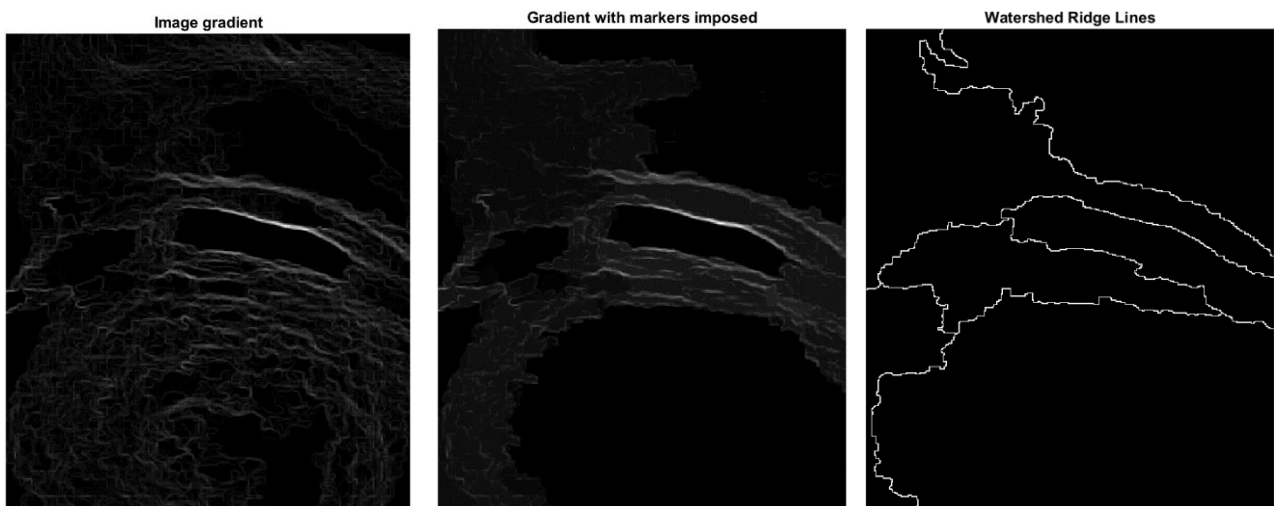


Figure 14. Image gradient, gradient with markers and watershed ridge lines. Source: Own creation

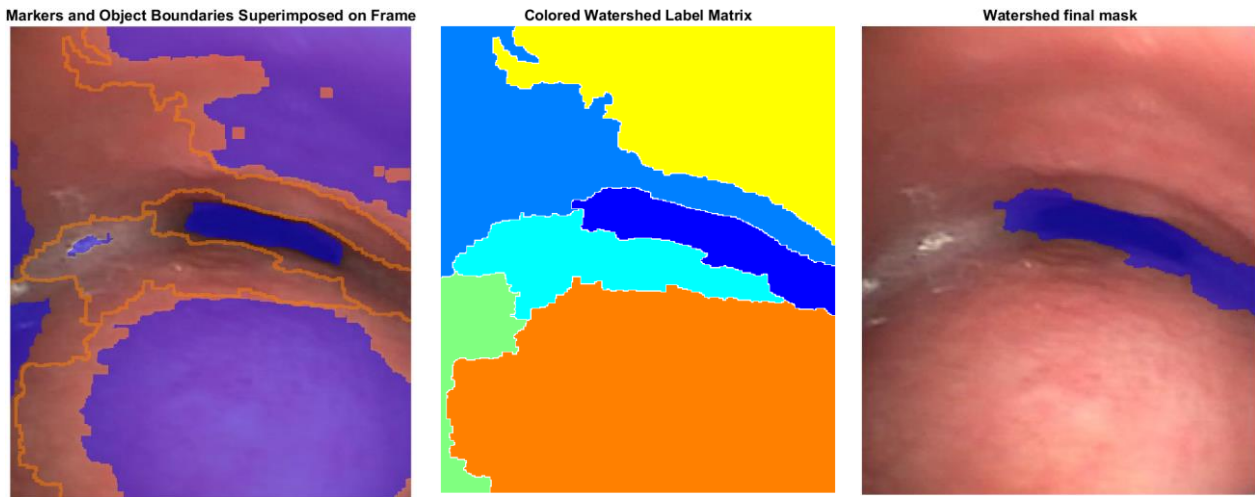


Figure 15. Ridge lines and markers imposed on frame, Watershed labels and Watershed final mask. Source: Own creation

- **Mix bag segmentation**

As stated previously, this segmentation is based on a fusion of the three algorithms (one from each type of segmentation) that yield better results.

It is based upon the usage of our own implementation of a binarization based segmentation algorithm, together with the k-means segmentation and the watershed segmentation.

Basically, the algorithm calculates the results of each segmentation (opening diameter of the trachea, area of the airway and the mean value of the red channel of the segmentation) using the *getSegment* and *getStenosisDegree* functions (more about this function in the next section of this chapter), and then, decides whether to use k-means, binarization or watershed, depending on the results gathered.

The logic behind choosing which segmentation algorithm use is the following:

- By default, k-means segmentation is used for the segmentation of the frame
- For each segmentation algorithm, if the calculated aperture of the airway is superior or inferior to the aperture of the airway of the previous frame times an aperture threshold (declared at 0.8 by default) set as a global variable at the beginning of the script, or the segment's area is superior or inferior than the area of the previous' frame segment times 0.75, the segmentation algorithm is invalidated and will not be used for the segmentation of the frame.
- If the k-means segmentation is valid and the aperture of the airway is greater than 80, or the aperture is in the interval (50, 80] and the mean values of the red channel's segment is less than 60, or the major axis

length of the segment is greater than two times the aperture, then the binarization segmentation is used (in case of being valid and having a greater aperture), if it isn't valid then the watershed segmentation is used. If neither the watershed nor the binarization are valid, or the previously mentioned condition for k-means isn't met, then the segmentation algorithm used is k-means.

- If k-means segmentation is invalid, binarization is used, if valid, otherwise watershed segmentation is used. If neither k-means, binarization nor watershed are valid, no segmentation algorithm is used and the frame is discarded.

4.3.3 Calculating the final blob

In order to calculate the blob that will represent the opening of the airway, the *getSegment* function is used. This function closes possible gaps that could be left inside the mask resulting of the segmentation, and creates a more uniform blob by closing holes (setting to true those pixels that have a false state but are surrounded by pixels with a true value) from the input logical matrix, and by using morphological closure with four structuring elements of line type with different slopes (vertical, horizontal and diagonal lines).

Then, small connected components (small blobs) are removed from the frame in order to clean up the final blob. All the blobs that are smaller than 0.3 times the area of the biggest blob of the frame are deleted. Moreover, in case that there still is more than one blob, only the one that is the closest to the centroid of the previous frame remains, the rest of blobs are deleted. In case that the previous frame is invalid, the center of the image is used as the previous centroid.

Finally, more gaps are closed and filled in order to make some final adjustments to the final blob.

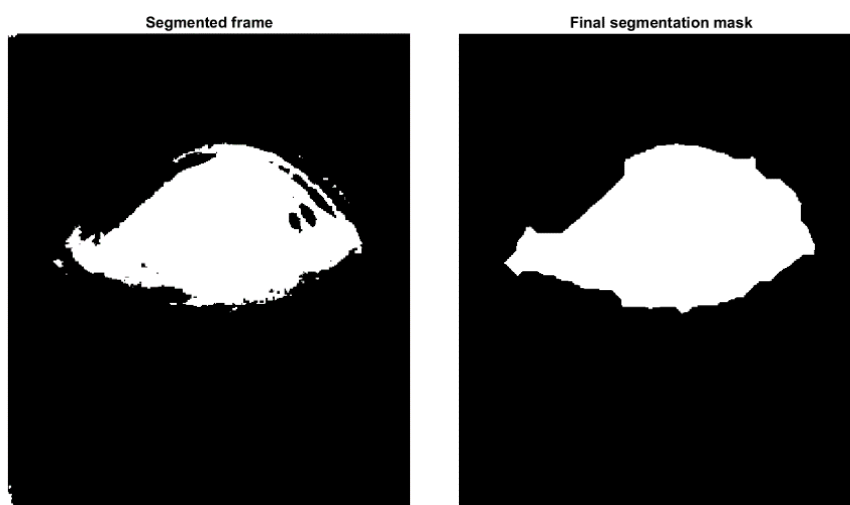


Figure 16. Segmented frame without treatment and final blob

4.3.4 Gathering information from the final blob

Once the final blob representing the opening of the airway is obtained, the gathering of the results from the frame is left. The function that is in charge of such task is the `getStenosisDegree` function, which returns the aperture, the coordinates that represent the line of the diameter of the aperture, the centroid of the blob (with some tuning depending on the orientation of the blob), the area of the blob, the bounding box, the midpoint of the aperture, and the minor axis length.

This function can be divided in the following steps:

1. Detection of the edge of the blob.
2. Getting the centroid, bounding box, orientation, area and the minor axis length of the blob.
 - a. The centroid represents the coordinates of the center of mass of the blob.
 - b. The bounding box represents the smallest box containing the whole blob.
 - c. The orientation is the angle between the x-axis and the major axis of the ellipse that has the same second-moments of the region. The value is in degrees and it ranges between -90 and 90. [13]
 - d. The area is basically the number of pixels inside the blob.
 - e. The minor axis length is the length of the minor ellipse that has the same normalized second central moments as the region. [13]

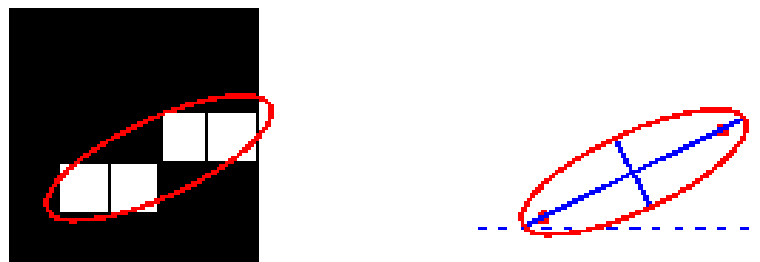


Figure 17. Ellipse that englobes a blob. Source: MathWorks

3. Depending on the orientation of the blob, the centroid has its y value assigned to half the height of the bounding box, or has its x value assigned to the half of the length of the bounding box.
4. The aperture of the blob is calculated by tracing a line, perpendicular to the orientation, that intersects with the two furthest points of the edge of the blob, with its origin at the centroid previously calculated.

- All the return values are assigned and the function ends.

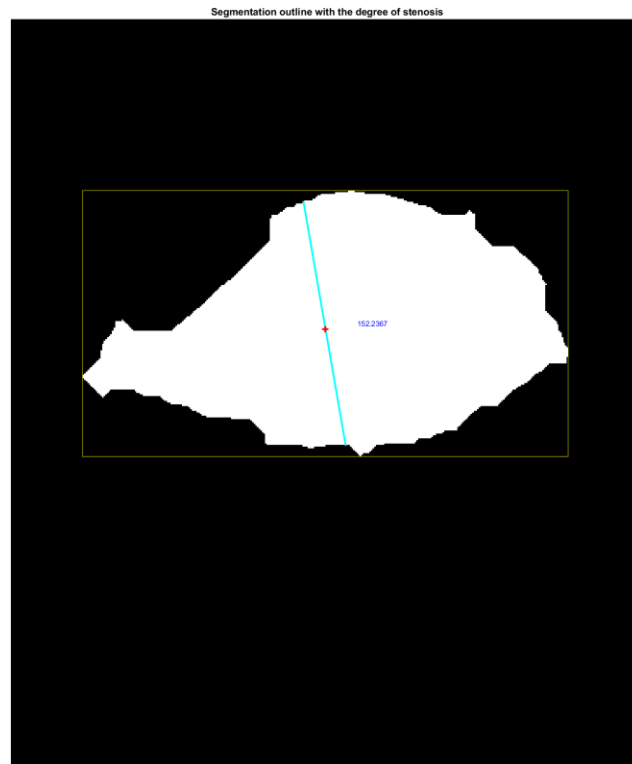


Figure 18. Bounding box, aperture, centroid and aperture diameter of a blob. Source: Own creation

4.3.5 Computation of the results

The final part of the *mainProgram* script, can be divided in two parts: the validation and storage (or not) of the current treated frame compared with previous frames, and the diagnosis.

