



HERIOT-WATT UNIVERSITY

VIBOT MASTER

---

## Tracking of Minimally Invasive Surgery Tools for Skill Assessment

---

*Authors:*

Eric PAIRET  
Jose BERNAL  
Rodrigo DAUDT

*Supervisor:*

Dr. Mustapha SUPHI  
*Examiner:*  
Prof. Yvan PETILLOT

*A project submitted in fulfilment of the requirements  
of the robotics project subject.*

December 2016

HERIOT-WATT UNIVERSITY

## *Abstract*

### **Tracking of Minimally Invasive Surgery Tools for Skill Assessment**

by Èric PAIRET

Jose BERNAL

Rodrigo DAUDT

Minimally Invasive Surgery is a specific surgical technique in which tools are inserted into the patient through small incisions on the body. Although the use of this kind of techniques has been increasing throughout the years since the traumas to the patient are reduced, the procedure is still challenging for surgeons due to the complexity that it supposes. The most critical drawback is that depth information is lost since a single camera is used during the operation. To overcome this situation, surgeons must train for several and expensive sessions to use the video feed to correctly operate the tools. Thus, identifying the flaws of each subject while performing training tasks is of interest. This process has been widely addressed in the literature by combining tracking and computer vision techniques for estimating the path executed by the trainees, but, to the best of the authors' knowledge, the obtained track has not been used for determining the level of skill of a subject. We propose a framework for tracking MIS tools on laparoscopy training environments and assessing the subjects based on the described paths. Due to the lack of data of skilled and unskilled persons, the assessment part of the project was only tackled theoretically. Results of the framework indicate that the proposal is able to produce results that look like the actual trajectory and that the maximum error when performing the same experiment several times is of the order of 12mm. Also, the features extracted from the synthetic data suggest that the considered measures may be useful for determining how skilled is a subject. However, further experiments on real data should be carried out to validate this assertion.

# Contents

|   |           |
|---|-----------|
| <b>Abbreviations</b>  | <b>iv</b> |
| <b>1 Introduction</b>                                       | <b>1</b>  |
| <b>2 Problem analysis</b>                                   | <b>5</b>  |
| <b>3 Project management</b>                                 | <b>8</b>  |
| <b>4 Considered methods</b>                                 | <b>10</b> |
| 4.1 Camera calibration . . . . .                            | 10        |
| 4.2 Tool detection . . . . .                                | 12        |
| 4.2.1 Tool segmentation . . . . .                           | 12        |
| 4.2.1.1 Motion segmentation . . . . .                       | 13        |
| 4.2.1.2 Colour segmentation in the distance space . . . . . | 15        |
| 4.2.1.3 Colour segmentation in the RGB space . . . . .      | 17        |
| 4.2.2 Tip localisation . . . . .                            | 19        |
| 4.3 Tool pose estimation . . . . .                          | 22        |
| 4.3.1 3D pose estimation . . . . .                          | 22        |
| 4.3.1.1 Pinhole camera model . . . . .                      | 22        |
| 4.3.1.2 Edge-crossing planes . . . . .                      | 24        |
| 4.3.2 Data filtering . . . . .                              | 26        |
| 4.4 Skill assessment . . . . .                              | 28        |
| <b>5 Framework design</b>                                   | <b>31</b> |
| 5.1 Framework 1 . . . . .                                   | 31        |
| 5.2 Framework 2 . . . . .                                   | 34        |
| <b>6 Implementation</b>                                     | <b>38</b> |
| 6.1 Camera calibration toolbox . . . . .                    | 38        |
| 6.2 Deciding the colour of the markers . . . . .            | 39        |
| 6.3 Pixel classification . . . . .                          | 39        |
| 6.4 Tip detection . . . . .                                 | 41        |
| 6.5 Kalman filter parametrisation . . . . .                 | 42        |
| 6.6 Speeding up the framework . . . . .                     | 42        |
| 6.6.1 Multi-scale segmentation . . . . .                    | 43        |
| 6.6.2 RGB classification pre-computation . . . . .          | 43        |

|   |           |
|---|-----------|
| <b>7 Results and evaluation</b>                 | <b>44</b> |
| 7.1 Implemented part of the framework . . . . . | 44        |
| 7.1.1 Single component evaluation . . . . .     | 44        |
| 7.1.2 General evaluation . . . . .              | 47        |
| 7.1.2.1 Trial 1 . . . . .                       | 47        |
| 7.1.2.2 Trial 2 . . . . .                       | 49        |
| 7.1.3 Quantitative evaluation . . . . .         | 52        |
| 7.1.4 Performance analysis . . . . .            | 54        |
| 7.2 Theoretical part of the framework . . . . . | 55        |
| <b>8 Final remarks</b>                          | <b>58</b> |
| <b>Bibliography</b>                             | <b>60</b> |

# Abbreviations

|               |                              |
|---------------|------------------------------|
| <b>CAD</b>    | Computer-Aided Design        |
| <b>DOF</b>    | Degree Of Freedom            |
| <b>EKF</b>    | Extended Kalman Filter       |
| <b>FOV</b>    | Field Of View                |
| <b>FPS</b>    | Frames Per Second            |
| <b>HSV</b>    | Hue Saturation Value         |
| <b>KF</b>     | Kalman Filter                |
| <b>KNN</b>    | K-Nearest Neighbors          |
| <b>LED</b>    | Light Emitting Diode         |
| <b>LMS</b>    | Least Mean Squares           |
| <b>MIS</b>    | Minimally Invasive Surgery   |
| <b>RAM</b>    | Random-Access Memory         |
| <b>RANSAC</b> | RANdom SAmple Consensus      |
| <b>RGB</b>    | Red Green Blue               |
| <b>ROI</b>    | Region Of Interest           |
| <b>SVM</b>    | Singular Value Decomposition |
| <b>UKF</b>    | Unscented Kalman Filter      |

# Chapter 1

## Introduction

Minimally Invasive Surgery (MIS) is a subgroup of surgical techniques that aim to reduce the damage done to the human body during surgery. Laparoscopy, a type of MIS, consists on inserting a laparoscope and thin, long tools into the abdominal or pelvic region through small incisions on the skin, while pumping an inert gas into the patient's body to allow for the movement of the tools. This procedure exchanges a large incision by a few small ones, reducing the risk of infection and time of recovery.

While laparoscopic techniques reduce the damage done to the patient's body, the techniques used in laparoscopic surgery are complicated and require lengthy training since the laparoscopic set-up provides a limited Field Of View (FOV), reduces hand-eye coordination and leads to loss of depth information. The surgeons must be trained to use the video feed from a single camera to be able to operate the tools accurately.

Tool tracking systems for laparoscopic setups have been proposed. Some use specialised hardware to perform electromagnetic, mechanical, or sonic localisation of the laparoscopic tools [Tonet et al. \[2007\]](#). These systems increase the cost and the size of the equipment used for laparoscopic surgery and therefore are not ideal. Another option is to use computer vision techniques on the video feed which is already available to estimate the position of the laparoscopic tools relative to the laparoscope.

Visual tracking of laparoscopic tools is a problem currently being tackled by researchers. Tracking the tools is useful for many different purposes. First, it can be used in computer-assisted surgery systems as a base for augmented reality systems that can help to guide the surgeon

during the medical procedure. A sufficiently accurate system may one day be used on an automatic robot surgeon, which could be able to assist a trained surgeon during the laparoscopic surgery. Finally, tool tracking can be used during the training of new surgeons. The tracking can assist the trainees by identifying their mistakes, by assessing their general skill level based on recorded data and to evaluate the trainees' progress.

Tracking methods for laparoscopic tools have already been proposed. These methods estimate the XYZ coordinates of the laparoscopic tools relative to the camera coordinate system. Some of them estimate also the rotation and grasper angles of the tools or the inclination of the tools for a more informative tracking. Methods with and without markers have been proposed with different rates of success.

[Hulke and Gupta \[2014\]](#) proposed a method with simple markers added to the tools which were tracked using the Kalman Filter (KF). The proposed markers were spheres with a saturated colour, that should be attached to the tools, facilitating the segmentation. Since these markers are actually coming out of the tool, the strongest flaw of the approach is that it changes the shape of the laparoscopic tool to something that is much harder to be inserted in the human body during a real surgery. Also, the markers are not visible for some rotation angles and, thus, the estimation of XYZ is limited. This technique has been tested with some degree of success on a simplified set-up where the laparoscopic tool was substituted by a simple stick with the addition of the markers.

[Shin et al. \[2014\]](#) introduced a method with an elaborated 5-part marker on each tool which allows for the measurement of the XYZ coordinates of the tool tip, the rotation angle of the tool and the opening angle of the grasper. The marker consisted of three main rings for XYZ estimation, one extra ring for estimating the grasper angle, and one helicoidal mark which enables to calculate the rotation angle. This method takes advantage of the colour of the markers in the HSV colour space to segment the tool from the background. Additionally, the authors considered a KF approximation in case the detection was not successful due to, for instance, crossing markers. This technique obtained good results, achieving errors of approximately 5mm in their tests. However, this accuracy comes at the cost of the addition of an elaborate marker to each tool which has to be completely visible all the time.

[Tonet et al. \[2007\]](#) proposed a tracking method that used a single cylindrical marker on each tool to facilitate the segmentation. In this case, the authors considered a marker which was distinguishable from the background in the HSV colour space. By using the pinhole camera

model and proportions between the width measured in the image plane with respect to the prior knowledge of the tool in the real world, the position and inclination angles of the tools relative to the camera coordinate system were estimated. As stated by the authors, this approximation is only suitable for applications in which accuracy is not critical since small miscalculations in the image plane may lead to high inaccuracy in the estimated pose.

[Zhou and Payandeh \[2014\]](#) proposed a tracking method that used no markers on the laparoscopic tools. Unlike the previous approaches, this approach was tested on real surgical video sequences and, thus, although the scenario gets more complicated in terms of tracking the movement of the tools and illumination conditions (e.g. some tissues may reflect the light coming from the lighting source), the scenario brings two advantages: (i) the camera is placed directly on top of the laparoscopy tools and, hence, assumptions for calculating the depth of the tool discussed later in this paper are considered, (ii) although there might be some shadows due to poor lighting conditions on the boundaries of the image, the black tube can be easily recognised since there is no tissue which appears in black and, consequently, accurate segmentation on the red channel is easily achieved. In this case, the method contemplates tracking the middle line of the segmented tools using the Extended KF (EKF) to cope with non-linearities. The method was able to achieve acceptable performance in high-quality videos in which high contrast between the tool and the background is observed, but it failed when the conditions were less than optimal, i.e. a very controlled environment is required for this method to work which is not usually the case.

Summarising, the literature suggests that:

- Marker-based approaches should be considered when high accuracy is required in uncontrolled environments. However, this feature is not completely necessary in training environments and, hence, we do not discard completely the idea of using a markerless approximation.
- Regardless the configuration, colour is a key feature for differentiating between the laparoscopic tools and the background. Note that colour could be ideally complemented with other types of features to enhance its performance.
- Pose estimation with a single camera is an ill-posed problem and, hence, prior information should be added in order to obtain stable and realistic results. The problem is often

addressed by computing proportions between the observed width of the tool in the image plane and the information of the tool in the real world.

- Tracking algorithms such as KF or EKF are commonly used for dealing with wrong or missing observations.

Based on the works in the state-of-the-art, in this paper, we present a framework for tracking minimally invasive surgery tools for assessing trainees while they perform some tasks. The framework is two-fold: tool tracking and skill assessment. The process starts when a video sequence is gathered from the training system and processed to estimated the 3D position with respect to the camera. Once the full trajectory executed by the operator is obtained, the next step consists in evaluating how skilled is the trainee based on metrics over the computed trajectory.

The report is structured as follows. Having in mind the brief discussion of methods adopted in the state-of-the-art of tracking tools for minimally invasive surgery and the characteristics of our specific scenario (i.e. training centre), the possible advantages and limitations as well as the initial workflow of the system are outlined in Chapter 2. The project was then divided into smaller tasks and, thus, in Chapter 3, we describe how this project was managed from the planning to the actual implementation and evaluation. Chapter 4 contains information relating all the techniques that we considered for elaborating this project and, in Chapter 5, we explain and criticise the different frameworks that we were able to develop. Chapter 6 is dedicated to describe details of the actual implementation of the final framework. Then, the obtained results with the proposed approach are presented and discussed Chapter 7. Finally, some final remarks and future work are stated in Chapter 8.

# Chapter 2

## Problem analysis

After reviewing the state-of-the-art on tracking of laparoscopic tools, an exhaustive evaluation of our problem was performed. The aim of such analysis was to come up with the most suitable approach to successfully track the laparoscopic tools, extract the executed path and finally assess the subjects.

The overall process starts with a subject performing specific training activities in the laparoscopy training centre displayed in Fig. 2.1. As the operator performs the assigned task, a video sequence is acquired.



Figure 2.1: Laparoscopic training centre.

Taking into account the high distortion in the raw images gathered from the laparoscopic training centre, which can be seen in Fig. 2.2, a camera calibration procedure should be considered at the very beginning of the proposed framework.



Figure 2.2: Raw frame acquired using the camera in the training centre.

Once the radial distortion of the lens has been corrected, the tracking of the tools can be performed in a more realistic representation of the scene. In this context, the concept of tracking includes two keystones of the project, which are detection of the tool and estimation of its position in the space. Out from these estimations, the 3D trajectories followed by the tools can be reconstructed.

The interest of obtaining the paths of the tools is that they can be used to assess the skills of the subjects training with the laparoscopic centre. The authors of this work think that training a classifier using features from tool paths from skilled and unskilled surgeons might be a good approach to assess the experience of the subject being evaluated.

To sum up, the described workflow is illustrated in Fig. 2.3, with which the authors expect to successfully address this challenging project.

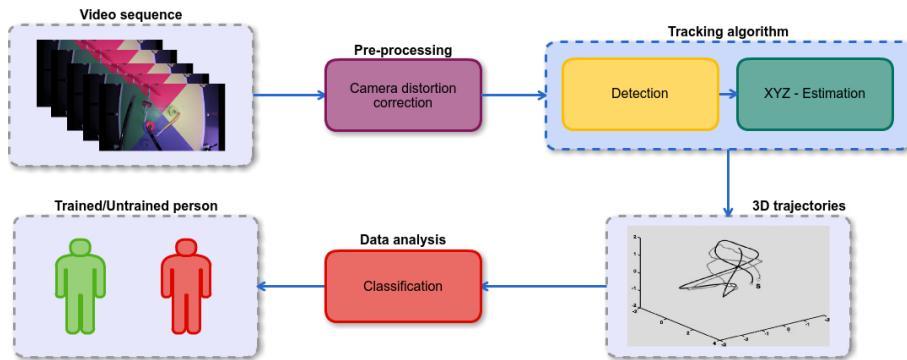


Figure 2.3: Proposed workflow.

From this initial analysis of the problem and considering all the work seen in the state-of-the-art, the following points have to be highlighted:

- Taking into account the type of distortion on the gathered frames, the camera calibration process has to be able to model the radial distortion of the lens.

- Since the tools are the main object moving around the scene, a motion-based detection approach is explored. Also, the dark colour of the tools can be leveraged to distinguish them from the background. However, the presence of glare and shadows might result in a handicap.
- Mathematically, estimating the position of an object with a single-camera based system with no additional information is not possible. Thus, considering the homogeneous cylindrical shape of the tool is necessary to relax the constraints of a single-camera based system.
- Even though a proper review of the state-of-the-art on feature extraction of trajectories has not been done, its length, duration and smoothness seem to be important characteristics to determine the skills of the subject on evaluation.

## Chapter 3

# Project management

When working in a team, the project management takes special importance. A bad organisation may lead to unnecessary waste of time, efforts and, consequently, penalise the final work. This is why we have used two platforms to organise the tasks and synchronise the work.

A platform called Bitbucket has been used to share the code and data, keep track of all the changes, notify the issues found during the implementation and assign tasks to the members of the group. Another one named Overleaf has been used to write the three presentations and the final report required throughout the Robotics Project subject in an efficient and collaborative way. This platform allows all the members of the group to edit L<sup>A</sup>T<sub>E</sub>X documents at the same time.

After the state-of-the-art review done in Chapter 1, the problem analysis introduced in Chapter 2 and considering the three presentations that had to be carried out throughout the course, the different tasks of the project were scheduled as shown in Fig. 3.1. This Gantt chart indicates in blue the planned duration of each task, while the real one is indicated in red.

In the Gantt chart above, the first week of the project corresponds to the third week of the semester. Then, it can be seen that we started working on the project as soon as it was assigned to us. All the intermediate deadlines, i.e. *Checkpoint 1*, *Checkpoint 2*, *Reading group* and *Final presentation*, have been accomplished in time. Some of the other planned tasks have been slightly delayed because they supposed a bigger amount of work than what it was initially expected.

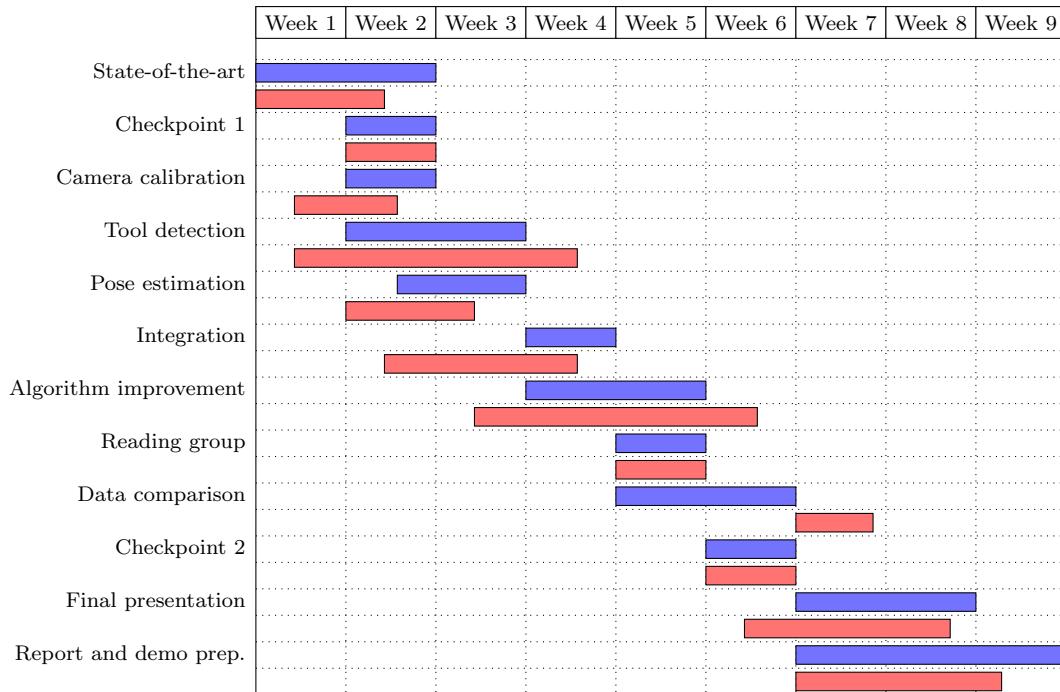


Figure 3.1: Gantt chart. Expected and final duration of each task respectively indicated in blue and red.

Regarding the *Data comparison* task, it has been finally reduced to a theoretical proposal; the necessary data for this step was provided to us by the sixth week when the project was already oriented to be compatible with scenarios similar to the one of the laparoscopic training centre. However, (i) the scenario in the given videos had a lot of black areas, (ii) the tool was moving extremely fast, and (iii) the tool was outside the field of view of the camera a considerable portion of the time. Despite this fact and the previously mentioned delays, the overall proposed approach has been successfully finished in time.

Even though some of the tasks have been done individually by one member of the group, the peer-review process forced the others to understand, check, criticise and correct the others' code. Only in that way, an equivalent amount of work can be attributed to all the members of the group. Additionally, the preparation of the presentations and the creation of the final report has been carried out simultaneously by all the members of the group.

# Chapter 4

## Considered methods

During the development of the framework for tracking the laparoscopic tools and assessing the skills of the subjects on evaluation, different methods have been tried. Thus, the aim of this section is to describe them before going into details of the final framework and its implementation.

Accordingly to Chapter 2, the workflow of the proposed approach is composed by four main parts: camera calibration, tool detection, pose estimation and skill assessment, which will be respectively addressed in Section 4.1, Section 4.2, Section 4.3 and Section 4.4, in that order.

### 4.1 Camera calibration

In Chapter 2, it has been stated that correcting the distortion of the gathered sequence of frames is indispensable for working with a more realistic representation of the scene. This implies knowing the model that allows us to place each pixel  $\{\tilde{u}_j^R \tilde{v}_j^R\}$  from the gathered image  $\tilde{J}$  (Fig. 4.1a) to its correct position  $\{u_j^R v_j^R\}$ , thus composing an undistorted image  $J$  (Fig. 4.1b). Note that such transformation is referenced to a specific reference frame  $\{R\}$ , which can be either the image frame  $\{I\}$  or the camera frame  $\{C\}$ . An accurate representation of the model of the camera takes into account both the camera intrinsic parameters and the lens distortion.

The camera intrinsic matrix is the basis of the well-known pinhole camera model [Zhang \[2000\]](#) introduced in Eq. 4.1. With such model, a point  $\{x_j^C y_j^C z_j^C\}$  represented with respect to the camera frame  $\{C\}$  can be projected onto the image plane  $I$ . Note that this model is linear and

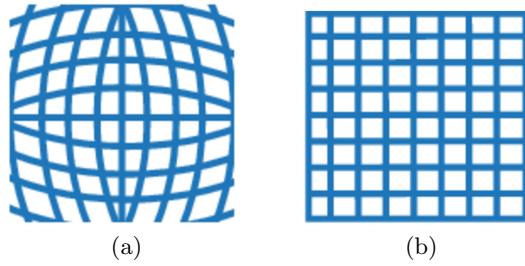


Figure 4.1: Aim of camera calibration [MathWorks \[2016b\]](#). (a) Gathered image  $\tilde{J}$ , (b) undistorted image  $J$ .

considers the absence of lens on the camera.

$$\begin{bmatrix} u_j^I \\ v_j^I \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & s & u_I^C \\ 0 & f_y & v_I^C \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_j^C \\ y_j^C \\ 1 \end{bmatrix}, \quad (4.1)$$

where  $f_x$  and  $f_y$  are the focal length in the x-axis and y-axis, respectively,  $s$  determines the skew of the pixels and  $\{u_I^C\ v_I^C\}$  describes the central point of the camera, i.e. the translation from the image frame  $\{I\}$  to the camera frame  $\{C\}$  in the  $x$  and  $y$  axis, respectively.

The effect of the lens on the projection is usually modelled by considering the radial and tangential distortion that they induce to the system [Heikkila and Silven \[1997\]](#). The former distortion is modelled according to

$$\tilde{u}_j^C = u_j^C \left( 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \right), \quad (4.2)$$

$$\tilde{v}_j^C = v_j^C \left( 1 + k_1 r^2 + k_2 r^4 + k_3 r^6 \right), \quad (4.3)$$

where the set of parameters  $k_1$ ,  $k_2$  and  $k_3$  model the radial distortion,  $\{u_j^C\ v_j^C\}$  the location of an undistorted pixel with respect to the camera frame  $\{C\}$ ,  $\{\tilde{u}_j^C\ \tilde{v}_j^C\}$  the location of a distorted pixel with respect to the camera frame  $\{C\}$  and  $r$  is the Euclidean distance from the pixel  $\{u_j^C\ v_j^C\}$  to the camera frame  $\{C\}$ .

The same notation can be used to define the tangential distortion induced by the lens as

$$\tilde{u}_j^C = u_j^C + \left( 2p_1 u_j^C v_j^C + p_2 \left( r^2 + 2(u_j^C)^2 \right) \right), \quad (4.4)$$

$$\tilde{v}_j^C = v_j^C + \left( p_1 \left( r^2 + 2(v_j^C)^2 \right) + 2p_2 u_j^C v_j^C \right), \quad (4.5)$$

where  $p_1$  and  $p_2$  are the parameters modelling the radial distortion.

Using these equations, the pinhole camera model can be extended to consider the distortion induced by the lens into the projection. In fact, the derivation of this set up leads to the Jean-Yves Bouguet model with 10 parameters [Bouguet \[2010\]](#). Note that due to the non-linearity of the lens model, the extended model is non-linear too. Thus, the Jean-Yves Bouguet model uses an iterative process to calibrate the camera.

## 4.2 Tool detection

Estimating the position of an object in the space requires establishing a point of reference. When it comes to the laparoscopic tools, such task is not easy due to the lack of features; as shown in Fig. 4.2, a homogeneous black tube and a shape variant jaw is everything that is observed about the tools.

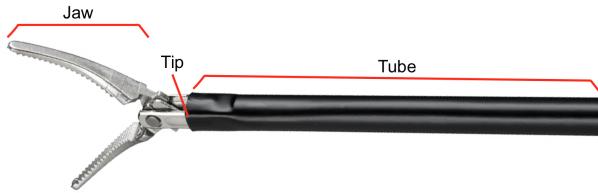


Figure 4.2: Main parts of a laparoscopic tool.

Many works on the state-of-the-art use the tip of the homogeneous black tube, i.e. the tip, for tracking the tool position. Due to the suitability of such approach in the current project, the authors considered to perform it in two steps: (i) segmenting the tool from the background, and (ii) localising the tip of the tube. The methods to cover these two tasks are respectively discussed in Section 4.2.1 and Section 4.2.2.

### 4.2.1 Tool segmentation

Once the images are initially preprocessed, the second task consists in separating the different regions of interest from the rest of the scene so that they can be analysed in later steps. This can be addressed by considering segmentation algorithms.

In the following sections, we analyse three strategies for performing segmentation in the two cases presented in the literature: markerless and marker-based. Note that the techniques are not exclusive and, hence, might be combined for achieving more refined results.

#### 4.2.1.1 Motion segmentation

Since our problem consists in classifying trainees based on the way they perform the proposed activities on the laparoscopic environment, the first approach we considered took advantage of the movement of the tools for differentiating them from the background.

Motion segmentation consists in thresholding the difference between the input frame  $I(x, y, t)$  at time-step  $t$  and the corresponding estimate of the background at the same time-step  $B(x, y, t)$  [Yang et al. \[2012\]](#), as described in the following expression:

$$F(x, y, t) = [ |I(x, y, t) - B(x, y, t)| > \tau ], \quad (4.6)$$

where  $\tau$  is a tolerance threshold. If the difference value is above  $\tau$ , the pixel belongs to the foreground, and background otherwise. An example of how the method processes an input image is shown in Fig. 4.3. Note that, under this paradigm, motion segmentation requires to have a background estimation at each iteration.