- **Evaluation of the current frame**

Once the information of a frame has been gathered, if there was no problem in any of the previous steps (the frame is valid) and the aperture of the resulting blob is considered valid, the name of the frame, the aperture, the area and the mean value of each of the RGB channels are stored for analysis. The global variables *previousAperture*, *previousCentroid* and *previousArea* are updated with the value of the current frame, the number of consecutive misses (number of invalid frames in a row) is set to 0 and the cropped frame, with all its gathered data plotted on top, is stored on a folder named after the name of the

folder containing the original frames of the video and the segmentation algorithm used.

In case the frame or the aperture were not valid and the consecutive misses are lower than the maximum number of consecutive misses allowed (global variable *maximumConsecutiveMisses* initialized at 2 per default), the name of the frame is saved, the aperture and area of the previous frame are stored, and the mean of the color channels is stored as -1. The counter for consecutive misses is incremented by 1 and the frame is stored the same way as previously mentioned, but the image plot receives the title of “*Missed Frame*” and no data is plotted on top.

In case that the frame was not valid and the number consecutive misses is equal or greater than the maximum number of consecutive misses allowed, the *previousThreshold*, *previousAperture* and *previousArea* are set to its default -1 and the name of the frame is stored in a discarded frames cell.

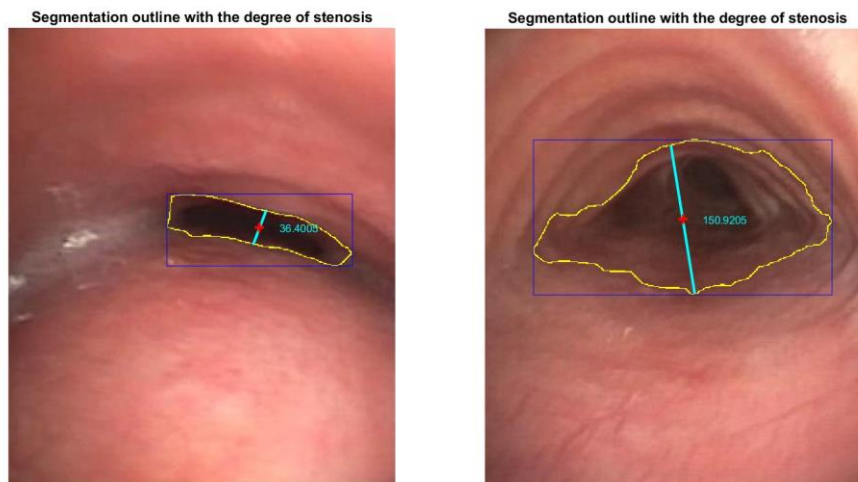


Figure 19. Examples of valid frames stored with results printed. Source: Own creation

- **Diagnosis**

To give a diagnosis, all the apertures of the frames calculated are used in order to calculate the percentage of stenosis of the airway.

First of all, in case of there being any kind of outliers on the results due to any erratic frame, a sliding window of 30% the range of the number of total frames analyzed is applied, removing all the elements that are more than three local scaled median absolute deviations (MAD [14]) away from the local median over the window.

Where the median absolute deviation is defined as:

$$\text{MAD} = \text{median}(|A_i - \text{median}(A)|)$$

for $i = 1, 2, \dots, N$.

While the scaled MAD is defined as

$$c * \text{median} \left(\text{abs}(A - \text{median}(A)) \right),$$

$$\text{where } c = -1/(\text{sqrt}(2) * \text{erfcinv}(3/2)).$$

And *erfcinv* is inverse complementary error function.

Then, the local maximas of the apertures, without outliers, are gathered in order to obtain the minimal and maximal peaks to find the percentage of stenosis of the trachea.

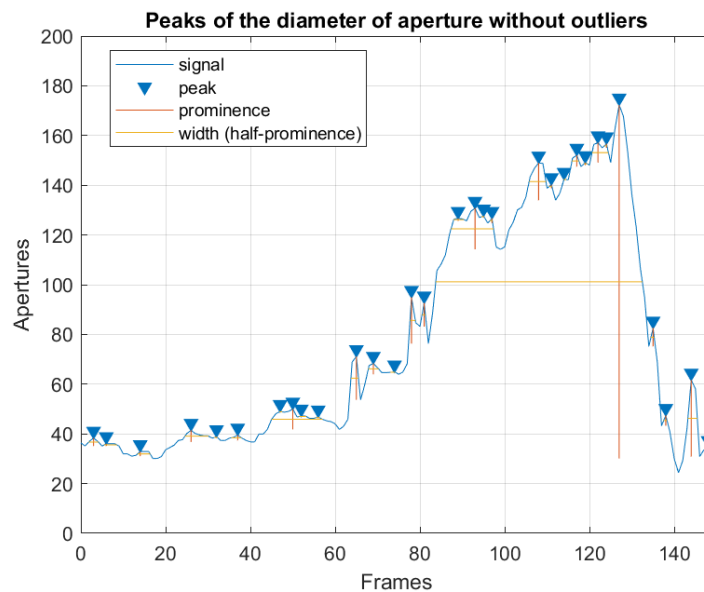


Figure 20. Aperture peaks without outliers

4.4 CREATING THE RESULTS VIDEO

Once the *mainProgram* script has been executed, the *getVideo* script can be used with the desired folder, storing the result frames, as the path for the creation of the video. The script will ask the user for the path of said folder and then will automatically create a video at the same path, with the name of the folder and the *.avi* extension.

5 EXPERIMENTS AND PERFORMANCE

To detect which are the segmentation algorithms that give the most stable and approximate results from the three types of segmentation algorithms implemented, and to evaluate the performance, a series of experiments have been done with each of the patients' videos provided by the hospital.

5.1 BINARIZATION BASED SEGMENTATION ALGORITHMS AND BEST FEATURES

In this section, thanks to a series of experiment comparisons between the binarization algorithms, we detect which is the one that better fits the purpose at hand and we see what are the results gathered by applying said algorithm to different chunks of videos.

5.1.1 Segmentation comparison

As we can see, by the results gathered in *table 9* (using gray level and no preprocessing), the binarization algorithms that seem to work better are our own implementation of a binarization algorithm, the adaptive binarization, and Niblack and Sauvola's binarization.

Analyzing the results gathered by these 4 algorithms, we can see that the one that gives the most approximate result is our implementation of binarization, due to the fact that the other three tend to overestimate the aperture of the airways.

Looking at the results obtained in *table 10* we can see how the two more accurate levels are the gray and red level, but the red level seems to be more accurate while the gray level seems to overestimate a little bit, but both levels seem to work well, so there would not be a big difference between picking one or the other. In the case of this project, the decided level to use for the segmentation by binarization in the mixed bag segmentation is the red level, due to the fact that the walls of the trachea have a lot of red pigmentation.

Lastly, by looking at the results gathered by *table 11*, we can see that it seems that when using the preprocessing we get a little bit of overestimation due to the fact that the preprocessing blends the values of input matrix.

Binarization algorithms

Frame	Own implementation	Adaptive	Otsu	Niblack	Sauvola	Mean	Kapur
1	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>
112	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Missed Frame</p>
137	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Missed Frame</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Segmentation outline with the degree of stenosis</p>	<p>Missed Frame</p>	<p>Missed Frame</p>

Table 9. Binarization algorithms segmentation

Own binarization with different features


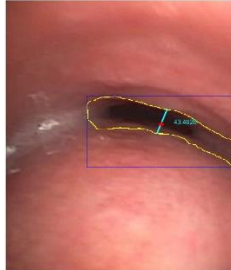
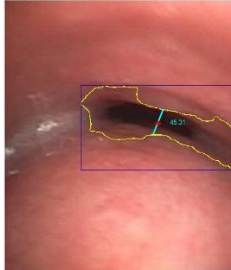

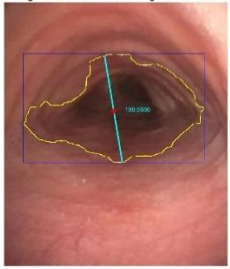
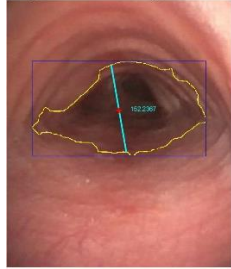
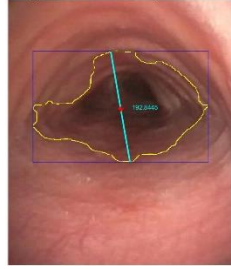
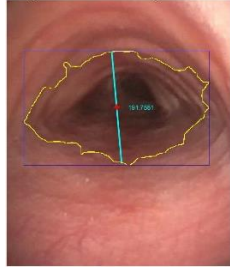
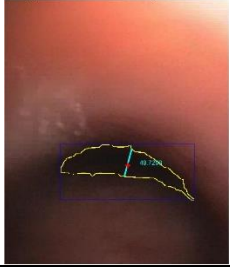



Frame	Gray level	Red level	Green level	Blue level
1	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 
112	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 
137	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 	<p style="font-size: small; text-align: center;">Segmentation outline with the degree of stenosis</p> 

Table 10. Own binarization with different features analysis. Source: Own creation

Own binarization (red level) with preprocessing on and off

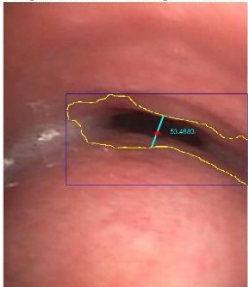
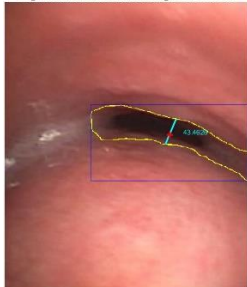




Frame	With Preprocessing	Without preprocessing
1	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 
112	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 
137	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 	<p style="text-align: center; font-size: small;">Segmentation outline with the degree of stenosis</p> 

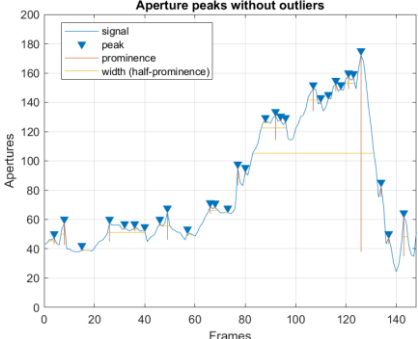
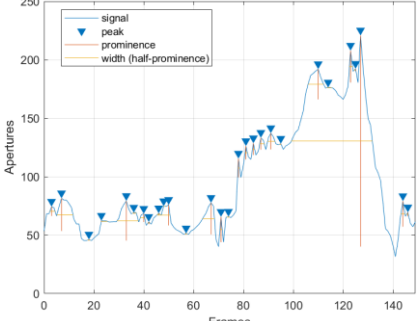
Table 11. Own binarization with and without preprocessing

5.1.2 Chunks of videos and performance

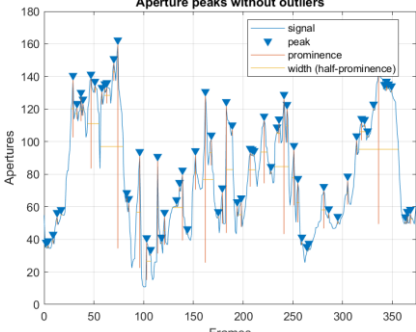
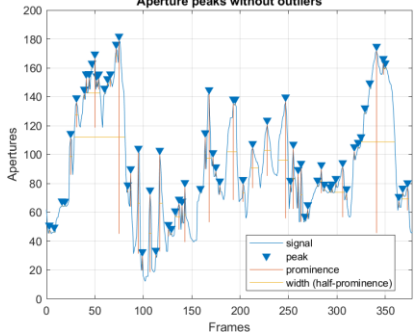
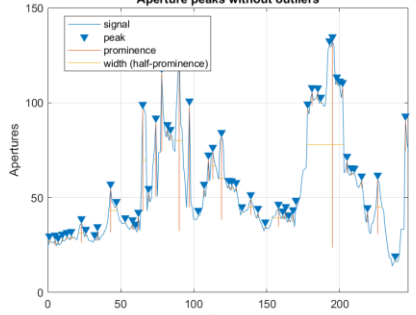
As we can see by the results obtained on *table 12*, there isn't a big difference on the results weather the preprocessing is on or off. As the results show, when preprocessing is on, the miss rate is a little bit higher than when its off, but It's negligible.