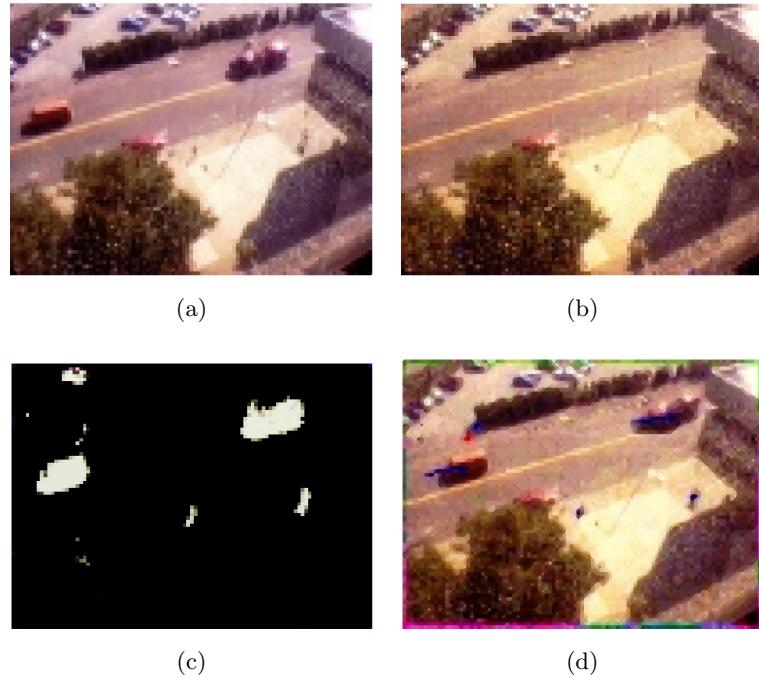


Figure 4.3: Execution of the method. (a) Current image, (b) image composed of the means of the most probable distributions in the background model, (c) foreground pixels, (d) current image with tracking information [Stauffer and Grimson \[1999\]](#).

In general, two main branches for backgrounding are presented in the literature: non-adaptive and adaptive methods. On one hand, the former strategy consider representing the background

based on a certain reference frame (e.g. individual pixel-wise voting method) or a set of previous inputs (e.g. mean value search). On the other hand, the latter approach involves fitting single or multiple distributions to the background, commonly Gaussian distributions [Friedman and Russell \[1997\]](#), [Koller et al. \[1994\]](#), [Ridder et al. \[1995\]](#), [Stauffer and Grimson \[1999\]](#), in order to achieve robustness against significant changes in the scene, issues caused by lighting conditions, among others.

For implementing motion segmentation, we adopted the adaptive background Gaussian mixture models approach presented by Stauffer and Grimson [Stauffer and Grimson \[1999\]](#). The main idea behind this technique is to describe the history a pixel in the scene at time-step  $t$ , denoted by  $X_t$ , using a mixture of  $k$  Gaussian distributions as follows:

$$P(X_t) = \sum_{i=1}^k \omega_{i,t} \cdot \mathcal{N}(X_t | \mu_{i,t}, \Sigma_{i,t}), \quad (4.7)$$

where  $\omega_{i,t}$ ,  $\mu_{i,t}$  and  $\Sigma_{i,t}$  are the normalised weight, the mean and the covariance matrix of the  $i$ -th Gaussian at time-step  $t$ , respectively. These Gaussian distributions are updated as the frames are processed and, specifically, they are adjusted every time a  $X_t$  matches one of them. The update is formulated as

$$\omega_{i,t} = (1 - \alpha)\omega_{i,t-1} + \alpha M_{k,t}, \quad (4.8)$$

where  $\alpha \in [0, 1]$  is the learning rate and  $M_{k,t}$  is a flag which is set to 1 if  $X_t$  is described by the  $k$ -th Gaussian distribution and 0 otherwise. The impact of the parameter  $\alpha$  is illustrated in Fig. 4.4. It can be observed that as the value of  $\alpha$  increases, the mixture tends to believe more in the observations and, thus, noise might be introduced to the model. On the contrary, a low value of  $\alpha$  means that the observation is not taken into account and, hence, the weights may not change.

There are three assumptions the authors follow to hypothetise the background distributions: (i) during the entire video, the background distributions have the most supporting evidence and least variance, (ii) the variance of a distribution resembling a moving object is expected to be larger than the one of the background, and (iii) new objects in the scene may require new distributions or increase the variance of one already in the mixture. Having this in mind, the ratio  $\omega/\sigma$ , being  $\omega$  the weight of the Gaussian within the mixture and  $\sigma$  its corresponding standard deviation, is high when the Gaussian represents the background. Thus, the authors suggested to approach the selection of the background distributions heuristically by taking the

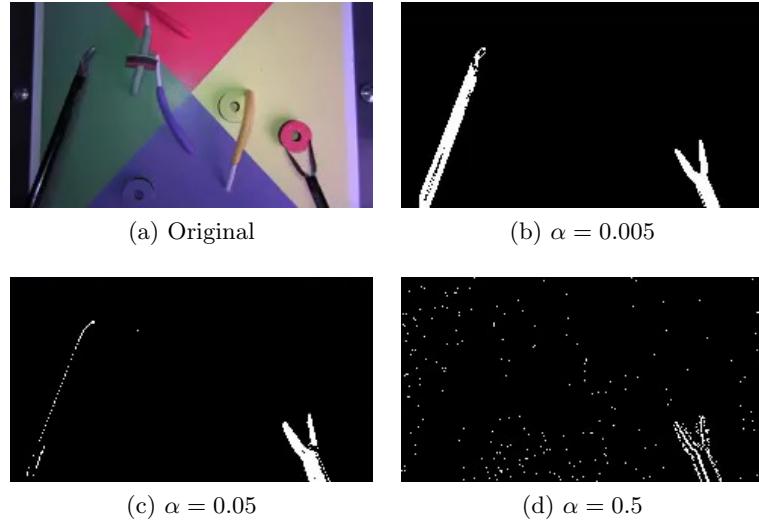


Figure 4.4: Impact of the parameter  $\alpha$  on the segmentation result.

first  $b$  distributions (according to the mentioned ratio) satisfying the following expression

$$B = \arg \min_b \left( \sum_{k=1}^b \omega_k > T \right), \quad (4.9)$$

where  $T \in [0, 1]$  the minimum portion of background within the entire mixture. The effect of this parameter is depicted in Fig. 4.5. It can be observed that small values of  $T$  will discard several Gaussian distributions and, hence, it is not expected to have “memory” of the movement that was happening in the scene. On the other hand, the background becomes a multi-modal distribution with larger values of  $T$  and, thus, when an object stops moving, it is still segmented as it was moving for several frames.

#### 4.2.1.2 Colour segmentation in the distance space

Assuming that the tool differs from everything else in the background in terms of colour, we can think of using colour segmentation. As shown in Fig. 2.2, the tool tube is black while most of the objects in the scene are colourful, i.e. there is high contrast between what is of interest and what is not.

Taking advantage of the fact that darker colours present low intensities in each of the RGB channels, a different domain with only one channel describing the euclidean distance a pixel in RGB has with respect to the origin  $\{0, 0, 0\}$  could be considered. For example, the pixels with RGB intensities  $\{200, 50, 100\}$  and  $\{10, 20, 30\}$  would be mapped in the distance space to

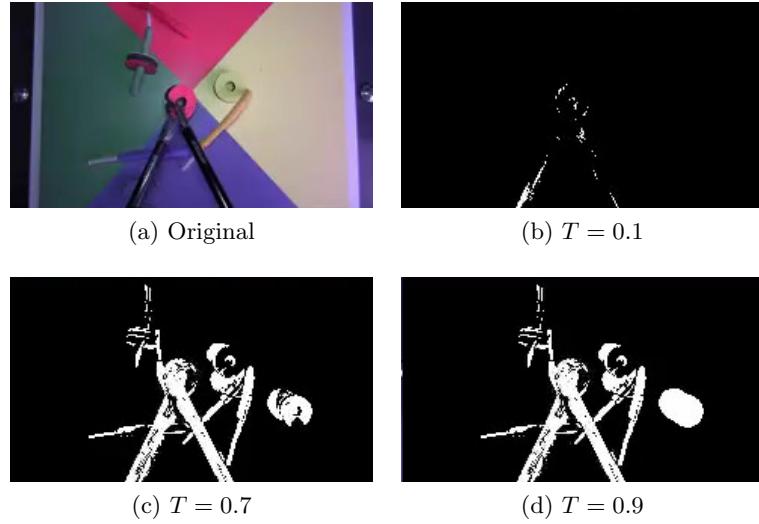


Figure 4.5: Impact of the parameter  $T$  on the segmentation result. The image is taken from a video sequence in which the tools are moving in the training environment.

the intensities 248.74 and 37.42, respectively. A visual example of how images look like in the mentioned space is shown in Fig. 4.6.

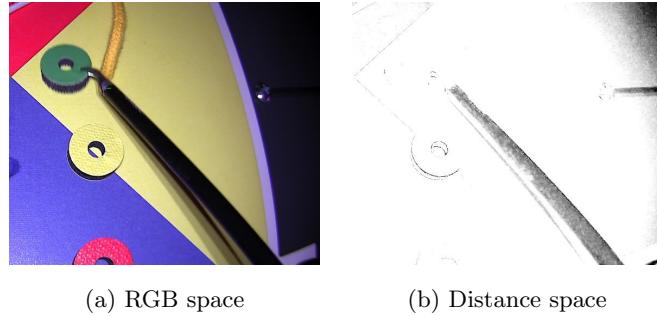


Figure 4.6: Image representation in the RGB and distance spaces. Note that the intensity range of the distance image has been altered.

The interesting part of using the distance space is that the dark colour of the tool can be easily distinguished from the rest if and only if the illumination in the scene favours the assumption that other regions in the scene are not as dark as the tool. It can be observed in Fig. 4.6 that the top right corner does not fulfil this assumption. But, since it would be used along with motion segmentation, the approach should easily remove this non-important regions (as long as the tool is not in contact with these regions).

#### 4.2.1.3 Colour segmentation in the RGB space

As we have discussed previously, most of the approaches in the literature use markers for easing computation while decreasing inaccuracies during the segmentation and, consequently, the pose estimation.

The key under this approximation is to consider markers which are sufficiently distinguishable from the other objects in the scenario so that the segmentation errors are reduced. Note that the concept of markers being distinguishable depends not only on the colour they exhibit, but also on the colour space which is taken into account for performing the processing. Different colour models are used as proxy for segmenting the markers, such as RGB, HSV, Lab, among others. Each of them exhibits interesting properties as a consequence of the way they characterise colour [Gonzalez and Woods \[2006\]](#). Also, observe that less computation is required if the markers are easy to discriminate in the colour space in which they are given.

The representation of an image taken from the training environment once the markers are set on the tools in RGB, HSV and Lab colour spaces is presented in Fig. 4.7. Note that for the sake of uniquely identifying the tools, we selected two different colours of markers. The criteria for choosing them will be discuss in coming sections. It can be seen that the orange marker is easily spotted in the channels: red, blue, value, and  $b$ ; while the purple marker in channels: red, blue, hue, value, lightness, and  $a$ . Having these facts in mind, we decided that RGB was enough for performing marker segmentation. Note that this selection is based on the specific configuration of the training scenario and it may vary depending on the colours of the background.

Once the colour space is selected, the next step consists in determining the method for performing the segmentation. Two approaches can be found in the literature for performing the colour segmentation: thresholding in the different channels with fixed values and learning-based techniques. In the case of the former, the computation is quite fast, but the values might be scene dependent and, in general, it is not robust to changes in the lighting conditions. In the case of the latter, the computation is more costly than in the former approach and requires a training step, but it can cope with lighting distortions. Since the camera has some LEDs, the training environment exhibits different lighting conditions, i.e. the image appears brighter in places close to the centre of the image plane and darker on the boundaries of the same. Therefore, the most suitable approach is the learning-based technique.

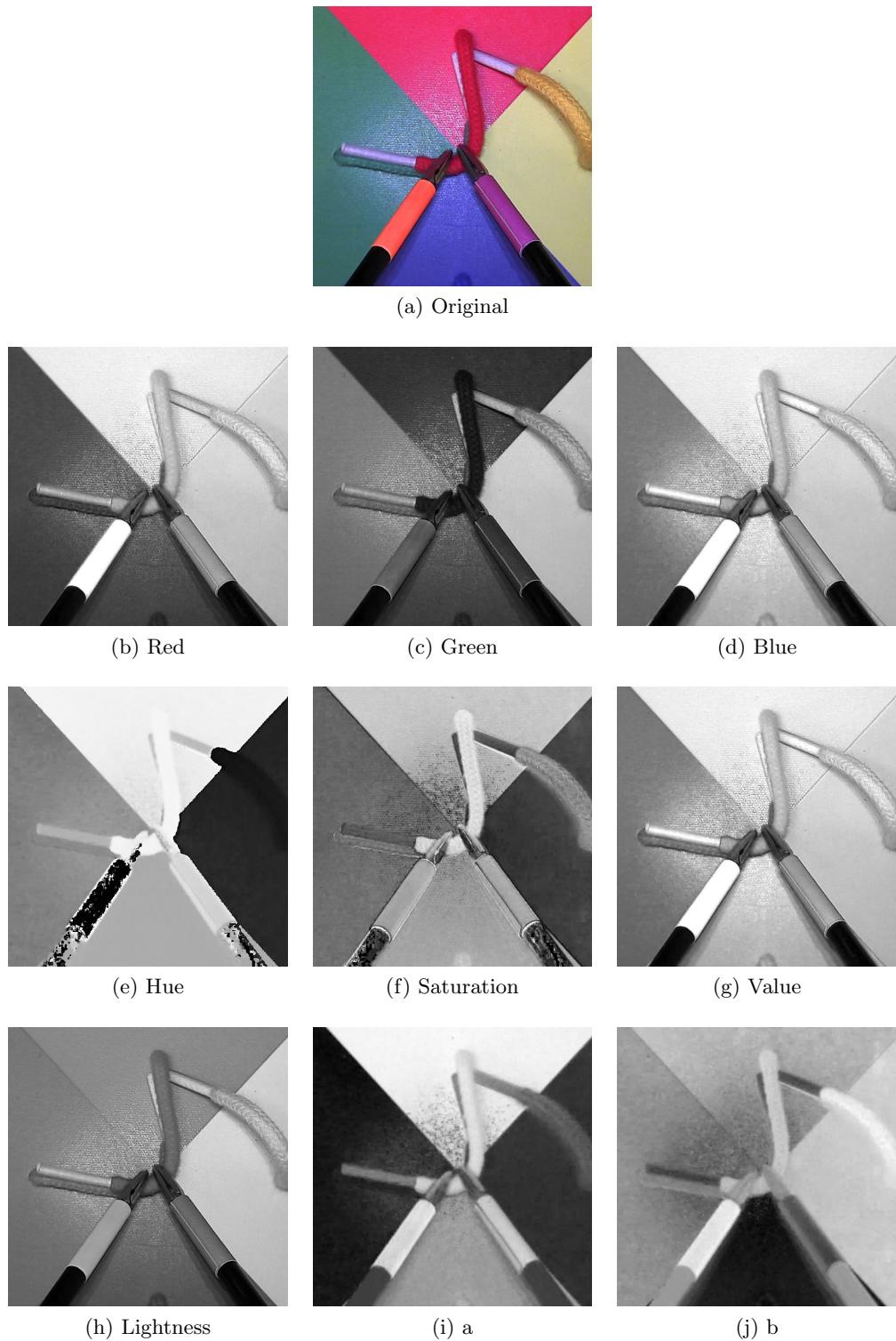


Figure 4.7: Original image and its corresponding representation in RGB, HSV and Lab colour models

### 4.2.2 Tip localisation

The localisation of the middle point of the tip solely relies on the obtained segmentation of the tool. In order to detect the tip of the tube out of the binary mask coming from the segmentation of the tool, the following three steps are considered:

1. Detection of the borders of the tool
2. Computation of the middle-line
3. Intensive search of the tip along the middle line

Firstly, lines are detected on the binary mask for identifying the boundaries of the tool. For this purpose, two approaches were considered: the well-known Hough transform and an *ad hoc* approach. The former operates using the following steps: (i) borders of the binary mask are detected using, for instance, the Canny edge detection algorithm, (ii) the Hough space is computed and (iii) only lines supported by a significant number of points are finally considered. This method requires the modelled geometrical figure to be well-defined in the image, but this might not be always the case of the obtained segmentation.

Taking into account such strong condition, the authors considered appropriate to design an *ad hoc* approach consisting of the following steps: (i) the information obtained from the segmentation, i.e. the binary mask and the bounding box of the Region Of Interest (ROI) (Fig. 4.8a) is used to place a grid on the segmented tool (Fig. 4.8b), (ii) the borders of the blob are computed using the Canny edge detection algorithm, allowing to (iii) specify some points laying on the borders of the tools (Fig. 4.8c), which are used to (iv) fit a line characterising the whole edge of the tool (Fig. 4.8d). The orientation and number of lines that compose the grid will be discussed in Chapter 6.

At the end, the approach working the best and at the same time being computationally efficient would be chosen for detecting the boundaries of the tool.

Secondly, once obtained the borders of the tool and indifferently from the approach used for computing them, the next step is determining a middle-line, i.e. a line which divides the tool into two identical parts through its longitudinal axis, as shown in Fig. 4.9. For this purpose, some middle-points are computed by averaging the location of a pair of points, each one of them laying on a different side of the tool. Then, the obtained set of middle-points is used to fit the

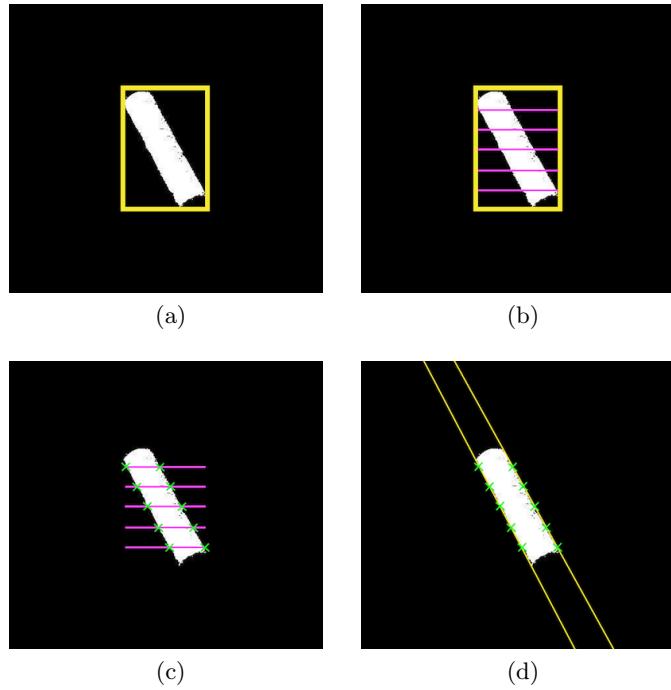


Figure 4.8: *Ad hoc* approach for identifying the edges of the laparoscopic tool. (a) Information obtained from the segmentation process, (b) placement of the grid within the bounding box, (c) localisation of points laying on the borders of the tool, (d) full characterisation of the borders of the tool.

middle-line of the tool using either Least Mean Squares (LMS) or RANdom SAmple Consensus (RANSAC). The approach used to fit the lines will be discussed in Chapter 6.

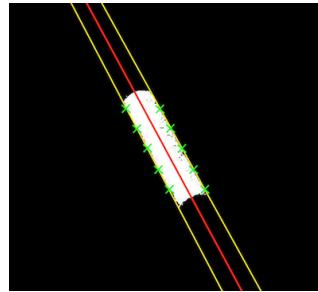


Figure 4.9: Computed middle-line of the tool.

Finally, the problem of localising the tip is reduced to perform an exhaustive search throughout the middle-line, as exemplified in Fig. 4.10. Such search starts from one of the previously computed middle-points and moves towards the location of the tip, until a significant amount of zeros, i.e. area not belonging to the tube, is found in the mask. How the search direction along the middle-line is decided will be detailed in Chapter 6.

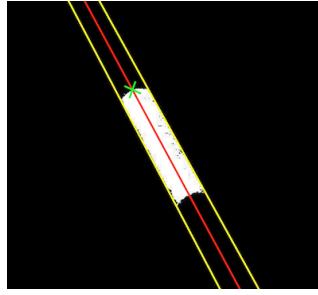


Figure 4.10: Localisation of the tip.

The lines computed to locate the tip of the tool can be useful to extract the width of the tool in the image, which might be useful for later algorithms. To extract such information, (i) a perpendicular line to the middle-line has to be computed (Fig. 4.11a) and (ii) the intersection of it with the edges of the tool have to be determined (Fig. 4.11b). Then, the Euclidean distance between both intersections will determine the width of the tool in the image.

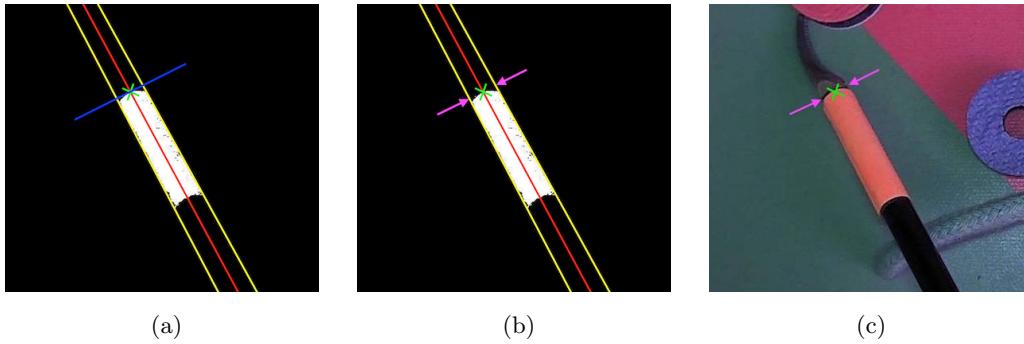


Figure 4.11: Characterisation of the width of the tool. (a) Computation of the perpendicular line to the middle-line, (b) characterisation of the width of the tool, (c) correspondence between the extracted information and the original frame.

At the end, Fig. 4.11c proves that even though the detection and width characterisation of the tip of the tool is done in the binary space, the extracted information accurately describes the tool in the input frame. Moreover, note that doing the same approach in the input image would not be possible due to the existence of many geometrical forms, including the one that the proposed method relies on.

## 4.3 Tool pose estimation

At this point, some methods for identifying a specific position of the tool at each frame have been introduced. Thus, assuming the correctness of the obtained information out of those methods, the extraction of the position of the tool is twofold: first, the 3D position is estimated with a single-camera, and second, the obtained set of positions are filtered to remove the noise. These two parts are respectively tackled in Section 4.3.1 and Section 4.3.2.

### 4.3.1 3D pose estimation

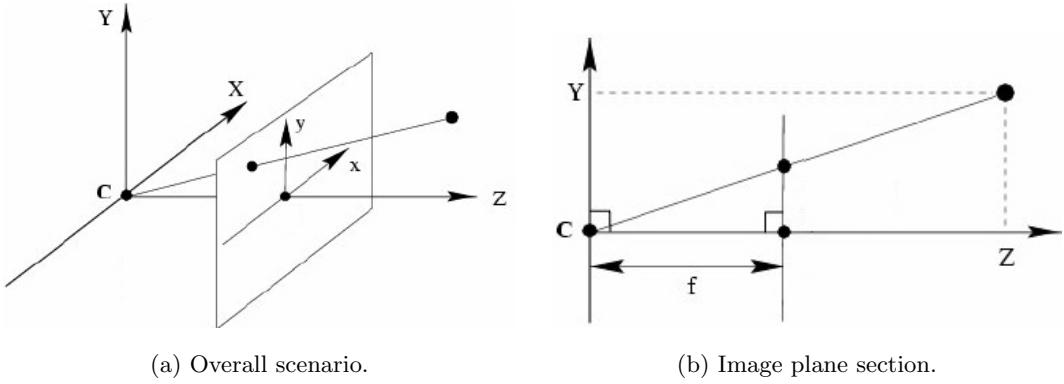
A stereo-vision system is usually required for determining the 3D position  $p^R = \{x^R \ y^R \ z^R\}^T$  of an object with respect to a reference frame  $\{R\}$ ; aiming to achieve the same task with a single-camera system leads to an ill-posed problem. Thus, the latest approach will only be useful to create depth maps or to estimate the 3D position of an object if additional information is available, e.g. odometry of the camera or known dimensions of the objects in the scene among others.

In the laparoscopic environment presented in Chapter 2, the location of the camera  $C$  with respect to the world frame  $\{W\}$  can be assumed to be static but it is unknown. Therefore, the 3D pose  $p_i^C = \{x_i^C \ y_i^C \ z_i^C\}^T$  of the tool  $i$ , being  $i$  a particular label for each tool in the scene, will be estimated with respect to the camera frame  $\{C\}$ , obtaining  $\tilde{p}_i^C = \{\tilde{x}_i^C \ \tilde{y}_i^C \ \tilde{z}_i^C\}^T$ .

For addressing the 3D pose estimation task, the reviewed works in Chapter 1 have been considered. Since markerless approaches are preferred to keep the final framework as less marker dependent as possible, the method of [Tonet et al. \[2007\]](#) using the pinhole camera model and the approach of [Zhou and Payandeh \[2014\]](#) using 3D geometrical properties have been analysed. Thus, despite the high accuracy reported in [Shin et al. \[2014\]](#), the work of using Haralick's algorithm has not been deeply studied.

#### 4.3.1.1 Pinhole camera model

As it has been introduced in Section 4.1, the pinhole camera model (Fig. 4.12) is the most trivial yet powerful method to project a 3D scene onto an image plane, where the modelled camera is considered to be an ideal pinhole camera, i.e. without lens and all the problems that it implies, e.g. distortion, blurring or finite field of view, among others.

Figure 4.12: Pinhole camera model [Forsman \[2011\]](#).

The pinhole camera model relates the unknown 3D position of an object of interest  $p_i^C = \{x_i^C \ y_i^C \ z_i^C\}^T$  referenced at the camera frame  $\{C\}$  with its projection  $o_i^I = \{u_i^I \ v_i^I\}$  in the image plane  $I$ . Such relation is decomposed in the  $XZ$  and the  $YZ$  planes and mathematically represented using the similar triangles theorem,

$$\frac{f}{z_i^C} = \frac{u_i^I - u_I^C}{x_i^C} = \frac{v_i^I - v_I^C}{y_i^C}, \quad (4.10)$$

where  $f$  and  $\{u_I^C \ v_I^C\}$  are the focal length and principal point of the camera  $C$ , respectively. Such parameters are known as camera intrinsic, and are extracted from the calibration procedure proposed in Section 4.1. Note that instead of considering a particular focal length for each axis, i.e.  $f_x$  and  $f_y$ , the average  $f$  of those is considered in this algorithm.

The model described in Eq. 4.10 leads to an ill-posed problem, since the  $x_i^C$  and  $y_i^C$  coordinates of the object can only be known up to a scale factor depending on  $z_i^C$ . Thus, additional information must be introduced in the system to solve the ill-posed problem. Introducing the diameter of the tool  $d_{tool}$  seems to be a good option, since the cylindrical shape is homogeneous along the tube and its size is non-dependant on the point of view.

In order to handle the projective relation of the diameter of the tool  $d_{tool}$  and its projection onto the image plane  $I$ , denoted as  $\Delta P_i$ , the previously introduced pinhole camera model has to be expanded. Such reformulation is possible considering that the points of the real tool defining the boundaries of the tip have the same  $Z$  component, or in other words, that the segment joining them is parallel to the image plane  $I$ .