On the other hand, by the results gathered during the tests on the video of the normal patient, due to the rapid movement of the endoscope and drastic changes in light intensity the results aren't good. Using preprocessing seems to alleviate the error, but still there is too much error due to the rapid movement of the endoscope. Instead, if we compare the results gathered with the simulation of a "normal" airway stenosis during a short span of time (due to the fact that we are using one of the sick patients' videos) the percentage of difference in opening of the airway is lower than 30%, so the program would notify that the patient is healthy. That is due to the fact that during the frames of the simulation of a "normal" airway the camera is moving slowly and without drastic changes on the orientation of the endoscope so the information gathered is much better and feasible.

Results on chunks of videos

Video and chunk	Apertures	Time	Percentage of stenosis	Discarded and missed frames	Preprocessing
Sick1 - frames [1,150]	 <p>Aperture peaks without outliers</p> <p>Legend: signal (blue line), peak (blue triangle), prominence (orange line), width (half-prominence) (yellow line).</p>	63.9587 seconds = 0.43 seconds/frame	77.13%	0	Off
Sick1 - frames [1,150]	 <p>Aperture peaks without outliers</p> <p>Legend: signal (blue line), peak (blue triangle), prominence (orange line), width (half-prominence) (yellow line).</p>	69.9747 seconds = 0.47 seconds/frame	79.06%	0	On

EXPERIMENTS AND PERFORMANCE

<p>Sick1 – frames [357, 740]</p>		<p>173.6062 seconds = 0.45 seconds/frame</p>	<p>80.64%</p>	<p>7</p>	<p>Off</p>
<p>Sick1 – frames [357, 740]</p>		<p>185.6499 seconds = 0.48 seconds/frame</p>	<p>83.48%</p>	<p>8</p>	<p>On</p>
<p>Sick2 – frames [1, 255]</p>		<p>127.5865 seconds = 0.5 seconds/frame</p>	<p>87.33%</p>	<p>7</p>	<p>Off</p>

EXPERIMENTS AND PERFORMANCE

<p>Sick2 – frames [1, 255]</p>		<p>128.2580 seconds = 0.5 seconds/frame</p>	<p>84.49%</p>	<p>9</p>	<p>On</p>
<p>Sick2 – frames [292, 450]</p>		<p>79.6755 seconds = 0.5 seconds/frame</p>	<p>84.67%</p>	<p>7</p>	<p>Off</p>
<p>Sick2 – frames [292, 450]</p>		<p>81.4108 seconds = 0.51 seconds/frame</p>	<p>77.50%</p>	<p>6</p>	<p>On</p>

EXPERIMENTS AND PERFORMANCE

<p>Normal - frames [340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>	<p>Aperture peaks with outliers</p> <p>42.4329 seconds = 0.47 seconds/frame</p>	<p>83.32%</p>	<p>6</p>	<p>Off</p>
<p>Normal - frames[340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>	<p>Aperture peaks with outliers</p> <p>32.3094 seconds = 0.36 seconds/frame</p>	<p>68.63%</p>	<p>7</p>	<p>On</p>

<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>16.4865 seconds = 0.5 seconds/frame</p>	<p>14.52%</p>	<p>0</p>	<p>Off</p>
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>16.6848 seconds = 0.51 seconds/frame</p>	<p>28.41%</p>	<p>0</p>	<p>On</p>

Table 12. Results obtained applying own implementation of segmentation based on binarization to chunks of videos. Source: Own creation

5.2 REGION BASED SEGMENTATION ALGORITHM

In this section, using similar series of experiments as the previously shown, but in this case applied to k-means segmentation, we detect which feature is better (RGB or L*a*b color space), and gather the results of applying said feature on chunks of videos.

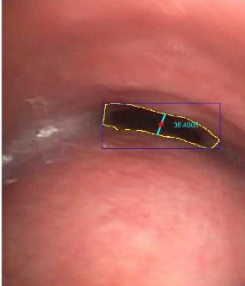
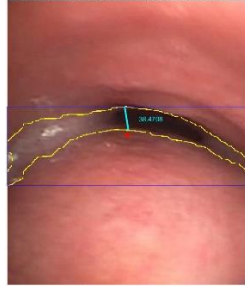
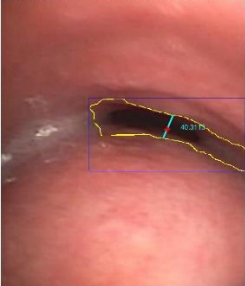
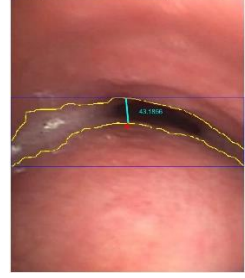

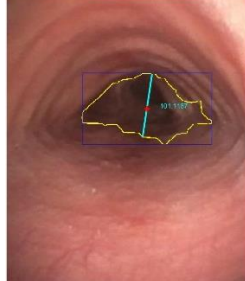
5.2.1 Feature comparison

As we can see by the results obtained in *table 13*, as the k-means algorithm uses the RGB coordinates for the clustering it doesn't take into account the light highlights reflecting on the camera from the liquid, so the results are more accurate (only takes the visible "hole" into account, while the L*a*b* implementation of k-means, as it only uses the a*b* coordinates, without taking into account the luminosity during the clustering, the results include the light reflecting on the liquid in the trachea.

Nevertheless, the results are close one to another (with or without preprocessing) so any one of the implementations could be used during the mixed bag segmentation. In the case of this project, mix bag segmentation uses the RGB features for the k-means segmentation.

Furthermore, k-means tends to underestimate the stenosis degree for bigger apertures, and tends to overestimate when there isn't much light. So only in the case for detecting little apertures with enough light the k-means algorithm will be used during the mix bag segmentation (as it is already stated during the mix bag segmentation algorithm explanation during section 4.3.2).

K-means segmentation features

Frame	RGB color space	L*a*b* color space	Preprocessing
1	<p data-bbox="779 371 1005 384">Segmentation outline with the degree of stenosis</p> 	<p data-bbox="1247 371 1473 384">Segmentation outline with the degree of stenosis</p> 	Off
1	<p data-bbox="779 686 1005 699">Segmentation outline with the degree of stenosis</p> 	<p data-bbox="1247 694 1473 707">Segmentation outline with the degree of stenosis</p> 	On
112	<p data-bbox="779 1000 1005 1013">Segmentation outline with the degree of stenosis</p> 	<p data-bbox="1247 1000 1473 1013">Segmentation outline with the degree of stenosis</p> 	Off

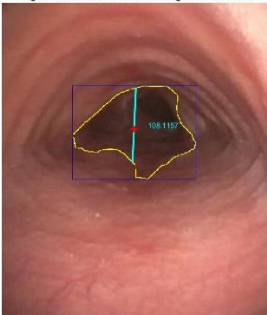
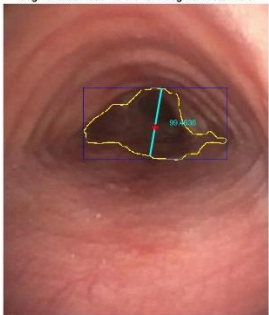
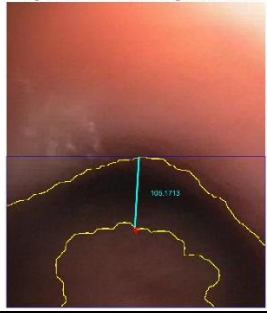
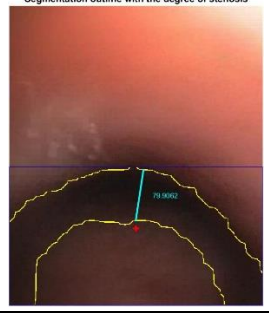
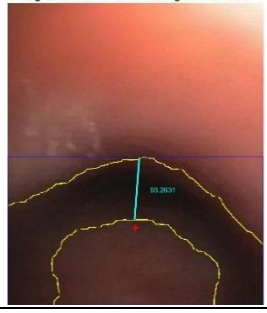
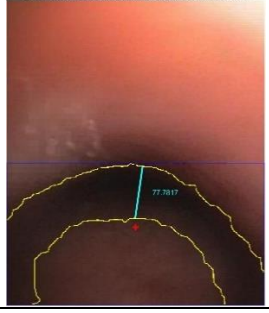
112	<p>Segmentation outline with the degree of stenosis</p> 	<p>Segmentation outline with the degree of stenosis</p> 	On
137	<p>Segmentation outline with the degree of stenosis</p> 	<p>Segmentation outline with the degree of stenosis</p> 	Off
137	<p>Segmentation outline with the degree of stenosis</p> 	<p>Segmentation outline with the degree of stenosis</p> 	On

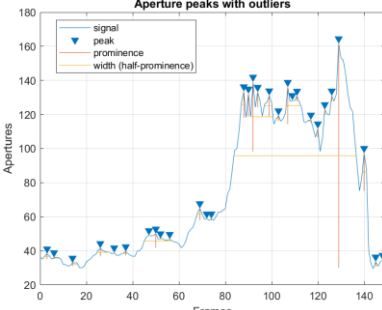
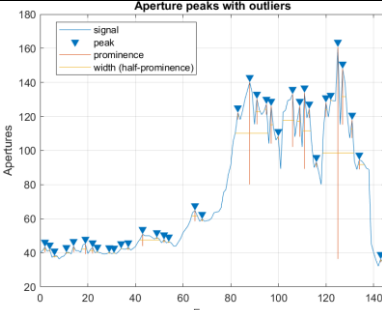
Table 13. K-means segmentation features. Source: Own creation

5.2.2 Chunks of videos and performance

As we can see by the results gathered in *table 14*, using preprocessing and not using it can change a little bit the percentage of stenosis, this could be because short chunks of videos can give a big difference on the stenosis degree due to the fact that any minor change on the segmentation mask can alter the aperture of the segmented frame.

On the other hand, the execution times are much higher than before due to a more complex computation. Still, the percentage of stenosis degree for the normal patient video chunk gives us a better output than using a binarization based algorithm, and the simulated normal video chunk gives us close results to the previously mentioned algorithm.