Once the diameter of the tool  $d_{tool}$  is known and some constraints regarding the projection of objects onto the image plane are relaxed, the following relation applies:

$$\frac{f}{z_i^C} \approx \frac{\Delta P_i}{d_{tool}}. \quad (4.11)$$

Finally,  $z_i^C$  can be directly computed from Eq. 4.11, which allows to solve the initial ill-posed problem setup by the pinhole camera model. Then, this method estimates the 3D position of the tool  $p_i^C = \{x_i^C \ y_i^C \ z_i^C\}^T$  according to

$$z_i^C \approx \frac{f \cdot d_{tool}}{\Delta P_i}, \quad (4.12)$$

$$x_i^C = z_i^C \frac{(u_i^I - u_I^C)}{f}, \quad (4.13)$$

$$y_i^C = z_i^C \frac{(v_i^I - v_I^C)}{f}. \quad (4.14)$$

The reformulation of the pinhole camera model done to solve the initial ill-posed problem might lead to some inaccuracies. Specifically, the accuracy of the estimated 3D position will decrease from the centre of the image to its boundaries, as the distortion of the lens and the approximation error increase.

#### 4.3.1.2 Edge-crossing planes

The other considered approach for estimating the 3D position of each tool  $i$  is the method introduced in Fig. 4.13, proposed in Zhou and Payandeh [2014]. The location of the tip  $P_i$  and the orientation  $\theta_i$  of the tool  $i$  with respect to the camera frame  $\{C\}$ , are estimated thanks to the projection of the tool edges and tip onto the image plane, the camera intrinsic parameters and the knowledge of the real width of the tool.

First of all, for each projected edge  $\bar{E}_{i,j}$  of the tool  $i$  onto the image plane, being  $j = 1, 2$ , a plane  $\Omega_{i,j}$  is defined to include the projected edge  $\bar{E}_{i,j}$  and the vector  $\bar{CE}_{i,j}$ , which goes from the camera frame  $\{C\}$  to the edge  $\bar{E}_{i,j}$ . Then, the direction of the unitary vector  $\bar{u}_{\Omega_{i,j}}$ , which is perpendicular to the plane  $\Omega_{i,j}$ , can be defined as

$$\bar{u}_{\Omega_{i,j}} = \bar{u}_{E_{i,j}} \times \bar{u}_{CE_{i,j}}. \quad (4.15)$$

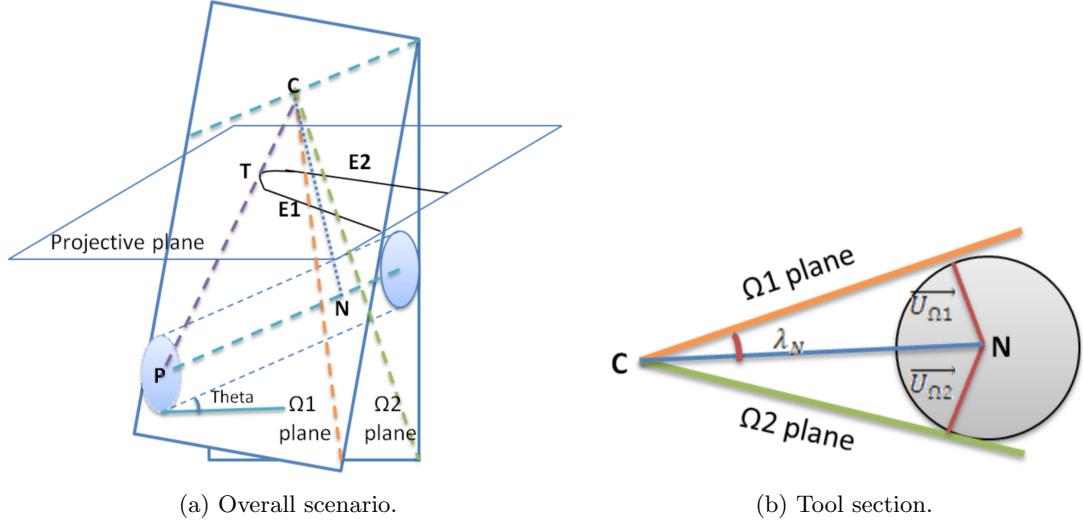


Figure 4.13: Edge-crossing planes method applied on one tool Zhou and Payandeh [2014].

The obtained vectors  $\vec{u}_{\Omega_{i,1}}$  and  $\vec{u}_{\Omega_{i,2}}$  point towards the centre of the real tool. Those vectors can be used to compute the angle  $\lambda_i$  as follows:

$$\tan \lambda_i = \frac{\left| \vec{u}_{\Omega_{i,1}} + \vec{u}_{\Omega_{i,2}} \right|}{\left| \vec{u}_{\Omega_{i,1}} - \vec{u}_{\Omega_{i,2}} \right|}. \quad (4.16)$$

Leveraging the knowledge of the diameter of the tool  $d_{tool}$  and the previously computed angle  $\lambda_i$ , the length of the vector  $\overline{CN}_i$  can be obtained straightforward by applying the trigonometric rule

$$|CN_i| = \frac{0.5 d_{tool}}{\sin \lambda_i}, \quad (4.17)$$

which allows to extract the orientation of the unitary vector  $\vec{u}_{CN_i}$  as

$$\vec{u}_{CN_i} = |CN_i| \frac{\vec{u}_{\Omega_{i,1}} + \vec{u}_{\Omega_{i,2}}}{\left| \vec{u}_{\Omega_{i,1}} + \vec{u}_{\Omega_{i,2}} \right|}. \quad (4.18)$$

Finally, the location of the tip  $P_i$  with respect to the camera frame  $\{C\}$  can be estimated by defining the vector  $\overline{CP}_i$  as

$$\overline{CP}_i = \frac{|CN_i|}{\vec{u}_{CT_i}} \frac{\vec{u}_{CT_i}}{\vec{u}_{CN_i}}, \quad (4.19)$$

and the orientation of the tool by determining the vector  $\overline{NP}_i$  as

$$\overline{NP}_i = -\overline{CN}_i + \overline{CP}_i. \quad (4.20)$$

After the analysis of the method proposed in Zhou and Payandeh [2014], it is not considered for the final framework; even if the general idea of the method makes sense from a geometrical point of view, the paper does not explain some details that are necessary for implementing such algorithm.

### 4.3.2 Data filtering

The outcome of any pose estimation algorithm might have outliers and noise. Thus, filtering such data is indispensable to get a more realistic representation of the path followed by the tools. The most well-known filters used for this purpose are the Kalman Filter (KF) Cuevas et al. [2005] and its extensions for handling non-linear systems, such as the Extended Kalman Filter (EKF) and the Unscented Kalman Filter (UKF). In all cases, the motion model of the objects to track and the observation model of the system has to be determined.

In the current filtering problem, the state-transition model of the tools can be linearly represented as

$$p_{i,t} = p_{i,t-1} + \Delta T \dot{p}_{i,t-1} + \frac{\Delta T^2}{2} \ddot{p}_{i,t-1}, \quad (4.21)$$

where  $p_{i,t}$  is the 3D position of the tool  $i$  at time-step  $t$ ,  $\Delta T$  is the lapse time between consecutive time-steps and  $p_{i,t-1}$ ,  $\dot{p}_{i,t-1}$  and  $\ddot{p}_{i,t-1}$  represent the 3D position, velocity and acceleration of tool  $i$  at time-step  $t - 1$ , respectively.

Regarding the observation model, it can also be linearly represented since the system directly observes the position of the tools. However, since neither the kinematics nor the measurements will be infinitely precise, their error can be assumed to follow a Gaussian distribution with zero mean. Thus, under such assumptions and dealing with linear systems, the KF seems to be a good approach for filtering the estimated 3D position  $\tilde{p}_{i,t}^C = \{\tilde{x}_{i,t}^C \ \tilde{y}_{i,t}^C \ \tilde{z}_{i,t}^C\}^T$  of any tool  $i$  in the scene over time.

The KF performs two stages at each time-step  $t$ . First, it predicts the state of the tools  $\hat{X}_{i,t}$  using the previously introduced state-transition model, which can be directly rewritten as

$$\hat{X}_{i,t} = A_{i,t} X_{i,t-1} + W_{i,t}, \quad (4.22)$$

which can be expressed in terms of mean  $\hat{\mu}_{i,t}$  and uncertainty  $\hat{P}_{i,t}$  as follows:

$$\hat{\mu}_{i,t} = A_{i,t} \mu_{i,t-1}, \quad (4.23)$$

$$\hat{P}_{i,t} = A_{i,t} P_{i,t-1} A_{i,t}^T + Q_{i,t}, \quad (4.24)$$

where  $X_{i,t} = [x_{i,t} \dot{x}_{i,t} y_{i,t} \dot{y}_{i,t} z_{i,t} \dot{z}_{i,t}]^T$  is the state vector of the tool  $i$ ,  $A_{i,t}$  (Eq. 4.25) defines the transition model,  $W_{i,t}$  (Eq. 4.26) models the acceleration term of the transition model as Gaussian noise which is expressed in terms of uncertainty by  $Q_{i,t}$  (Eq. 4.27). Note that  $Q_{i,t}$  is the covariance matrix of  $W_{i,t}$ , which assumes independence between the different Degrees Of Freedom (DOF).

$$A_{i,t} = \begin{bmatrix} 1 & \Delta T & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \Delta T & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & \Delta T \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (4.25)$$

$$W_{i,t} = \mathcal{N}_{i,t}(0, \sigma_w) \left[ \frac{\Delta T^2}{2} \Delta T \frac{\Delta T^2}{2} \Delta T \frac{\Delta T^2}{2} \Delta T \right]^T \quad (4.26)$$

$$Q_{i,t} = \sigma_w^2 \begin{bmatrix} \frac{\Delta T^4}{4} & \frac{\Delta T^3}{2} & 0 & 0 & 0 & 0 \\ \frac{\Delta T^3}{2} & \Delta T^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\Delta T^4}{4} & \frac{\Delta T^3}{2} & 0 & 0 \\ 0 & 0 & \frac{\Delta T^3}{2} & \Delta T^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \frac{\Delta T^4}{4} & \frac{\Delta T^3}{2} \\ 0 & 0 & 0 & 0 & \frac{\Delta T^3}{2} & \Delta T^2 \end{bmatrix} \quad (4.27)$$

After the prediction stage, the KF updates the predicted state vector  $\hat{X}_{i,t}$  by comparing the observed information  $\tilde{Z}_{i,t} = [\tilde{x}_i^C \tilde{y}_i^C \tilde{z}_i^C]^T$  with the expected observation  $Z_{i,t}$  obtained with the measurement model

$$Z_{i,t} = H_{i,t} \hat{X}_{i,t} + V_{i,t}, \quad (4.28)$$

where  $H_{i,t}$  (Eq. 4.29) defines what should be measured according to the predicted position  $\hat{X}_{i,t}$  and  $V_{i,t}$  (Eq. 4.30) models the Gaussian noise in the observation.

$$H_{i,t} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \quad (4.29)$$

$$V_{i,t} = \mathcal{N}_{i,t}(0, \sigma_v) [1 \ 1 \ 1]^T \quad (4.30)$$

According to the measurement model described in Eq. 4.28, the update of the predicted state vector is twofold: firstly, the Kalman gain  $K_{i,t}$  is computed as

$$K_{i,t} = \hat{P}_{i,t} H_{i,t}^T \left( H_{i,t} \hat{P}_{i,t} H_{i,t}^T + R_{i,t} \right), \quad (4.31)$$

to, secondly, update the mean  $\mu_{i,t}$  and uncertainty  $P_{i,t}$  of the state vector  $X_{i,t}$  as follows:

$$\mu_{i,t} = \hat{\mu}_{i,t} + K_{i,t} \left( \tilde{Z}_{i,t} - H_{i,t} \hat{\mu}_{i,t} \right), \quad (4.32)$$

$$P_{i,t} = (I - K_{i,t} H_{i,t}) \hat{P}_{i,t}, \quad (4.33)$$

where  $R_{i,t} = \sigma_v^2 \mathbb{I}_{3 \times 3}$  is the covariance of  $V_{i,t}$ , in which independence between the different measurements is assumed.

Once presented all the steps of the KF, it can be noticed that the mean  $\mu_{i,t}$  and uncertainty  $P_{i,t}$  of the state vector  $X_{i,t}$  has to be initialised at  $t = 0$ . Then, apart from the  $\Delta T$  in the motion model, only two parameters have to be tuned:  $\sigma_w$  and  $\sigma_v$ . The parametrisation given to these variables will be discussed in Chapter 6.

## 4.4 Skill assessment

As it was stated in Section 1, the aim of this project is to be able to track the laparoscopic tools in order to evaluate an operator's skill level and to improve the training efficiency. In order to do so, once the tool's path has been filtered, it is necessary to analyse it.

Our initial aim was to collaborate with another group who would train new operators in the basics of laparoscopic surgery. Videos were to be recorded in the beginning and in the end of the

training of each subject and these videos would be used to train a classifier able to perform the desired skill assessment. Due to unexpected problems, it was not possible to finish the training of the new operators and therefore data relative to skilled and unskilled operators was not available. As a consequence of this setback, this part of the project could not be implemented nor validated with real instances. Nevertheless, we present a theoretical approach in which we outline the features we would extract from the paths to determine the level of training of a subject.

The problem of evaluating the skill level based on the tools' paths can be seen as a classification/regression problem. To apply machine learning techniques to this problem, first it is necessary to be able to extract features from the filtered path to use as input to the classifier. These features could be computed from the path as a whole or calculated using windows of pre-defined length. Some possible features to be extracted from the path as well as the assumption behind them are listed below. In these equations,  $x(t)$  and  $y(t)$  are the path coordinates at time  $t$ ,  $t_i$  is the initial time and  $t_f$  is the final time of the task.

- **Mean absolute velocity:** The mean velocity can be easily calculated from the path points and it is expected to be a strong indicator of skill level, i.e. more skilled operators tend to move the tools in a faster manner.

$$\text{mean\_vel} = \frac{1}{t_f - t_i} \int_{t_i}^{t_f} \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2} dt \quad (4.34)$$

- **Mean absolute acceleration (jitter):** More skilled operators will have steadier hands and this will likely lead to smaller mean absolute acceleration values.

$$\text{jitter} = \frac{1}{t_f - t_i} \int_{t_i}^{t_f} \sqrt{\ddot{x}(t)^2 + \ddot{y}(t)^2} dt \quad (4.35)$$

- **Thinking time:** More skilled operators will likely not spend large amounts of time thinking about what to do next during a task. This thinking time can be estimated by calculating the amount of time the tools' absolute velocities were below a chosen threshold.

$$\text{thinking\_time} = \int_{|v(t)| \geq T} dt, \text{ where } |v(t)| = \sqrt{\dot{x}(t)^2 + \dot{y}(t)^2} \quad (4.36)$$

- **Total task time:** Newer operators will take longer to accomplish the same task as compared to more experienced operators. This can be used to help in the skill assessment

calculations.

$$\text{duration} = t_f - t_i \quad (4.37)$$

- **Path smoothness:** The path described by the tools of a skilled operator is expected to be smoother than the path from an inexperienced operator. In this particular case, we adopt the smoothness definition presented in [Moll et al. \[2014\]](#),

$$\text{smoothness} = \sum_{i=2}^{n-1} \left( \frac{2 \left( \pi - \arccos \left( \frac{a_i^2 + b_i^2 - c_i^2}{2a_i^2 b_i^2} \right) \right)}{a_i + b_i} \right)^2, \quad (4.38)$$

where  $a_i = \text{dist}(s_{i-2}, s_{i-1})$ ,  $b_i = \text{dist}(s_{i-1}, s_i)$ ,  $c_i = \text{dist}(s_{i-2}, s_i)$ ,  $s_i$  is the  $i^{th}$  point of the path, and  $\text{dist}(p, q)$  is the Euclidean distance between points  $p$  and  $q$ .

More features can still be proposed, and inspiration for such features can be drawn from path planning theory [Moll et al. \[2014\]](#).

It is important to point out that the acquisition of the type of data needed to train such a classifier is very hard and expensive to obtain. It is very likely that the eventual implementation of such a classifier will use a small dataset for training, while it is trying to perform a complex classification task. In this situation it is important to keep in mind the overfitting problem to avoid creating models that are too complex and that describe the training data well but does not generalise well to new data points.

# Chapter 5

## Framework design

Based on the techniques detailed in Chapter 4, we designed different instances of the framework presented in Chapter 2. In this section, we present the characteristics of each framework, point their issues out and discuss about possible improvements.

### 5.1 Framework 1

The outline of Framework 1 is presented in Fig. 5.1. Once the video sequence of the training activity is acquired, the different frames are individually processed using the following steps:

1. Each frame is transformed using the camera intrinsic parameters which were determined using the camera calibration algorithm.
2. The corrected image is then segmented using colour segmentation in the distance space and motion segmentation. Note that using the two sources of information may enhance the segmentation results.
3. A blob detection algorithm is used on the obtained binary mask to detect large connected component which may represent the laparoscopic tools.
4. Information regarding the detected areas is passed to the KF so previous tracks can be associated with the new observations.

5. If KF corroborates that a blob has been observed previously, tip and tool detection algorithms are applied for measuring the width of the tool which is lately used for estimating the 3D pose.

In this framework, we do not consider classification algorithms since the unique data that was available by that time was the one we took ourselves and the segmentation was not promising enough.

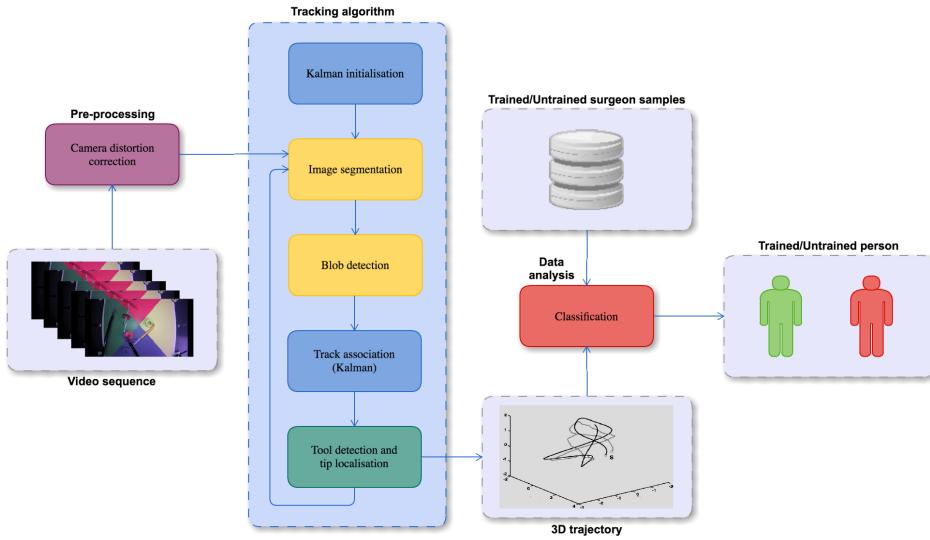


Figure 5.1: Workflow of the framework 1.

Although the framework described above sounds convincing in terms of its capabilities to perform the task, several issues were observed when using it with real video sequences. The problems were concentrated around one out of the four processes of the framework: detection.

The problems regarding the detection using motion and colour segmentation in the distance space are detailed as follows:

- The concept of motion segmentation requires that the laparoscopic tools are constantly on movement which might not always be the case. Hence, depending on the values for  $\alpha$  and  $T$ , the record of the tools inside the Gaussian mixture may disappear quickly and no more observations of the tools are made.
- Due to the way the multiple hypothesis tracker is designed, the framework is not able to solve the ambiguity when different objects are continually crossing. As a result, very noisy observations are given at the end to the KF which cannot deal properly with them. Note

that this issue can be handled by uniquely identifying the two tools. In this case, if one of the tools is overlapping the other one, at least the measurement of one of them would be correctly given to the following processes.

- Distortions coming from lighting conditions, such as glare and shadows, affect the estimation of the Gaussian distributions representing the background.

**Shadows:** as stated in [Stauffer and Grimson \[1999\]](#), since the shadows are moving along with the objects, they are usually classified as a region of interest as illustrated in Fig. 5.2. Although combining the knowledge coming from the motion and colour segmentation may refine the results, the problem related to the shadows remains in the process since they also exhibit dark colours.

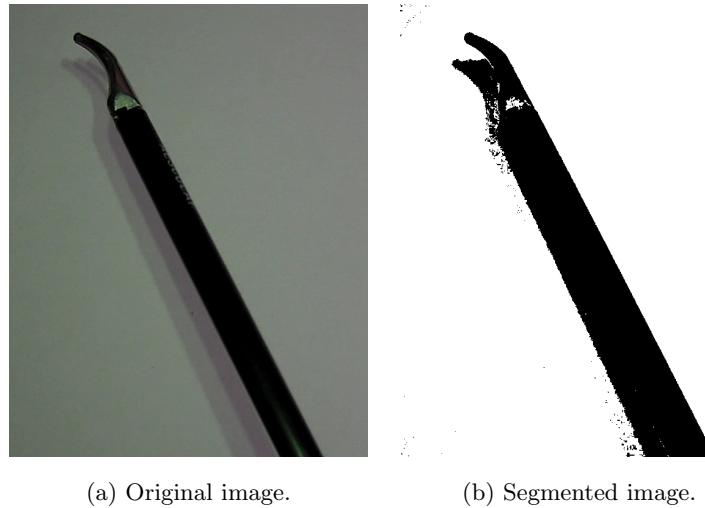


Figure 5.2: Original image exhibiting shadows and its corresponding segmentation.

**Glare:** in our particular case, the illumination in the scene is given not only by external light but also by some LEDs the camera has got. This means that the closer the tool to the centre, the brighter the area will look in the image. The same effect is seen when the tool approaches the camera. In this way, there should be more than one Gaussian representing the same object since the intensity values vary all over the tool. The result is, as expected, incomplete or irregular segmentation of the tools (i.e. holes on the segmented area) as presented in Fig. 5.3.

- Motion segmentation will detect all the moving objects in the scene. Since we are considering a training environment in which the trainee needs to perform different tasks with the laparoscopic tools, such as moving some other objects in the scene, several objects along

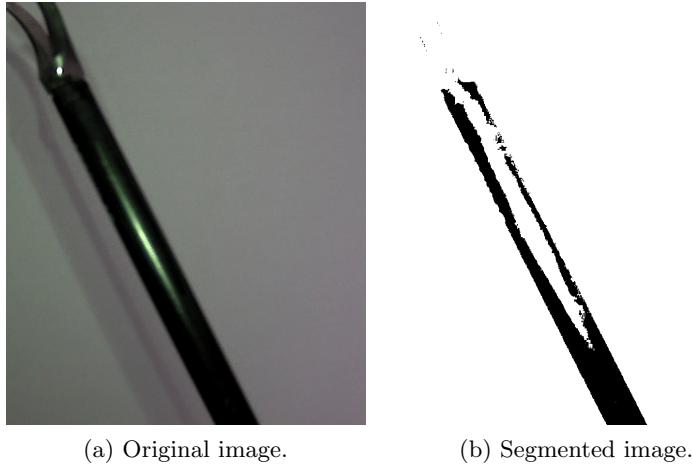


Figure 5.3: Original image exhibiting glare and its corresponding segmentation.

with the tools are going to be detected and, thus, refinement of the segmentation should be carried out. In this case, we combined the knowledge of motion and the distance space so that the segmentation was enhanced. However, other objects in the scene may look as dark as the tool and, hence, false alarms appear in the process.

- The background subtraction method requires setting up the learning rate  $\alpha$  and the minimum background portion  $T$ :

**$\alpha$ :** a high value means very noisy observations while a low value implies less credibility on the current observation and, consequently, the Gaussian mixture relies on its records which may be a problem when new objects enter the scene.

**$T$ :** affects in a way the memory of the Gaussian mixture since the lower the value of  $T$ , the less the number of Gaussian distributions representing the background. As a result, the mixture will not be able to characterise several moving objects with different colours. On the other hand, if the value of  $T$  is very high, the background is modelled with a multi-modal distribution, but changes in motion are not properly addressed since objects that stop at a certain time-step may be still detected in the following frames.

## 5.2 Framework 2

As we have seen in the previous sections, an accurate estimation of the width of the tool is required in order to reduce the error in XYZ estimation. However, we observed that the

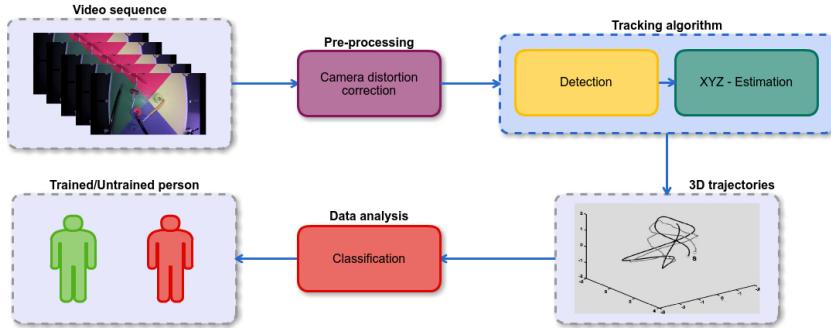


Figure 5.4: Workflow of the framework 2.

markerless segmentation approximations are affected by several environmental conditions and, hence, the final result may not be as good as desired. Thus, we adopted the common approach in the state-of-the-art: to put markers on the tools.

The outline of Framework 2 is presented in Fig. 5.4. Once the video sequence of the training activity is acquired, the different frames are individually processed using the following steps:

1. Each frame is transformed using the camera intrinsic parameters which were determined using the camera calibration algorithm.
2. The corrected image is then segmented using colour segmentation in the RGB space.
3. A blob detection algorithm is used on the obtained binary mask to detect large connected component which may represent the laparoscopic tools.
4. Tool and tip detection algorithms are applied on the different blobs to measure the width of the tools in the image plane.
5. Based on this information, the 3D pose of the object is estimated using the approaches discussed in Chapter 4.
6. Information regarding the XYZ position of the tool is given to the KF so previous tracks can be associated with the new observations.

The previous steps are repeated until all the frames in the provided video sequence are processed so that a 3D trajectory is obtained. Finally, features are extracted from the path and compared against data of skilled and unskilled persons to determine the level of skill of the subject.

The second proposed framework is able to cope with some of the issues encountered using the initial approximation. However, it is not flawless. The problems regarding the marker-based approach are detailed as follows:

- When the laparoscopic tools are out of focus, the contrast between the tools and the background decreases due to blurring effects. If the colour is preserved after blurring there is no major issue. However, we observed that in fact the distortion slightly mixes the colours of the tools and the background. The resulting colour may look quite similar to the marker colour and, consequently, it may be incorrectly classified. For instance, we observed this issue when an orange marker was out of focus on top of the red and yellow or when a purple marker appeared blurred on top red and blue. Moreover, when the illumination conditions are poor, the colours of the markers tend to look like colours of the background and, thus, they are not detected by the framework. However, note that this issue is not a direct consequence of the framework itself, but a result of the data acquisition system and the complexity of the training environment (colourful pattern).
- As described by authors using marker-based approaches, the markers should be visible in every moment in order to have good detection of the tools. However, note that, unlike some of the approaches in the literature, our approach can cope with overlap between the markers in the sense that we will obtain correct observations as long as the non-overlapping areas are big enough to be detected and the visible part of the marker corresponds to the one closer to the grasper. Also, this issue is mitigated by the properties of the KF for dealing with missing observations at a certain time step.
- As we mentioned before, the method for pose estimation requires some assumptions and, hence, we will be incurring in some inaccuracies. However, since the idea is to compare the trajectories of different trainees under the same evaluation conditions, this distortion could be considered negligible.
- The selection of the markers is essential as they should be distinguishable enough from the other objects in the environment. This means that the decision for choosing the colour of the markers is not universal and, hence, it should be adapted depending on the training scenario.
- The segmentation is highly dependent on the training data and, hence, outliers may appear in the process. Hence, additional pre-processing is required in order to use the data coming from the pose estimation step in the KF.
- The fact that classification is used inside the segmentation process implies that the process is more costly than before. Moreover, depending on the classification technique the overall complexity can worsen dramatically. Since the framework is desired to be used for assessing

trainers in real time this may suppose an inconvenient. In Chapter 6, we discuss about two techniques we implemented for reducing the complexity of the overall process to the same of the initial framework.