K-means applied to chunks of videos

Video and chunk	Apertures	Time	Percentage of stenosis	Discarded and missed frames	Preprocessing
Sick1 - frames [1,150]	 <p style="font-size: small;">Aperture peaks with outliers</p> <p style="font-size: x-small;">Legend: signal (blue line), peak (blue triangle), prominence (orange line), width (half-prominence) (yellow line)</p>	105.8639 seconds = 0.71 seconds/frame	79.65%	0	Off
Sick1 - frames [1,150]	 <p style="font-size: small;">Aperture peaks with outliers</p> <p style="font-size: x-small;">Legend: signal (blue line), peak (blue triangle), prominence (orange line), width (half-prominence) (yellow line)</p>	102.7583 seconds = 0.69 seconds/frame	77.46%	0	On

EXPERIMENTS AND PERFORMANCE

<p>Sick2 – frames [1, 255]</p>		<p>172.9951 seconds = 0.68 seconds/frame</p>	<p>74.50%</p>	<p>6</p>	<p>Off</p>
<p>Sick2 – frames [1, 255]</p>		<p>167.5553 seconds = 0.66 seconds/frame</p>	<p>72.54%</p>	<p>6</p>	<p>On</p>
<p>Normal - frames [340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>60.4959 seconds = 0.66 seconds/frame</p>	<p>53.39%</p>	<p>5</p>	<p>Off</p>

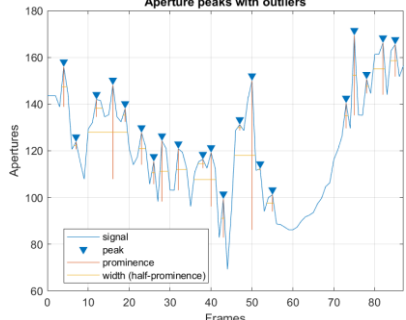
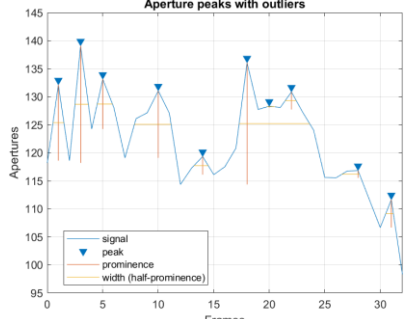
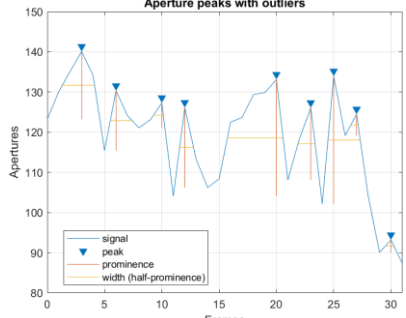
<p>Normal - frames[340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>59.1277 seconds = 0.65 seconds/frame</p>	<p>56.26%</p>	<p>5</p>	<p>On</p>
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>23.8648 seconds = 0.72 seconds/frame</p>	<p>19.73%</p>	<p>0</p>	<p>Off</p>
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>22.0681 seconds = 0.67 seconds/frame</p>	<p>33.48%</p>	<p>0</p>	<p>On</p>

Table 14. RGB k-means applied to chunks of videos

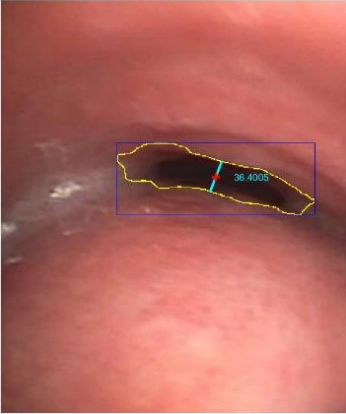
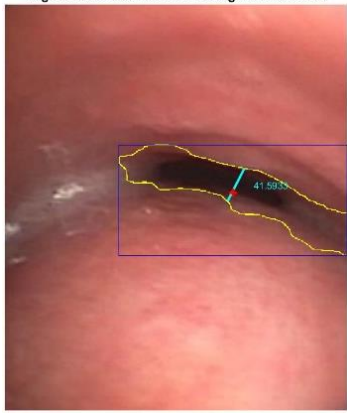
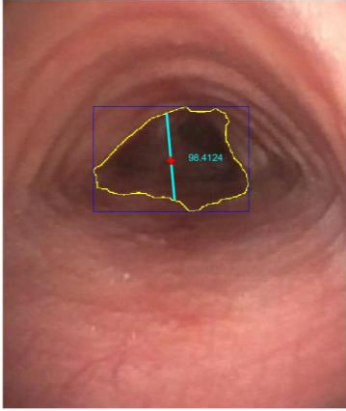

5.3 EDGE BASED SEGMENTATION ALGORITHM

In this section, as previously done medium a series of experiments, we will analyze the segmentation results of applying watershed segmentation with markers based on k-means segmentation, and see what is the performance and results of applying the algorithm to different chunks of videos.

5.3.1 Segmentation results

Looking at the results of *table 15*, obtained by applying the k-means segmentation algorithm to those 3 frames, we can see that the only big difference between using preprocessing and not is when the angle of aperture of the airway is more opened, the cases with a smaller aperture or with low light intensity bring similar results.

Watershed segmentation

Frame	With preprocessing	Without preprocessing
1	<p data-bbox="969 395 1279 411">Segmentation outline with the degree of stenosis</p> 	<p data-bbox="1592 395 1901 411">Segmentation outline with the degree of stenosis</p> 
112	<p data-bbox="969 842 1279 858">Segmentation outline with the degree of stenosis</p> 	<p data-bbox="1592 842 1901 858">Segmentation outline with the degree of stenosis</p> 

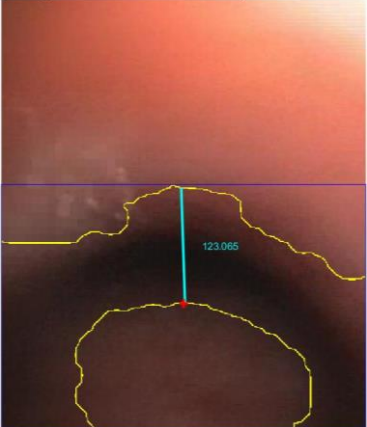
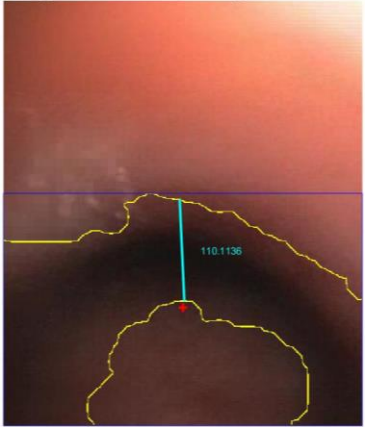
<p>137</p>	<p>Segmentation outline with the degree of stenosis</p>  <p>123.065</p>	<p>Segmentation outline with the degree of stenosis</p>  <p>110.136</p>
------------	---	--

Table 15. Watershed segmentation with markers based on k-means segmentation

5.3.2 Chunks of videos and performance

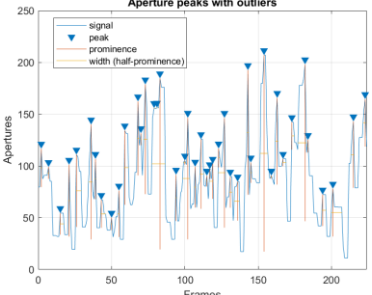
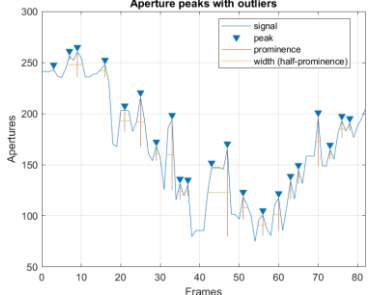
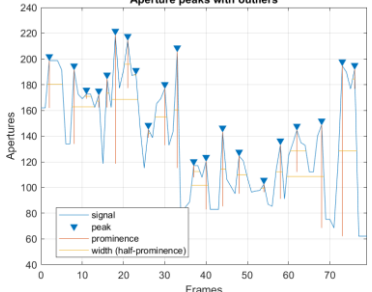
As the results shown in *table 16*, this algorithm performs in time, results and miss rate worse than the two previously analyzed, that's, why it is used during the mix bag segmentation algorithm as a third and last option, in case the other two algorithms don't give a valid output.

As the table clearly shows, it seems that the no preprocessed execution of the algorithm performs a little bit better than the preprocessed execution, and the resulting aperture values seem to diverge less between them (there aren't as much sudden spikes as the preprocessed version).

Watershed segmentation applied to chunks of videos

Video and chunk	Apertures	Time	Percentage of stenosis	Discarded and missed frames	Preprocessing
Sick1 - frames [1,150]		141.0197 seconds = 0.94 seconds/frame	81.65%	11	Off
Sick1 - frames [1,150]		131.6440 seconds = 0.88 seconds/frame	77.97%	15	On
Sick2 – frames [1, 255]		226.5498 seconds = 0.89 seconds/frame	81.85%	74	Off

EXPERIMENTS AND PERFORMANCE

<p>Sick2 – frames [1, 255]</p>		<p>212.1012 seconds = 0.83 seconds/frame</p>	<p>75.84%</p>	<p>85</p>	<p>On</p>
<p>Normal - frames [340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>81.88 seconds = 0.9 seconds/frame</p>	<p>61.35%</p>	<p>11</p>	<p>Off</p>
<p>Normal - frames[340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>76.2066 seconds = 0.84 seconds/frame</p>	<p>53.03%</p>	<p>15</p>	<p>On</p>

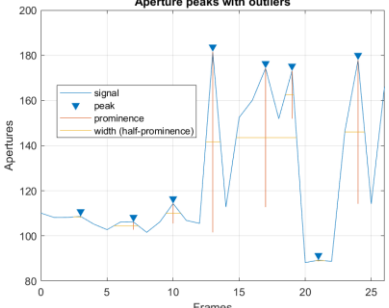
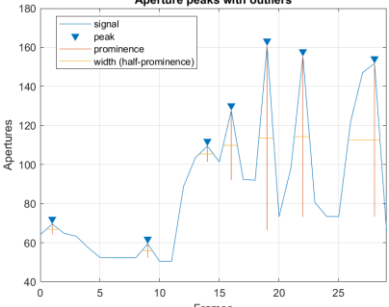
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>33.3528 seconds = 1.01 seconds/frame</p>	<p>50.84%</p>	<p>1</p>	<p>Off</p>
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>28.43 seconds = 0.86 seconds/frame</p>	<p>63.05%</p>	<p>0</p>	<p>On</p>

Table 16. Watershed segmentation applied to chunks of videos. Source: Own creation

5.4 MIX BAG SEGMENTATION

In this section the performance and results of the mix bag segmentation algorithm, which as previously stated uses a mix of our own binarization algorithm, k-means clustering based on the RGB color space, and watershed segmentation.

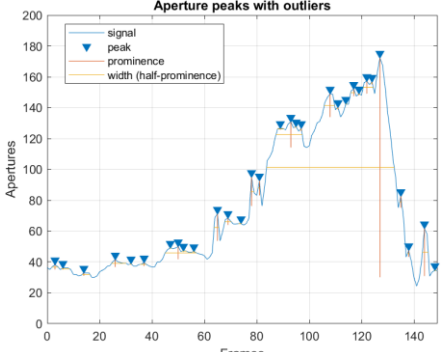
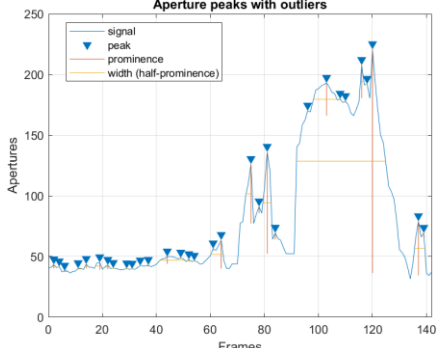
5.4.1 Chunks of videos and performance

As we can see by the results obtained in *table 17*, the mix bag segmentation algorithm normally has a less abrupt change on the apertures using its preprocessed version than using its non-preprocessed counterpart, but at the same time, its non-processed version seem to bring a little more accurate results.

The results gathered during the simulated “normal” chunk of video, are very close to the ones gathered using our own binarization based segmentation due to the fact that, for bigger apertures, the principal algorithm used during the mix bag segmentation is the same one because shows better results for this kind of segmentation as previously commented.

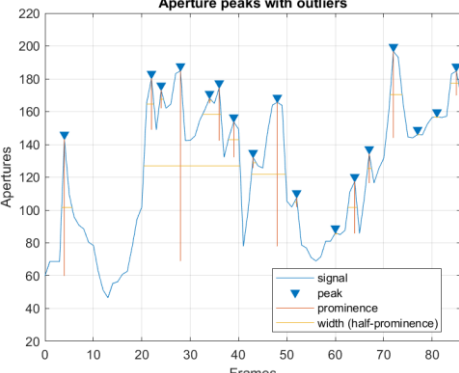
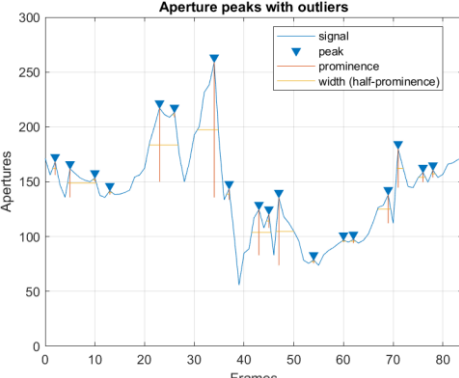
As we can see by the times gathered this is the algorithm that takes the longest due to the fact that it calls the previous three segmentation algorithms for each iteration.