# Chapter 6

## Implementation

The proposed framework has been implemented in MATLAB to cover from the reading of the previously acquired video of the laparoscopic centre training to the storage and plotting of the computed 3D position of the tools. At this point, it has to be recalled that the data comparison part for assessing the skill of a subject has not been implemented due to the problems stated in Chapter 3.

Since all the used methods have already been introduced in Chapter 4 and the general flow of the framework has been explained in Chapter 5, the current section uniquely aims to give an insight of the most crucial points of the implementation of the framework.

### 6.1 Camera calibration toolbox

Instead of implementing the Jean-Yves Bouguet model from scratch, the already built-in camera calibration toolbox using such method in MATLAB has been used. As schematised in Fig. 6.1, the method consists in iteratively minimising the error between the theoretical non-distorted projection of a known pattern and the real projection of such pattern onto the image plane.

Specifically, the camera calibration was carried out using a checkerboard, the squares of which measured 21 mm of side. Apart from this information, the toolbox also requires some images of the checkerboard gathered with the camera that has to be calibrated. Thus, a total of 19 images were initially considered; the higher the amount of data used to calibrate the camera, the more accurate the estimation of the parameters would be.

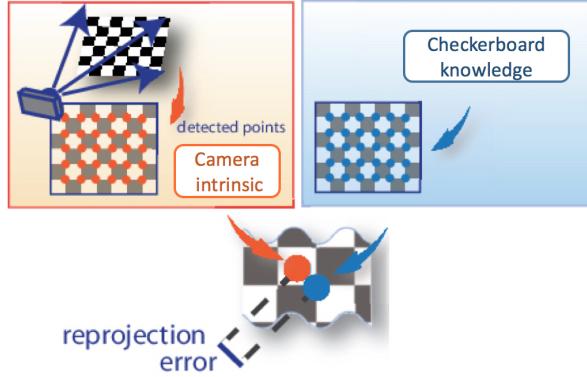


Figure 6.1: Camera calibration workflow [MathWorks \[2016b\]](#).

After obtaining the first estimation of the parameters, which had an accuracy of 2.63 pixels, it was spotted out that 3 images were taken while the checkerboard was slightly bent. Thus, the above procedure was repeated but removing those images, to finally obtain an accuracy of 0.92 pixels.

## 6.2 Deciding the colour of the markers

The colour of the markers is essential in the segmentation process and, thus, should be carefully chosen. Evidently, the more the distant the colour of the markers to the components of the background, the better and easier the segmentation. Having a close look at Fig. 2.2 and Fig. 6.2, it can be seen that based on the colours of the scenario, orange or cyan should be considered.

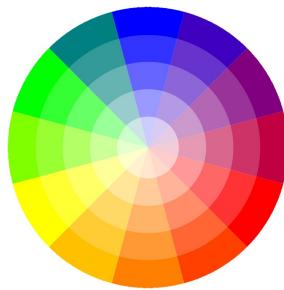


Figure 6.2: RGB colour wheel [Designs \[2008\]](#).

## 6.3 Pixel classification

As we described in previous sections, illumination conditions are not the same in the training scenario not only because of external lighting sources but also due to the light coming from the

LEDs on the camera. Thus, the second framework incorporates machine learning techniques for determining whether a pixel represents a marker or not. Then, the process consists in taking each pixel of a given image and label it using a trained classifier.

The question that arises at this point is which classifier is preferred in our particular scenario. For answering the question, we evaluated the supervised machine learning techniques provided by MATLAB according to previous experience with the techniques and some features presented in Table 6.1.

| Classifier            | Multi-class support | Prediction speed                    | Memory usage                             |
|-----------------------|---------------------|-------------------------------------|--|
| Decision trees        | Yes                 | Fast                                | Fast                                     |
| Discriminant analysis | Yes                 | Fast                                | Small for linear and large for quadratic |
| SVM                   | No                  | Medium if linear;<br>slow otherwise | Medium                                   |
| KNN                   | Yes                 | Medium                              | Medium                                   |

Table 6.1: Comparison of different machine learning techniques for classification purposes [MathWorks \[2016a\]](#).

The analysis is presented as follows:

- **Decision trees:** In terms of the features presented in the table, decision trees seem to be an appealing technique for performing classification. However, it is important to know that they tend to overfit the training data and that small perturbations on the input data may lead to a different classification tree.
- **Discriminant analysis:** This technique is fast to predict the classification of the pixels and assumes that the samples are drawn from a Gaussian distribution. However, this assumption does not necessarily resemble the way the samples of the markers are distributed in the RGB space.
- **SVM:** Unlike the other implementations, this classifier is not able to support multiple classes. A possible workaround is to have one instance of the classifier trained for one marker and another for the other one. However, as suggested in the table, the prediction and memory usage is considerably slower than in the previous approaches.

- **KNN:** This technique is maybe the simplest compared to the ones mentioned previously. However, it should not be taken lightly since (i) the model is resilient to small perturbations, (ii) the technique does not assume a specific distribution of the samples, and (iii) it supports the classification of multiple classes with a single classifier.

The previous analysis suggests that the KNN classifier is a suitable option.

## 6.4 Tip detection

As it has been introduced in Section 4.2.2, a threefold method has been considered for detecting the tip of the tool. Since only the general flow of such approach has been explained, its parametrisation and further details about the implementation are given as follows:

- For detecting the borders of the tool, two approaches were initially considered: the Hough transform and an *ad hoc* algorithm. After comparing both methods on several hundreds of frames, the former was discarded; even though parametrising the Hough transform for merging collinear segments and discarding the remaining small lines, better results were achieved with the *ad hoc* approach. However, it must be said that its performance is completely dependent on the segmentation, even though not as much as the Hough transform.
- Considering the *ad hoc* approach implies placing the grid accordingly to the orientation of the tool in the scene. For this purpose, the framework allows to manually set the number of lines in each direction; note that the implementation allows to set a complete grid up, i.e. with horizontal and vertical lines, which leads to a slightly less accurate placement of the lines but working in all cases, indifferently from the orientation of the tools.
- Two algorithms were initially considered for fitting a line on a cloud of points: LMS and RANSAC. During the trials, it was spotted out that RANSAC was only performing better than LMS on those few cases where extreme outliers were obtained by the *ad hoc* approach due to holes or leakages in the computed binary mask. Taking into account this reasoning, keeping a low computational time with the LMS at the cost of having a less accurate estimation of the lines was preferred.

- Once the middle line is computed, an exhaustive search along it is required to find the position of the tip. For this purpose, the framework requires to set a parameter regarding the position of the camera; during the trials, it was noticed that a top view of the scene implies a specific direction of search, while a side view needs the opposite direction of search. Thus, this parametrisation was kept user-friendly.

## 6.5 Kalman filter parametrisation

The general formulation of the Kalman filter has been introduced and extended to our specific problem in Section 4.3.2. Thus, only the parametrisation of the proposed models is missing for giving the full details of the implementation of such filter. In short, those parameters are the noise in the motion and measurement models, and were respectively set to  $\sigma_w = 0.30$  and  $\sigma_v = 0.45$ . The characterisation of those parameters was done empirically.

Apart from those parameters, the state vector and covariance have to be initialised to start the iterative data filtering. Thus, it was considered appropriate to set the initial position of the tool at the centre of the scenario with zero velocity and an uncertainty covering half of the scenario. In that way, a fast convergence from the initial state to the state vectors proposed by the models is granted. Moreover, the time step between frames was fixed to be of 40ms, i.e. the inverse of the FPS rate.

After all this parametrisation, the filtered data was not noisy but it presented some discontinuities. This was the result of having extreme outliers on the 3D estimation of the tool or, in other words, that the assumption of the noise in the measurement model is Gaussian might not be valid. In order to overcome this problem, the 3D data was bounded within the boundaries of the laparoscopic training centre. With this last step, the appearance of the filtered data was satisfactory.

## 6.6 Speeding up the framework

As we detailed in the previous section, the classification of each pixel using the learning based techniques can be an expensive task. Thus, we explored two approaches for speeding up the segmentation process: multi-scale segmentation and RGB classification pre-computation. The two techniques are described in the following sections.

### 6.6.1 Multi-scale segmentation

Multi-scale segmentation addresses the segmentation from two perspectives: location and refinement. Initially, the image is shrunk to a specific proportion of the initial dimensions. At this scale, we perform a rough classification so that the location of areas of interest in which the markers are found, as presented in Fig. 6.3a. This knowledge of location can be used for reducing the search space and, hence, the computation itself. Then, since the transformation to the shrunk image is known beforehand, the correspondence between the area in that image and the one in the original-size image can be obtained. Finally, the classification is performed once again in these areas obtaining a refined segmentation, as presented in Fig. 6.3b. Note that small or sparse regions may be discarded before carrying out the last step.

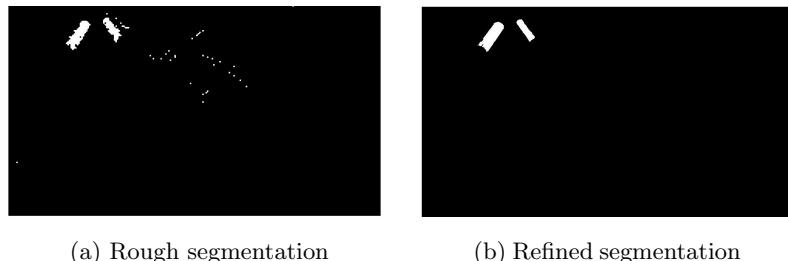


Figure 6.3: Results at two different scales using the multi-scale segmentation approximation.

### 6.6.2 RGB classification pre-computation

As we explained before, the RGB colour of each pixel is used for determining whether the pixel belongs to a marker or not. Since this operation is performed for every single pixel in the image and the image contains around 2,073,600 pixels, the classification becomes considerably expensive (e.g. if the learning technique is a KNN method and its implementation uses a K-D tree, the segmentation complexity is of the order of  $\mathcal{O}(n \log(m))$ , being  $n$  the number of pixels in the image and  $m$  the number of samples in the classifier). However, the classification is not being dynamically updated while processing the different frames and, hence, one could think of pre-calculating the values for each possible combination of RGB vectors and storing them in a lookup table. Therefore, the complexity of the process is reduced to  $m$  accesses to positions in memory which cost is assumed to be  $\mathcal{O}(1)$ , i.e. an overall complexity of  $\mathcal{O}(n)$ . That means that the computation required by the first framework and the second is approximately the same.

# Chapter 7

## Results and evaluation

The proposed framework has been tested to determine the validity of its results and to show its flaws. For this purpose, experiments with real data have been carried out in Section 7.1 to evaluate the implemented part of the framework, i.e. camera calibration, tool detection and tool pose estimation. Then, the theoretical proposal for skill assessment has been tested in Section 7.2 by simulating some data.

### 7.1 Implemented part of the framework

In order to analyse the data obtained with the implemented part of the framework, not only its final results were considered but also the outcome of its different components. The partial, general results and experiments regarding the repeatability of the process are presented and discussed in Section 7.1.1 and Section 7.1.2, and Section 7.1.3, respectively. Finally, the computational cost of the framework is analysed in Section 7.1.4.

Apart from the results reported in this section and due to the nature of the work, supplementary videos are provided along with this document.

#### 7.1.1 Single component evaluation

The aim of this section is to analyse the behaviour of the key parts of the implemented framework, i.e. camera calibration, tool detection and tool pose estimation. However, due to the

impossibility of determining accurately the camera extrinsic parameters, the evaluation of the tool pose estimation part is done in Section 7.1.2.

As stated in Section 2, a more realistic representation of the scene was required to locate the tools in the image. This was successfully achieved using the camera calibration procedure. This is observed in Fig. 7.1 in which from a different point of views curved lines are transformed into straight ones. Note that removing the distortion of the camera leads to a slight reduction of information of the scene, which might be taken into account when setting the scene up.

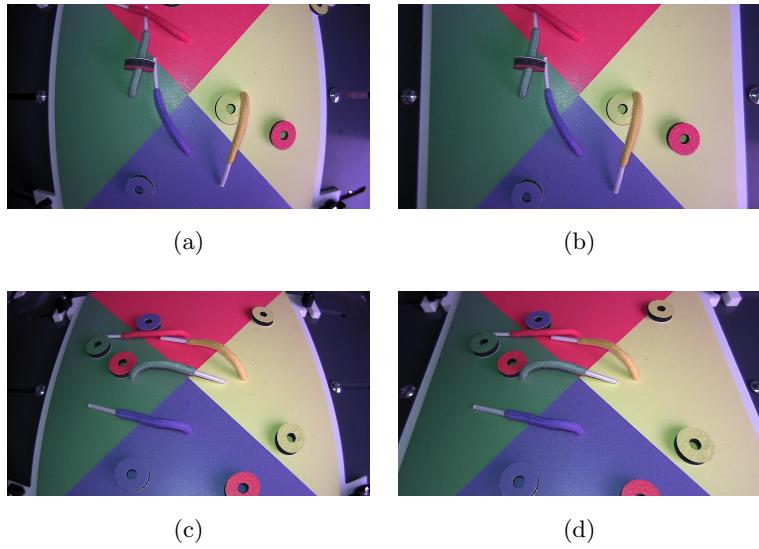


Figure 7.1: Benefits of camera calibration. (a) and (c) are two input frames, (b) and (d) are their corrected version, respectively.