Mixed bag segmentation applied to chunks of videos

Video and chunk	Apertures	Time	Percentage of stenosis	Discarded and missed frames	Preprocessing
Sick1 - frames [1,150]		182.0747 seconds = 1.21 seconds/frame	80.91	0	Off
Sick1 - frames [1,150]		176.4709 seconds = 1.18 seconds/frame	82.66%	6	On

EXPERIMENTS AND PERFORMANCE

<p>Sick2 – frames [1, 255]</p>	<p>Aperture peaks with outliers</p> <p>Legend: signal (blue line), peak (blue triangle), prominence (red vertical line), width (half-prominence) (yellow vertical line)</p>	<p>269.9 seconds = 1.01 seconds/frame</p>	<p>78.22%</p>	<p>1</p>	<p>Off</p>
<p>Sick2 – frames [1, 255]</p>	<p>Aperture peaks with outliers</p> <p>Legend: signal (blue line), peak (blue triangle), prominence (red vertical line), width (half-prominence) (yellow vertical line)</p>	<p>275.246 seconds = 1.07 seconds/frame</p>	<p>80.11%</p>	<p>0</p>	<p>On</p>

<p>Normal - frames [340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>85.618 seconds = 1.2 seconds frame</p>	<p>56.22%</p>	<p>3</p>	<p>Off</p>
<p>Normal - frames[340, 430] (the only section that seems to have less movement, and still, it has too much movement)</p>		<p>78.4295 seconds = 0.86 seconds/frame</p>	<p>69.60%</p>	<p>3</p>	<p>On</p>

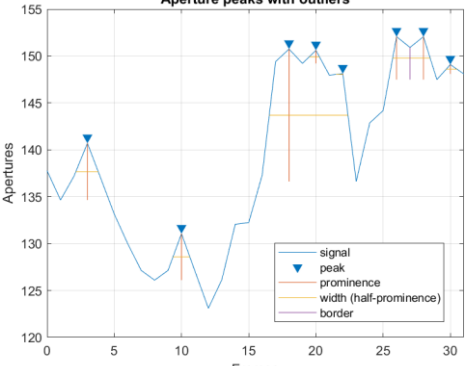
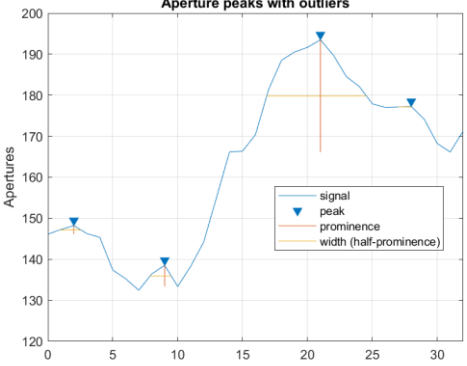
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>33.997 seconds = 1.03 seconds/frame</p>	<p>13.81%</p>	<p>0</p>	<p>Off</p>
<p>Sick1 - frames [90, 122] (Simulating a “normal” trachea with an steady camera)</p>		<p>28.0324 seconds = 0.85 seconds/frame</p>	<p>28.42%</p>	<p>0</p>	<p>On</p>

Table 17. Mix bag segmentation algorithm applied to video chunks

6 CONCLUSIONS AND FUTURE WORK

6.1 CONCLUSIONS

Looking back at all the development of this project, we can say that it isn't an easy task to give an exact percentage of the degree of stenosis of the airways solely using segmentation algorithms and relevant information from its segmentation, due to the fact that this project is, first and foremost, based only on three videos in order to detect the tracheobronchomalacia disease, one of which (the normal patient one) has a lot of fast movements of the bronchoscope and many occlusions of the camera making the extraction of useful marks harder for the segmentation algorithms and the video nearly unusable.

Another important point to talk about is that because the mix bag segmentation algorithm uses three different types of algorithms for each iteration, the execution time increases drastically compared to using only one algorithm, so if time is of the essence and the number of frames to analyze is very wide, using only the binarized segmentation or k-means segmentation would be better.

Taking into account the results gathered from this project, we can say that given a bronchoscopy with steady movement of the camera or none at all, and a good focus of the airway, using our own binarization based segmentation algorithm, k-means or mix bag segmentation, an accurate diagnosis of the degree of stenosis of the trachea is given and so, a diagnosis for tracheobronchomalacia is possible.

6.2 FUTURE WORK

There is plenty of room for improvement of this project, for example by expanding more the possible decisions to take during the mix bag segmentation depending on the information of the frame or by taking more characteristics into account during the segmentation, for example using more the color of the frames in order to make better decisions.

Or maybe, instead of using segmentation algorithms, if given a big data set of videos and photos from patients suffering from tracheobronchomalacia and healthy patients, an implementation with image classification could be put in place in order to detect frames that show a very low stenosis degree.

I just started computer vision the last quarter, and my intention is to keep on growing my knowledge of it in order to make better decisions during future projects and, if possible, keeping on expanding and improving this one.

BIBLIOGRAPHY

- [1] J. Gao, Y. Yang, P. Lin and D. S. Park, "Computer Vision in Healthcare Applications", *Journal of Healthcare Engineering*, 2018. [Online]. Available: <https://doi.org/10.1155/2018/5157020>. [Accessed 24 February 2022].
- [2] "IBM," [Online]. Available: <https://www.ibm.com/topics/computer-vision>. [Accessed 24 February 2022].
- [3] NORD, "rarediseases.org," 2021. [Online]. Available: <https://rarediseases.org/rare-diseases/tracheobronchomalacia/>. [Accessed 24 February 2022].
- [4] D. Gil, R. M. Ortiz, C. Sánchez and A. Rosell, "Objective Endoscopic Measurements of Central Airway Stenosis: A Pilot Study," 2017.
- [5] A. Hayashi, S. Takanashi, T. Tsushima, J. Denpoya, K. Okumura and K. Hirota, "New method for quantitative assessment of airway calibre using a stereovision fiberoptic bronchoscope," *British Journal of Anaesthesia*, p. 5, 2011.
- [6] K. Harris, "Quantifying Central Airway Obstruction during Therapeutic Bronchoscopy," *The Proceduralist*, p. 4, 2017.
- [7] MATLAB, "mathworks," 1994 - 2022. [Online]. Available: <https://es.mathworks.com/discovery/what-is-matlab.html>. [Accessed 26 February 2022].
- [8] "Tusalarario.es," [Online]. Available: <https://tusalarario.es/salario/comparatusalarario#/>. [Accessed 12 March 2022].
- [9] "bonpreuescal.cat," [Online]. Available: <https://www.bonpreuesclat.cat/es/plans-energia>. [Accessed 12 March 2022].
- [10] MATLAB, "mathworks," MathWorks, 1994 - 2022. [Online]. Available: <https://www.mathworks.com/help/images/ref/adaptthresh.html#bu2a9fp-1-sensitivity>. [Accessed 10 June 2022].
- [11] D. A. Greensted, "The Lab Book Pages," 17 June 2010. [Online]. Available: <http://www.labbookpages.co.uk/software/imgProc/otsuThreshold.html>. [Accessed 10 June 2022].
- [12] L. Chunbo and J. Heming, "MDPI," 23 March 2019. [Online]. Available: <https://www.mdpi.com/1099-4300/21/3/318/htm#B12-entropy-21-00318>. [Accessed 10 June 2022].
- [13] MATLAB, "mathworks," 1994 - 2022. [Online]. Available: <https://www.mathworks.com/help/images/ref/regionprops.html#buoixjn-1-properties>. [Accessed 12 June 2022].

- [14] MATLAB, "mathworks," 1994 - 2022. [Online]. Available: <https://www.mathworks.com/help/matlab/ref/fitoutliers.html#bvml247>. [Accessed 12 June 2022].
- [15] MATLAB, "mathworks," MathWorks, 1994 - 2022. [Online]. Available: <https://www.mathworks.com/help/images/ref/strel.html>. [Accessed 10 June 2022].

LIST OF FIGURES

Figure 1: Example of usage of intelligent character recognition on human writing. Source: Google	7
Figure 2: Comparison between a normal airway (up), and a tracheobronchomalacia patient. Source: Cleveland Clinic	8
Figure 3: Image showing the trachea and the bronchi. Source: pediaa.com	9
Figure 4. Object detection of a self-driving car. Source: Unbundling The Autonomous Vehicle, cbinsights.....	31
Figure 5. Original patient frame and cropped frame. Source: Own creation.....	33
Figure 6. RGB channels and gray scale. Source: Own creation	34
Figure 7. Erosion, Open by Reconstruction, Dilation and Open/Close by Reconstruction of the frame being treated. Source: Own creation	34
Figure 8. Histogram with a total of 10 bins. Source: Own creation.....	36
Figure 9. Otsu thresholding. Source: Wikipedia.....	37
Figure 10. L*a*b* color space representation. Source: Google.....	40
Figure 11. Labeled RGB image with the k clusters and final mask. Source: Own creation	40
Figure 12. K-means labeled a*b* image and final resulting mask. Source: Own creation	41
Figure 13. Inner-most and outer-most markers imposed on frame. Source: Own creation	42
Figure 14. Image gradient, gradient with markers and watershed ridge lines. Source: Own creation.....	42
Figure 15. Ridge lines and markers imposed on frame, Watershed labels and Watershed final mask. Source: Own creation	43
Figure 16. Segmented frame without treatment and final blob.....	44
Figure 17. Ellipse that englobes a blob. Source: MathWorks.....	45
Figure 18. Bounding box, aperture, centroid and aperture diameter of a blob. Source: Own creation.....	46
Figure 19. Examples of valid frames stored with results printed. Source: Own creation	47
Figure 20. Aperture peaks without outliers	48

LIST OF TABLES

Table 1: Summary table with time estimations. Source: Own creation	17
Table 2. Software resources cost	21
Table 3. Hardware resources cost.....	22
Table 4. Tasks assigned to each role	23
Table 5. Human resources cost.....	23
Table 6. Indirect costs	24
Table 7. Unexpected costs	25
Table 8. Total budget of the project.....	25
Table 9. Binarization algorithms segmentation	50
Table 10. Own binarization with different features analysis. Source: Own creation	51
Table 11. Own binarization with and without preprocessing	52
Table 12. Results obtained applying own implementation of segmentation based on binarization to chunks of videos. Source: Own creation.....	58
Table 13. K-means segmentation features. Source: Own creation	61
Table 14. RGB k-means applied to chunks of videos	65
Table 15. Watershed segmentation with markers based on k-means segmentation.....	68
Table 16. Watershed segmentation applied to chunks of videos. Source: Own creation	72
Table 17. Mix bag segmentation algorithm applied to video chunks	77

LIST OF CHARTS

Chart 1. Gantt chart of the project's planning. Source: Own creation.....	19
---	----

APPENDIX

GET FRAMES SCRIPT

```
clear
clc

prompt = 'Insert the video file path: ';
inputStr = input(prompt,'s')

% I get the name of the video file to use it as the name of the output
% folder
tokens = regexp(inputStr,'(\w+)\.\.+$', 'tokens');
outputFolder = tokens{1}{1};

if exist(outputFolder, 'dir')
    rmdir(outputFolder,'s');
end
mkdir(outputFolder);

% Importing the video file
obj = VideoReader(inputStr);
vid = read(obj);

% Get the total number of frames
numFrames = obj.NumFrames;

% File format of the frames to be saved in
ST = '.jpg';

% Reading and writing the frames
for x = 1 : numFrames
    num = num2str(x);
    frameName = strcat(num,ST);
    frame = vid(:, :, :, x);

    imwrite(frame,[outputFolder '\' frameName]);
end
```

Script 1. Script used to segmentate patients' videos in frames

MAIN PROGRAM SCRIPT

```

clear
clc

% ////////////////////////////////// INPUT //////////////////////////////////

prompt = 'Type in the name of the folder with the frames: ';
folderName = input(prompt, 's');

prompt = ['Choose the segmentation algorithm:\n      ' ...
         '1) Own implementation of segmentation based on histogram binarization\n      ' ...
         '2) Segementation based on adaptive binarization\n      ' ...
         '3) Otsu Binarization\n      ' ...
         '4) Niblack Binarization\n      ' ...
         '5) Sauvola Binarization\n      ' ...
         '6) Mean value Binarization\n      ' ...
         '7) Kapur entropy Binarization\n      ' ...
         '8) Kmeans segmentation\n      ' ...
         '9) Watershed segmentation with markers based on color using kmeans\n      ' ...
         '10) Mixed bag\n\n'];
implementation = input(prompt);

folderPath = '';
filePath = '';
switch implementation
    case 1
        folderPath = ['ownResults_' folderName]
        filePath = [folderPath '\ownBinarization_']
    case 2
        folderPath = ['adaptiveResults_' folderName]
        filePath = [folderPath '\adaptiveBinarization_']
    case 3
        folderPath = ['otsuResults_' folderName]
        filePath = [folderPath '\otsuBinarization_']
    case 4
        folderPath = ['niblackResults_' folderName]
        filePath = [folderPath '\niblackBinarization_']
    case 5
        folderPath = ['sauvolaResults_' folderName]
        filePath = [folderPath '\sauvolaBinarization_']
    case 6
        folderPath = ['meanResults_' folderName]
        filePath = [folderPath '\meanBinarization_']
    case 7
        folderPath = ['kapurResults_' folderName]
        filePath = [folderPath '\kapurBinarization_']
    case 8
        folderPath = ['kmeansResults_' folderName]

```

```

        filePath = [folderPath '\kmeansSegmentation_']
    case 9
        folderPath = ['watershedResults_' folderName]
        filePath = [folderPath '\watershedSegmentation_']
    case 10
        folderPath = ['mixBagResults_' folderName]
        filePath = [folderPath '\mixBagSegmentation_']
    otherwise
        errordlg('Choose a valid segmentation algorithm','Input Error')
        return
end

% Feature Choosing
if implementation > 0 && implementation < 8
    prompt = ['Choose the segmentation feature: \n      ' ...
            '1) Gray level\n      ' ...
            '2) Red level\n      ' ...
            '3) Green level\n      ' ...
            '4) Blue level\n'];
elseif implementation == 8
    prompt = ['Choose the segmentation feature: \n      ' ...
            '5) RGB color space\n      ' ...
            '6) L*a*b* color space\n\n'];
end

% Case of mixbag implementation
if implementation >= 9 % Watershed and mix bag
    feature = 5;
else
    feature = input(prompt);
end

% Preprocessing of the image
prompt = 'Do you want to use preprocessing of the frame? (y/n)';
preproc = input(prompt,"s");

% Folder creation to store results
if exist(folderPath, 'dir')
    rmdir(folderPath,'s');
end
mkdir(folderPath);

% Desired frame range
prompt = 'Choose the number of the initial frame: ';
initialFrame = input(prompt);
prompt = 'Choose the number of the final frame: ';
finalFrame = input(prompt);

```

```

% Declaration of variables and default values
global S frameIx previousThreshold previousCentroid previousArea previousAperture
apertureThreshold
S = dir(fullfile(['.\' folderName '\'], '*.jpg'));
S = natsortfiles(S);
% Variables
frames = {};
discardedFrames = {};

apertures = [];
aperture = -1;
areas = [];
meanReds = [];
meanGreens = [];
meanBlues = [];

previousThreshold = -1;
previousCentroid = [0 0];
previousArea = -1;
previousAperture = -1;
apertureThreshold = 0.8; % Scalar from 0 to 1

minimumAperture = 7;

maximumConsecutiveMisses = 2;
consecutiveMisses = 0; % Counter of the number of consecutive misses

first = true;

tic
% initializing progress bar
f = waitbar(0, 'Obtaining frames...');
for frameIx = initialFrame:finalFrame

    frameName = fullfile(['.\' folderName], S(frameIx).name)
    frame = imread(frameName);

    % updating progress bar
    percentage = (frameIx - initialFrame)/(finalFrame - initialFrame);
    waitbar(percentage, f, 'Processing frames...');

%     figure
%     imshow(frame);
%     title('Original Frame');

% ////////// PREPROCESSING \\\\\\\\\\\
% ----- Image Cropping -----

```

```

frameCropped = imcrop(frame,[168,54,387,458]);

% figure
% imshow(frameCropped);
% title('Cropped Frame');

% -----

% Variables actualization
validFrame = true;
% Setting the previous centroid to the center of the image for the first treated
% frame
if first
    previousCentroid = [size(frameCropped,1)/2,size(frameCropped,2)/2];
    first = false;
end

% ----- Channels -----
redChannel = frameCropped(:,:,1);
greenChannel = frameCropped(:,:,2);
blueChannel = frameCropped(:,:,3);

% Create an all black channel.
allBlack = zeros(size(frameCropped, 1), size(frameCropped, 2), 'uint8');
% Create a color version of each Channel
frameRed = cat(3, redChannel, allBlack, allBlack);
frameGreen = cat(3, allBlack, greenChannel, allBlack);
frameBlue = cat(3, allBlack, allBlack, blueChannel);
% -----

% ----- Gray scale -----
frameGray = rgb2gray(frameCropped);
% -----

% ----- Prints -----
% Print red channel
% figure
% imshow(frameRed);
% title('Red Channel only');
% Print green channel
% figure
% imshow(frameGreen);
% title('Green Channel only');
% Print blue channel
% figure
% imshow(frameBlue);
% title('Blue Channel only');
% Gray scale print
% figure
% imshow(frameGray);
% title('Gray scale');
% -----

preProcessedFrame = [];

```



```

switch implementation
case 1
    try
        % Own binarization
        binFrame = binBinarization(preProcessedFrame,10,preproc);
    catch
        sprintf('Error during binBinarization in frame %d',frameIx)
        validFrame = false;
    end
case 2
    try
        % Adaptive binarization
        binFrame = imbinarize(preProcessedFrame,"adaptive", ...
            "ForegroundPolarity","bright","Sensitivity",0.66);
        binFrame = ~binFrame;
    catch
        sprintf('Error during adaptiveBinarization in frame %d',frameIx)
        validFrame = false;
    end
case 3
    try
        % Otsu binarization
        level = graythresh(preProcessedFrame);
        binFrame = ~imbinarize(preProcessedFrame,level);
    catch
        sprintf('Error during Otsu binarization in frame %d',frameIx)
        validFrame = false;
    end
case 4
    try
        % Niblack binarization
        binFrame = ~niblack(preProcessedFrame, ...
            [ceil(size(preProcessedFrame,2)*0.6),ceil(size(preProcessedFrame,1)*0.6)]);
    catch
        sprintf('Error during Otsu binarization in frame %d',frameIx)
        validFrame = false;
    end
case 5
    try
        % Sauvola binarization
        binFrame = ~sauvola(preProcessedFrame, ...
            [ceil(size(preProcessedFrame,2)*0.6),ceil(size(preProcessedFrame,1)*0.6)]);
    catch
        sprintf('Error during Sauvola binarization in frame %d',frameIx)
        validFrame = false;
    end
case 6
    try
        % Mean value binarization
        level = mean2(preProcessedFrame);
        binFrame = preProcessedFrame <= level;
    catch

```



```

        sprintf('Error during SauvolaRidler-Calvard binarization in frame
%d',frameIx)
        validFrame = false;
    end
case 7
    try
        % Kapur entropy binarization
        binFrame = ~kapur(preProcessedFrame);
    catch
        sprintf('Error during Kapur entropy binarization in frame %d',frameIx)
        validFrame = false;
    end
case 8
    try
        % kmeans segmentation
        if feature == 6
            binFrame = kmeansLabSegmentation(preProcessedFrame,frameCropped,8);
        elseif feature == 5
            binFrame = kmeansRGBSegmentation(preProcessedFrame,8,0);
        end
    catch
        sprintf('Error during Kmeans segementation in frame %d',frameIx)
        validFrame = false;
    end
case 9
    try
        % watershed segmentation using kmeans to extract markers
        binFrame = watershedSegmentationKmeans(preProcessedFrame, preproc);
    catch
        sprintf('Error during watershed segementation in frame %d',frameIx)
        validFrame = false;
    end
case 10
    try
        % mixBag Segmentation (combination of various techniques)
        [closeDiffBlobFrame, aperture, cols, rows, centroid, area, bbox, midpoint]
= mixBagSegmentation( ...
            preProcessedFrame, frameCropped, preproc);
    catch
        sprintf('Error during mix-bag segmentation in frame %d',frameIx)
        validFrame = false;
    end
end

% -----
% \\\\\\\\\\\\\\\\\\\\\\\|\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\\

% ////////// POSTPROCESSING \\\\\\\\\\\\\\\\\\\\\\\
try
    if validFrame && implementation ~= 10 % Not mix bag
        closeDiffBlobFrame = getSegment(binFrame);
    end
catch

```



```

previousArea = maxDiffArea;
consecutiveMisses = 0;

% Plot of all the data on top of the original image
edgeDetected = edge(closeDiffBlobFrame,"canny");
frameOutlined = imoverlay(frameCropped,edgeDetected,'yellow');
firstFig = figure('visible','off');
imshow(frameOutlined);

hold on;
plot(cols,rows,'c','LineWidth',2)
plot(centroid(1), centroid(2), 'r+', 'LineWidth', 2, 'MarkerSize', 7)
rectangle('Position',bbox,"EdgeColor",'b')
title('Segmentation outline with the degree of stenosis');
text(midpoint(2) + 20,midpoint(1),string(aperture),'Color',[0 1 1],'FontSize',8)
saveas(firstFig,[pwd ['\ ' filePath] S(frameIx).name]);

elseif consecutiveMisses < maximumConsecutiveMisses
frames{end + 1} = S(frameIx).name;
apertures = [apertures previousAperture];
areas = [areas previousArea];

meanReds = [meanReds -1];
meanGreens = [meanGreens -1];
meanBlues = [meanBlues -1];

consecutiveMisses = consecutiveMisses + 1;

firstFig = figure('visible','off');
imshow(frameCropped);
title('Missed Frame');
saveas(firstFig,[pwd ['\ ' filePath] S(frameIx).name]);

else
previousThreshold = -1;
previousAperture = -1;
previousArea = -1;

discardedFrames{end + 1}= S(frameIx).name;
consecutiveMisses = consecutiveMisses + 1;
end
% ----- %
end

% closing progress bar
close(f)

% //////////// RESULTS \\\\\\\\\\\\\\\\\\\
T = table(frames',apertures', areas', meanReds',meanGreens',meanBlues')

figure

```

```

plot(apertures)
title('Diameter of aperture (in pixels)')
xlabel('Frames')
ylabel('Apertures')

aperturesWout = rmoutliers(apertures,'movmedian',(finalFrame - initialFrame)*0.3);
plot(aperturesWout)
title('Diameter of aperture without outliers')
xlabel('Frames')
ylabel('Apertures')

% Finding peaks
peaksIx = [1:size(aperturesWout)];
[peaks, locs, p] = findpeaks(aperturesWout,peaksIx);

figure
findpeaks(aperturesWout,peaksIx,'Annotate','extents')
title('Aperture peaks with outliers')
xlabel('Frames')
ylabel('Apertures')

figure
plot(locs,peaks)
title('Peaks of the apertures only')
xlabel('Frames')
ylabel('Apertures')

% Getting the biggest difference in the peaks
[minVal, minIx] = min(peaks);
[maxVal, maxIx] = max(peaks);

% Getting the difference on the degree of stenosis
degreeDiff = 100 - (minVal*100 / maxVal);

discardedAndMissedFrames = size(discardedFrames,1) + numel(meanReds(meanReds == -1));

timeElapsed = toc

% Printing the results
formatStruct.Interpreter = 'tex';
formatStruct.WindowStyle = 'modal';
if degreeDiff > 30
    msgbox(sprintf('The patient has tracheobronchomalacia with a degree of stenosis of
%.2f %%',degreeDiff),'Diagnosis',formatStruct)
else

```



```

        if kmeansAperture >= previousAperture + previousAperture*apertureThreshold ||
...
            kmeansAperture <= previousAperture - previousAperture*apertureThreshold
            kmeansValid = false;
        end
    end
    if previousArea ~= -1
        if kmeansArea >= previousArea + previousArea*0.75 || ...
            kmeansArea <= previousArea - previousArea*0.75
            kmeansValid = false;
        end
    end
end
if binValid
    [binAperture, binCols, binRows, binCentroid, binArea, binBbox, binMidpoint,
binMAL] = getStenosisDegree( ...
        frameCropped, binFrame);

    if previousAperture ~= -1
        if binAperture >= previousAperture + previousAperture*apertureThreshold || ...
            binAperture <= previousAperture - previousAperture*apertureThreshold
            binValid = false;
        end
    end
    if previousArea ~= -1
        if binArea >= previousArea + previousArea*0.75 || ...
            binArea <= previousArea - previousArea*0.75
            binValid = false;
        end
    end
end
if watershedValid
    [watershedAperture, watershedCols, watershedRows, watershedCentroid,
watershedArea, watershedBbox, watershedMidpoint, watershedMAL] = getStenosisDegree( ...
        frameCropped, watershedFrame);

    if previousAperture ~= -1
        if watershedAperture >= previousAperture + previousAperture*apertureThreshold
|| ...
            watershedAperture <= previousAperture - previousAperture*apertureThreshold
            watershedValid = false;
        end
    end
    if previousArea ~= -1
        if watershedArea >= previousArea + previousArea*0.75 || ...
            watershedArea <= previousArea - previousArea*0.75
            watershedValid = false;
        end
    end
end

% Evaluate Results

if kmeansValid

```

```

        if kmeansAperture > 80 ...
            || (kmeansAperture > 50 && (kmeansAperture <= 80 && meanRChannelKmeans <
60)) ...
            || kmeansMAL > 2*kmeansAperture
        if binValid && binAperture > kmeansAperture
            selection = 1;
        elseif ~binValid && watershedValid
            selection = 2;
        end
    end
else
    if binValid
        selection = 1;
    elseif watershedValid
        selection = 2;
    else
        selection = -1;
    end
end

if selection == 0
    sprintf('kmeansSegmentation Used')

    segment = kmeansFrame;
    aperture = kmeansAperture;
    cols = kmeansCols;
    rows = kmeansRows;
    centroid = kmeansCentroid;
    area = kmeansArea;
    bbox = kmeansBbox;
    midpoint = kmeansMidpoint;
elseif selection == 1
    sprintf('ownSegmentation Used')

    segment = binFrame;
    aperture = binAperture;
    cols = binCols;
    rows = binRows;
    centroid = binCentroid;
    area = binArea;
    bbox = binBbox;
    midpoint = binMidpoint;
elseif selection == 2
    sprintf('watershed Segmentation Used')

    segment = watershedFrame;
    aperture = watershedAperture;
    cols = watershedCols;
    rows = watershedRows;
    centroid = watershedCentroid;
    area = watershedArea;
    bbox = watershedBbox;
    midpoint = watershedMidpoint;

```

```

else
    segment = [];
    aperture = -1;
    cols = [];
    rows = [];
    centroid = [];
    area = 0;
    bbox = [];
    midpoint = [];
end
end
end

function mask = kmeansRGBSegmentation(frame, k, flagOuterMarks)

% Separating the 3 different dimensions
R = frame(:,:,1);
G = frame(:,:,2);
B = frame(:,:,3);

% Applying kmeans
[labels, centroids]= imsegkmeans(frame,k,"NumAttempts",3);

% C = labeloverlay(frame,labels);
% figure
% imshow(C)
% title("Labeled Image RGB")

minR = 255; % Red channel minimum intensity
minG = 255; % Green channel minimum intensity
minB = 255; % Blue channel minimum intensity
maxR = 0; % Red channel maximum intensity
maxG = 0; % Green channel maximum intensity
maxB = 0; % Blue channel maximum intensity
ix = 0; % Index of the cluster to return
count = 0;
for i = 1:k
    % Mask applied to input image
    mask = labels == i;
%     cluster = frame.*uint8(mask);
%     figure
%     imshow(cluster)
%     title("RGB Cluster "+i);

    if flagOuterMarks == 0
        % Minimum values of lighth intensity for each channel in the cluster
        auxMinR = min(R(mask > 0));
        auxMinG = min(G(mask > 0));
        auxMinB = min(B(mask > 0));

        % Count the number of channels that has lower values that the current
        % lowest
        if minR > auxMinR
            count = count + 1;

```



```

end
if minG > auxMinG
    count = count + 1;
end
if minB > auxMinB
    count = count + 1;
end

% If 2 or more channels has less lighting I select it as the well
% channel
if count >= 2
    ix = i;
    minR = auxMinR;
    minG = auxMinG;
    minB = auxMinB;
end
else
    % Minimum values of lighth intensity for each channel in the cluster
    auxMaxR = max(R(mask > 0));
    auxMaxG = max(G(mask > 0));
    auxMaxB = max(B(mask > 0));

    % Count the number of channels that has lower values that the current
    % lowest
    if maxR < auxMaxR
        count = count + 1;
    end
    if maxG < auxMaxG
        count = count + 1;
    end
    if maxB < auxMaxB
        count = count + 1;
    end

    % If 2 or more channels has less lighting I select it as the well
    % channel
    if count >= 2
        ix = i;
        maxR = auxMaxR;
        maxG = auxMaxG;
        maxB = auxMaxB;
    end
end

count = 0;
end

% Mask of the resulting cluster
mask = labels == ix;

% B = labels .* uint8(mask);
% C = labeloverlay(frame,B);
% figure

```

```

%   imshow(C)
%   title("Labeled frame with the resulting mask")
end

function mask = kmeansLabSegmentation(frame, originalFrame, k)

    % Separating the 3 different dimensions
    imLuminance = frame(:,:,1);
    ab = frame(:,:,2:3);
    ab = im2single(ab);
    a = ab(:,:,1);
    b = ab(:,:,2);

    % Applying kmeans only to the color dimensions
    labels = imsegkmeans(ab,k,"NumAttempts",3);

    B = labeloverlay(originalFrame,labels);
%   figure
%   imshow(B)
%   title("Labeled Image a*b*")

    wellIx = 0; % Index of the cluster to return
    minimalLuminance = 100; % Luminance in L*a*b* goes from [0 to 100]
    for i = 1:k
        % Mask applied to input image
        mask = labels == i;
        cluster = originalFrame.*uint8(mask);
%       figure
%       imshow(cluster)
%       title("Cluster "+i);

        % Getting the luminance of the cluster
        luminance = imLuminance.*mask;

        % Minimum value of luminance per cluster
        maskMinLuminance = min(luminance(luminance > 0));

        % Storing the index of the cluster with less luminance
        if maskMinLuminance < minimalLuminance
            minimalLuminance = maskMinLuminance;
            wellIx = i;
        end
    end

    % Mask of the resulting cluster
    mask = labels == wellIx;

%   B = labels .* uint8(mask);
%   C = labeloverlay(originalFrame,B);
%   figure
%   imshow(C)
%   title("Labeled frame with the resulting mask")
end

```

```

function watershedSegmentation = watershedSegmentationKmeans(frame, preproc)

    outerMostRegion = kmeansRGBSegmentation(frame, 2, 1);

    innerMostRegion = kmeansRGBSegmentation(frame, 14, 0);

    imGray = rgb2gray(frame);
    % If the frame hasn't been preprocessed, we preprocessed it in order to get
    % good regional minimas and maximas and to eliminate noise from the
    % gradient
    if preproc == 'n'
        grayProcessed = openCloseByReconstruction(imGray);
    else
        grayProcessed = imGray;
    end

    % Regional minimas and maximas
    minimas = imregionalmin(grayProcessed);
    maximas = imregionalmax(grayProcessed);

    minimas = minimas & ~innerMostRegion;
    outerMostRegion = outerMostRegion | minimas | maximas;

    % Eliminating small connected components and noise from the markers
    se = strel(ones(5,5));
    innerMostRegion = imclose(innerMostRegion,se);
    innerMostRegion = imerode(innerMostRegion,se);
    innerMostRegion = bwareaopen(innerMostRegion,50);

    outerMostRegion = imclose(outerMostRegion,se);
    outerMostRegion = imerode(outerMostRegion,se);
    outerMostRegion = bwareaopen(outerMostRegion,50);

    % figure
    % imshow(outerMostRegion)
    % title('Outer-Most Regions')

    % figure
    % imshow(innerMostRegion)
    % title('Inner-Most Regionals')

    I2 = labeloverlay(frame,innerMostRegion | outerMostRegion);
    % figure
    % imshow(I2)
    % title('Outer-most markers and inner-most markers imposed on frame')

    % Gradient of the image
    gmag = imgradient(grayProcessed);
    % figure
    % imshow(gmag,[]);

```

```

% title('Image gradient')

fgm = (innerMostRegion | outerMostRegion);
gmag2 = imimposemin(gmag, fgm);
% figure
% imshow(gmag2,[])
% title('Gradient with markers imposed')

L = watershed(gmag2);
rL = L == 0;
% figure
% imshow(rL)
% title('Watershed Ridge Lines')

labels = imdilate(L==0,ones(3,3)) + 2*(innerMostRegion | outerMostRegion) + 3*fgm;
I4 = labeloverlay(frame,labels);
% figure
% imshow(I4)
% title('Markers and Object Boundaries Superimposed on Frame')

% labeling the watershed results
Lrgb = label2rgb(L, 'jet', 'w', 'shuffle');
% figure
% imshow(Lrgb)
% title('Colored Watershed Label Matrix')

% Getting the the value of the labels for the biggest minimum blob in the frame
labelValue = L(innerMostRegion);
% Getting the labels of the inner-most regions
values = unique(labelValue(:,1));
% Getting the watershed segmentation of the stenosis level
watershedSegmentation = zeros(size(L,1), size(L,2));
for i = 1:size(values)
    watershedSegmentation = watershedSegmentation | L == values(i);
end
I5 = labeloverlay(frame,watershedSegmentation);
% figure
% imshow(I5)
% title('Watershed final mask')
end

function watershedSegmentation = watershedSegmentation(preProcessedFrame, frameCropped,
preproc)
% If the frame hasn't been preprocessed, we preprocessed it in order to get
% good regional minimas and maximas
if preproc == 'n'
    preProcessedFrame = openCloseByReconstruction(preProcessedFrame);
end

maximums = imregionalmax(preProcessedFrame);
% figure

```

```

% imshow(maximums)
% title('Regional maximums')

minimums = imregionalmin(preProcessedFrame);
% figure
% imshow(minimums)
% title('Regional Minimums')

minimum = extractClosestBlob(minimums, [size(frameCropped,1) / 2 size(frameCropped,2)
/ 2]);
% figure
% imshow(minimum)
% title('Biggest minimum')

D = bwdist(maximums | minimums);
DL = watershed(D);
bgm = DL == 0;
% figure
% imshow(bgm)
% title('Watershed Ridge Lines')

I2 = labeloverlay(frameCropped,minimums | maximums);
% figure
% imshow(I2)
% title('Regional Minimum and Maxima Superimposed on Frame')

se2 = strel(ones(5,5));
fgm2 = imclose(minimums | maximums,se2);
fgm3 = imerode(fgm2,se2);
fgm4 = bwareaopen(fgm3,20);
I3 = labeloverlay(frameCropped,fgm4);
% figure
% imshow(I3)
% title('Modified Regional Maxima Superimposed on Frame')

% Gradient of the image
gmag = imgradient(preProcessedFrame);
% figure
% imshow(gmag,[]);
% title('Image gradient')

gmag2 = imimposemin(gmag, fgm4);
% figure
% imshow(gmag2,[])
% title('Gradient with markers imposed')
L = watershed(gmag2);
labels = imdilate(L==0,ones(3,3)) + 2*(minimums | maximums) + 3*fgm4;
I4 = labeloverlay(frameCropped,labels);
% figure
% imshow(I4)
% title('Markers and Object Boundaries Superimposed on rame')
% labeling the watershed results
Lrgb = label2rgb(L, 'jet', 'w', 'shuffle');
% figure

```

```

%   imshow(Lrgb)
%   title('Colored Watershed Label Matrix')
% Getting the the value of the labels for the biggest minimum blob in the frame
labelValue = L(minimum);
% Getting the label with maximum occurrence
value = mode(labelValue);
% Getting the watershed segmentation of the stenosis level
watershedSegmentation= L == value;
%   figure
%   imshow(watershedSegmentation);
%   title('Watershed final mask')
end

function [aperture, cols, rows, segmentCentroid, area, bbox, midpoint, minorAxisLen] =
getStenosisDegree(frameCropped, frameMask)

% ----- Getting the degree of aperture and saving the image -----
% Max diff result
edgeDetected = edge(frameMask, "canny");
%   frameOutlined = imoverlay(frameCropped,edgeDetected, 'yellow');

%   firstFig = figure('visible','off');
%   firstFig = figure;
%   imshow(frameMask);

stats = regionprops(frameMask, 'Centroid', 'BoundingBox', ...
    'Orientation', 'MinFerretProperties', 'Area', 'MinorAxisLength');

area = stats.Area;
bbox = stats.BoundingBox;
orientation = stats.Orientation; % Getting the perpendicular orientation
minorAxisLen = stats.MinorAxisLength;

if orientation < 45 && orientation > -45
    xCentroid = bbox(1) + bbox(3)/2;
    yCentroid = stats.Centroid(2);
else
    xCentroid = stats.Centroid(1);
    yCentroid = bbox(2) + bbox(4)/2;
end
%   previousCentroid = [xCentroid yCentroid];

% Calculating the aperture using the orientation of the blob
% and the centroid
sinOrient = sind(orientation);
cosOrient = cosd(orientation);

if orientation < 45 && orientation > -45
    vertUpperDist = pdist([xCentroid, yCentroid; xCentroid, bbox(2),
], 'euclidean');
    vertLowerDist = pdist([xCentroid, yCentroid; xCentroid, (bbox(2) +
bbox(4))], 'euclidean');

```

```

else
    vertUpperDist = pdist([xCentroid, yCentroid; bbox(1), yCentroid,
], 'euclidean');
    vertLowerDist = pdist([xCentroid, yCentroid; (bbox(1) + bbox(3))
,yCentroid], 'euclidean');
end
% Calculating the endpoints of the line of aperture from the centroid,
% adding the relative distance to the bounding box plus a half
xCoords(1) = xCentroid + (vertUpperDist + vertUpperDist/2) * sinOrient;
yCoords(1) = yCentroid + (vertUpperDist + vertUpperDist/2) * cosOrient;
xCoords(2) = xCentroid - (vertLowerDist + vertLowerDist/2) * sinOrient;
yCoords(2) = yCentroid - (vertLowerDist + vertLowerDist/2) * cosOrient;

% Distance between the two endpoints of the line
nPoints = ceil(sqrt((xCoords(2) - xCoords(1)).^2 + (yCoords(2) - yCoords(1)).^2))
+ 1;

% X and Y values of the endpoints
xvalues = round(linspace(xCoords(1), xCoords(2), nPoints));
yvalues = round(linspace(yCoords(1), yCoords(2), nPoints));

% Cropping the aperture pixels that surpass the bounding box of the blob
if orientation < 135 && orientation > -45
    xvalues(yvalues < bbox(2)) = [];
    yvalues(yvalues < bbox(2)) = [];
    xvalues(yvalues > (bbox(2) + bbox(4))) = [];
    yvalues(yvalues > (bbox(2) + bbox(4))) = [];

    yvalues(xvalues < bbox(1)) = [];
    xvalues(xvalues < bbox(1)) = [];
    yvalues(xvalues > (bbox(1) + bbox(3))) = [];
    xvalues(xvalues > (bbox(1) + bbox(3))) = [];
else
    xvalues(yvalues < bbox(2)) = [];
    yvalues(yvalues < bbox(2)) = [];
    xvalues(yvalues > (bbox(2) + bbox(4))) = [];
    yvalues(yvalues > (bbox(2) + bbox(4))) = [];

    yvalues(xvalues < bbox(1)) = [];
    xvalues(xvalues < bbox(1)) = [];
    yvalues(xvalues > (bbox(1) + bbox(3))) = [];
    xvalues(xvalues > (bbox(1) + bbox(3))) = [];
end

% Mask the values of intercection of the line with the edge of the segmentation
mask
lineMask = false(size(edgeDetected,1),size(edgeDetected,2));
lineMask(sub2ind(size(lineMask), yvalues, xvalues)) = 1;

% Dilating the line to ensure that it intersects with the edge
lineMask = imdilate(lineMask, strel('line',5,orientation));
lineMask = edgeDetected .* lineMask;

% Values of the two farthest points of the line;

```

```

[row1,col1] = find(lineMask, 1, 'first');
[row2,col2] = find(lineMask, 1, 'last');
cols = [col1 col2];
rows = [row1 row2];
midpoint = ([row1 col1] + [row2 col2]).'/2;
aperture = pdist([row1, col1; row2, col2], 'euclidean');
segmentCentroid = [xCentroid yCentroid];

% Plot of all the data on top of the original image
% hold on;
% plot([col1,col2],[row1,row2],'c','LineWidth',2)
% plot(xCentroid, yCentroid, 'r+', 'LineWidth', 2, 'MarkerSize', 7)
% rectangle('Position',bbox,"EdgeColor",'y')
% title('Segmentation outline with the degree of stenosis');
% text(midpoint(2) + 20,midpoint(1),string(aperture),'Color',[0 0 1],'FontSize',8)
% saveas(firstFig,[pwd path S(frameIx).name]);
end

function mask = binBinarization(frame,bins,preproc)
    global previousThreshold
    [counts,binLocations] = imhist(frame,bins);
    % Getting first local maxima in the hist with a threshold of 3500
    % I use this threshold to ignore small local peaks
    [pks,locs] = findpeaks(counts,'MinPeakHeight',3500);
    if not isempty(pks)
        ix = locs(1) - 1; %indexing to the previous bin for the calculus
        % Setting the threshold using the % of divergence between bins
        countDiff = (counts(ix) * 100) / counts(ix + 1);
        binDiff = binLocations(ix + 1) - binLocations(ix);
        binThresh = binLocations(ix) + binDiff * ((100-countDiff)/100);
    else
        diffs = diff(counts);
        [~, diffIx] = max(diffs(diffs>=0));
        countDiff = (counts(diffIx) * 100) / counts(diffIx + 1);
        binDiff = binLocations(diffIx + 1) - binLocations(diffIx);
        binThresh = binLocations(diffIx) + binDiff * ((100-countDiff)/100);
    end

    % A 15% is substracted for fine tuning of the binarization, due to
    % overestimation of the threshold in case of not using preprocessing
    if preproc == 'n'
        binThresh = binThresh - binThresh*0.125;
    end

    % The previous threshold is taken into account in order to get more
    % accurate results
    if previousThreshold ~= -1
        threshDiff = binThresh - previousThreshold;
        if threshDiff >= 0
            percentageOfDiff = previousThreshold*100 / binThresh;
            binThresh = binThresh - threshDiff*(percentageOfDiff/100);
        else

```



```

        percentageOfDiff = binThresh*100 / previousThreshold;
        binThresh = binThresh + abs(threshDiff)*(percentageOfDiff/100);
    end
end
previousThreshold = binThresh;
mask = frame <= binThresh;
end

function openClosedImage = openCloseByReconstruction(frame)

    se = strel('disk',20);
    % Erosion
    frameErode = imerode(frame,se);
%   figure
%   imshow(frameErode)
%   title('Erosion')
    % Opening using reconstruction of the erode frame
    frameObR = imreconstruct(frameErode,frame);
%   figure
%   imshow(frameObR)
%   title('Opened by reconstruction')
    % Dilation of the opened frame
    frameObRDilated = imdilate(frameObR,se);
%   figure
%   imshow(frameObRDilated)
%   title('Opened by eeconstruction and dilated')
    % Open and Close by reconstruction
    frameOCRecon = imreconstruct(imcomplement(frameObRDilated),imcomplement(frameObR));
    openClosedImage = imcomplement(frameOCRecon);
%   figure
%   imshow(openClosedImage)
%   title('Opening-Closing by Reconstruction')
end

function binaryImage = extractClosestBlob(binaryImage, point)
    try
        % Get all the Centroids
        [labeledImage, numberOfBlobs] = bwlabel(binaryImage);
        blobStats = regionprops(labeledImage, 'Centroid');
        allCentroids = [blobStats.Centroid];
        allCentroids = reshape(allCentroids,2,[]).';

        % Get the closest Blob to the point
        [k, dist] = dsearchn(point,allCentroids);
        [~, ixStats] = min(dist);

        % Extract the blob using ismember().
        biggestBlob = ismember(labeledImage, ixStats);
        % Convert from integer labeled image into binary (logical) image.
        binaryImage = biggestBlob > 0;
    end
end

```

```

catch ME
    errorMessage = sprintf('Error in function extractClosestBlob().\n\nError
Message:\n%s', ME.message);
    fprintf(1, '%s\n', errorMessage);
    uiwait(warndlg(errorMessage));
end
end

function segmentFrame = getSegment(binFrame)
    global previousCentroid

    % ----- Filling holes -----
    filledFrame = imfill(binFrame,8,'holes');
    % -----

    % ----- Closing gaps -----
    % With lines
    positiveSlope = strel('line',15,45);
    negativeSlope = strel('line',15,-45);
    horizontalLine = strel('line',15,0);
    verticalLine = strel('line',15,90);
    closedFrame=imclose(filledFrame,positiveSlope);
    closedFrame=imclose(closedFrame,negativeSlope);
    closedFrame=imclose(closedFrame,horizontalLine);
    closedFrame=imclose(closedFrame,verticalLine);
    closedFilledFrame = imfill(closedFrame,8,'holes');
    % -----

    % -- Removing small Conn. Comp. and getting the closest blob --
    % ----- to the previous centroid -----
    props = regionprops(closedFilledFrame, 'Area');
    allAreas = [props.Area];
    cleanedFrame = bwareaopen(closedFilledFrame, ceil(max(allAreas)*0.3));

    props = regionprops(cleanedFrame, 'Centroid', 'BoundingBox');
    CC = bwconncomp(cleanedFrame);
    L = labelmatrix(CC);
    % If there are more than one blob, I get the closest to the centroid of
    % the last frame with a valid centroid
    ixProps = 1; % Index of the stats struct
    if numel(props) > 1
        centroids = [props.Centroid];
        centroids = reshape(centroids,2,[]).';
        [~, dist] = dsearchn(previousCentroid,centroids);
        [~, ixProps] = min(dist);
    end

    cleanedFrame = L == ixProps;
    % -----

%    imshow(cleanedFrame);
%    title('Removal of unnecessary connected components and filling gaps');
%    figure

```

```
% Closing any left gaps
closeBlobFrame=imclose(cleanedFrame,positiveSlope);
closeBlobFrame=imclose(closeBlobFrame,negativeSlope);
closeBlobFrame=imclose(closeBlobFrame,horizontalline);
closeBlobFrame=imclose(closeBlobFrame,verticalline);
closeBlobFrame=imfill(closeBlobFrame,8,'holes');

%   imshow(closeBlobFrame);
%   title('Final segmentation mask');
%   figure

segmentFrame = closeBlobFrame;
% -----
end
```

Script 2. Script used to give a diagnosis based on a selection of frames.

GET VIDEO SCRIPT

```

clear
clc

prompt = 'Insert the path of the folder containing the frames: ';
inputStr = input(prompt,'s');

% I get the name of the video folder to use it as the name of the video
tokens = regexp(inputStr,'(\w+)$','tokens');
workingDir = tokens{1}{1};

videoName = [workingDir '.avi'];

% Video Reader to read
outputVideo = VideoWriter(fullfile(workingDir,videoName));
outputVideo.FrameRate = 15;

% Getting the names of the frames
imageNames = dir(fullfile(workingDir,'*.jpg'));
imageNames = {imageNames.name}';
% I sort the images taking into account the values of the numbers on the
% strings
sortedImages = natsortfiles(imageNames);

% Creating the video
open(outputVideo)
for ii = 1:length(sortedImages)
    img = imread(fullfile(workingDir,sortedImages{ii}));
    writeVideo(outputVideo,img)
end
close(outputVideo)

```

Script 3. Script used to obtain a video from the resulting frames of mainProgram.mlx script