The tool detection is mainly composed of two parts: segmentation and tip detection. Due to the importance of both parts in the detection of the tool, they have been analysed in Tab. 7.1 for different scene conditions, i.e. point of view, lighting conditions and number of tools. These results depict how the framework might manage real-life scenarios.

Scene 1 and 2 in Tab. 7.1 show the obtained results when only having one laparoscopic tool in the scenario, but with different points of view and lighting conditions. In both cases, the binary masks properly represent the area of the marker and, thus, the location of the tip as well as the corresponding width measurement are precisely computed.

The other scenarios in Tab. 7.1 deal with two laparoscopic tools in the laparoscopic training centre. Scene 3 also reports a good tip localisation and characterisation, even though there are some outliers in the segmentation; because of either its size or its shape they are not considered

|         | Undistorted | Binary mask | Tool detection |
|---------|-------------|-------------|----------------|
| Scene 1 |             |             |                |
| Scene 2 |             |             |                |
| Scene 3 |             |             |                |
| Scene 4 |             |             |                |
| Scene 5 |             |             |                |

Table 7.1: Binary mask and tool detection obtained from scenes with different points of view, lighting conditions and number of tools.

as ROIs. Finally, scene 4 and 5 report some flaws of the framework during both the segmentation and location of the tip. Regarding the segmentation, it is really sensitive at the manufacture of the makers; faded regions and the crease on one of the markers make the segmentation fail.

From the previously observations it can be stated that small errors in the segmentation already affect the location of the tip, specifically the placement of the lines on the borders of the tool. Despite these issues, the obtained results are still acceptable and may be properly handled by the Kalman filter.

### 7.1.2 General evaluation

Once the different components have been evaluated separately, the next step is to assess the general output of the framework. To do so, we designed two different scenarios in which the idea consisted in following a specific pattern with the tools so we could check the estimated trajectory. The results of the two trials are described in the following sections.

#### 7.1.2.1 Trial 1

The first trial contemplated the configuration illustrated in Fig. 7.2. In this case, a single tool was displaced along the metallic m-shaped tube on the upper part of the image.



Figure 7.2: Training environment in trial 1.

Having in mind that (i) the x-axis increases from left to right and zero is located right in the middle of the image plane, (ii) the y-axis increases from top to bottom and, as in the previous case, zero corresponds to the middle point of the image plane along this axis, and (iii) the z-axis decreases when the tool approaches the camera being zero at the camera itself; the expectations on XYZ for this particular test are detailed as follows:

- The values for X should go from a negative value to a positive value during all the experiment
- The variation in terms of Y should be small since the tool does not move in this direction. However, note that the white scenario is slightly rotated with respect to the big scenario and, thus, we should expect errors coming from this situation.
- The values of Z should reach their highest values at the beginning and ending parts of the trajectory and the smallest ones when the tool is exactly on top of one of the peaks of the shape

The resulting trajectory in each of the axes throughout the time is presented in Fig. 7.3. It can be observed that X and Y agree with the described expectations, but considerable variations in Z are observed. In this case, we expected the three peaks to be aligned, but they are seen at different distances to the camera. In this case, we expected the three peaks to be aligned, but they are seen at different distances to the camera. The same behaviour is noted for the starting and ending points. This issue could be a consequence of the orientation of the camera in the training centre. We noticed that the camera could be tilted and, hence, one of the sides of the scenario may appear closer than the other and due to the approximations done in the 3D pose estimation step, small variations in the measurements in the image plane lead to large variations in the estimation of the pose in the real world.

One workaround to this problem would be to transform the computed trajectory to a corrected one by including prior information of the test (as the expectations we mentioned). Nevertheless, if all the trajectories are computed with the camera having the same orientation, there should be no major issue when assessing the subjects since they all will present the same “distortion”. Moreover, none of the features is affected by this kind of transformation.

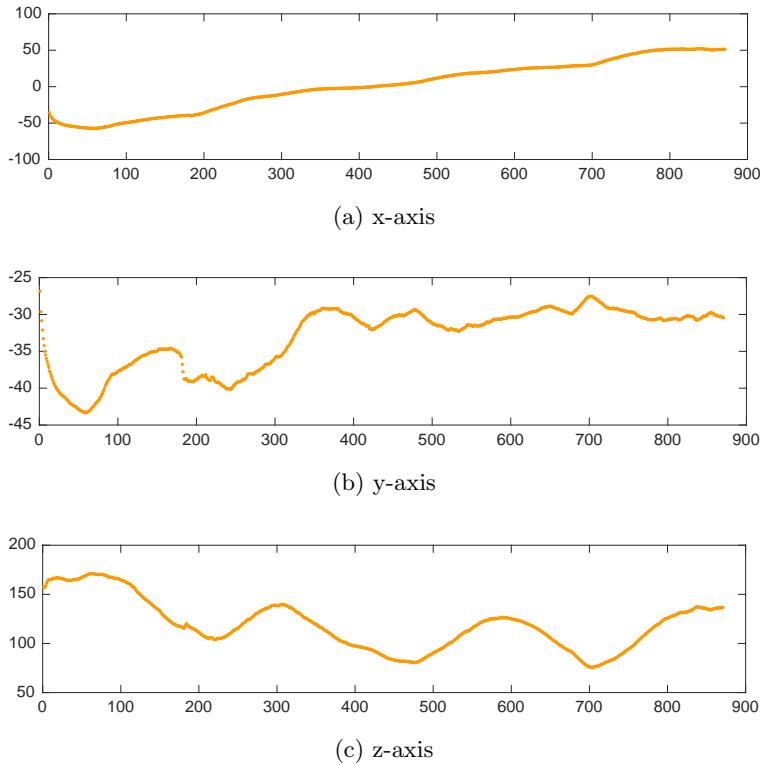


Figure 7.3: Estimated XYZ position of the tool along the video sequence for trial 1.

The resulting 3D path is displayed in Fig. 7.4. It can be observed that although it presented the issues regarding Z, the expected m-shaped figure is traced. It is also important to note that the

movement of the tool itself did not perfectly match the shape of the wire since it was operated by unskilled subjects.

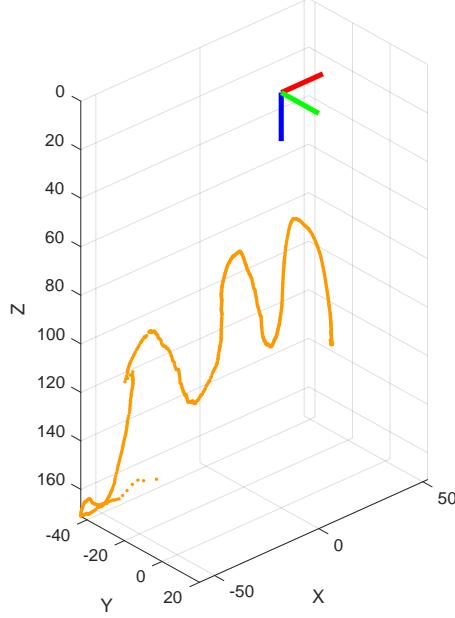


Figure 7.4: Computed 3D path for the m-shaped figure.

### 7.1.2.2 Trial 2

For the second trial, our idea consisted in evaluating the framework using the colourful scenario while (i) drawing a rectangular-like form with two tools, as depicted in Fig. 7.5, and (ii) once that shape is completed, the two tools are moved to the centre of the training scenario and, after, approach the camera. For this experiment, the tools were separated a certain distance which was more or less kept along the trajectory. Note that since one of the tools is violating the discussed approach for selecting the marker colour, its trajectory is expected to be affected by false alarms.

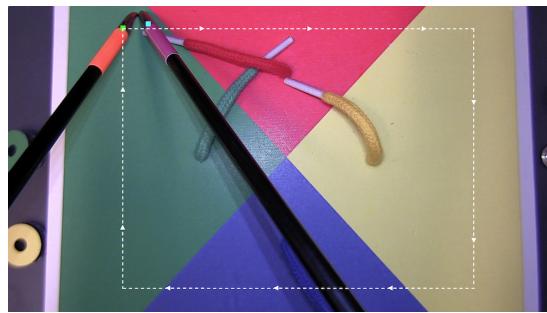


Figure 7.5: Training environment in trial 2. In this experiment, the subject was asked to perform a rectangular-like form as described by the dashed white lines.

The expectations for XYZ along the second trial are detailed as follows:

- For the rectangular-like form, the tools should go through four behaviours:

**On the first side** (i.e. the upper part), the values for X are expected to go from negative to positive, while Y is kept constant

**On the second side** (i.e. the rightmost part), the values for X should be constant, while Y increases

**On the third side** (i.e. the lower part), the values for X should go from positive to negative, while Y is constant

**On the fourth side** (i.e. the leftmost part), the values for X should be constant, while Y goes from positive to negative values

No large variations should be observed in the z-axis while describing the rectangle since the movement is not drastic in this direction.

- Since in the second part the tool gets closer and closer to the camera, Z is expected to decrease, while X and Y are kept mostly constant.

The results along the three axes as the frames are processed are shown in Fig. 7.6. It can be observed that the orange tool agrees with the expectations on the experiment, while the purple marker exhibits some issues in the estimation of Z. As we mentioned before, the purple marker does not comply with the rules for selecting the markers and, hence, when it is displaced on top the red and blue triangles, the segmentation starts to fail. Also, note that the situation is aggravated by the fact that the middle area of the scenario displays a purple-like colour. Nonetheless, the two tools are observed to move together along the x-axis and y-axis which satisfies the expectations of the trial.

The 3D path described in this second trial is presented from a top view in Fig. 7.8 and side view in Fig. 7.7. It can be seen that from a top view the trajectories are similar to a rectangular shape. What is definitely not correct about the trajectories is that the purple tool describes a form which holds the trajectory of the orange marker inside since former is always located to the right of the latter. Also, note that although the two markers are not expected to trace exactly the same form, the difference should not be that high, for instance, in the Z axis in which there is no much movement.

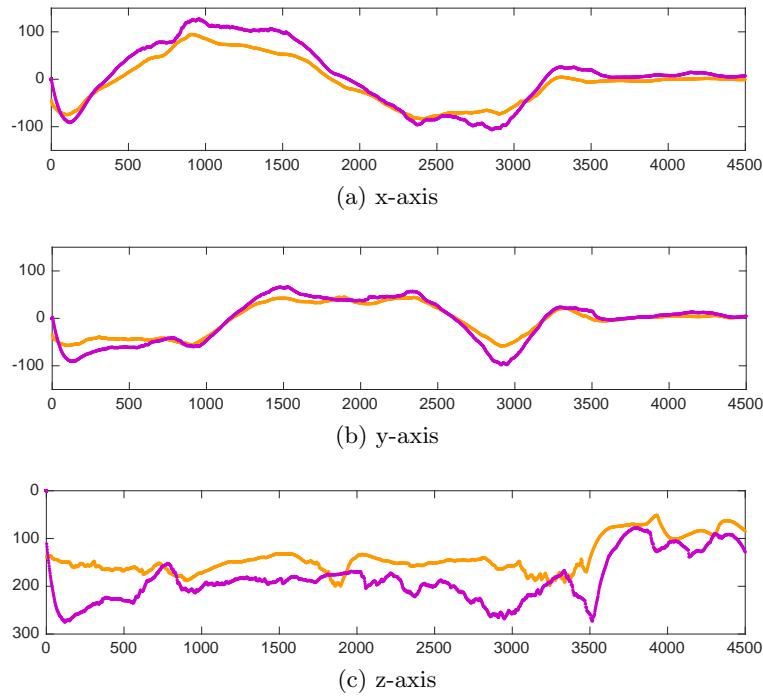


Figure 7.6: Estimated XYZ position of the two tools for trial 2 as the frames were read. Note that the orange and purple lines correspond to the trajectory of the marker with the same colour.

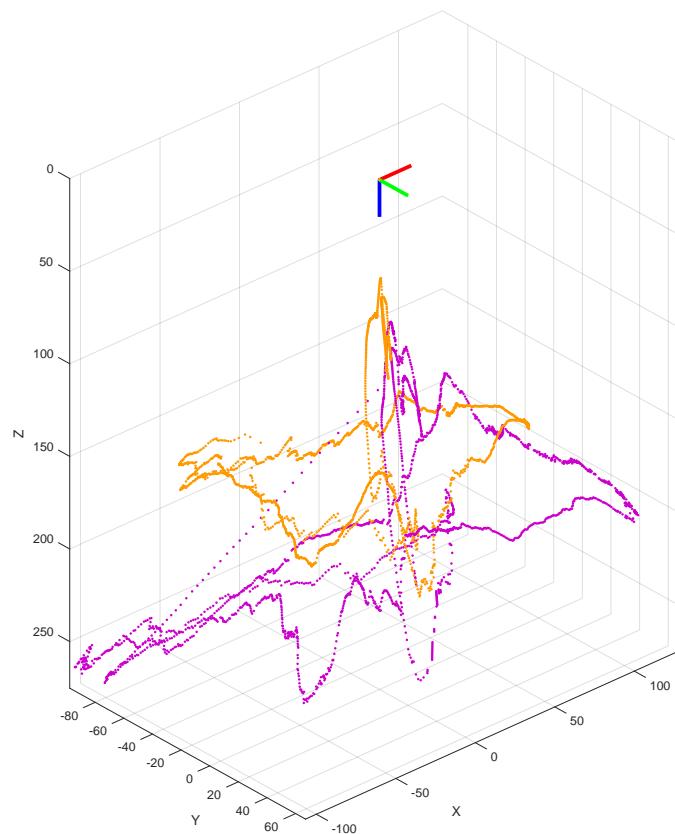


Figure 7.7: Side view of the path described in trial 2.

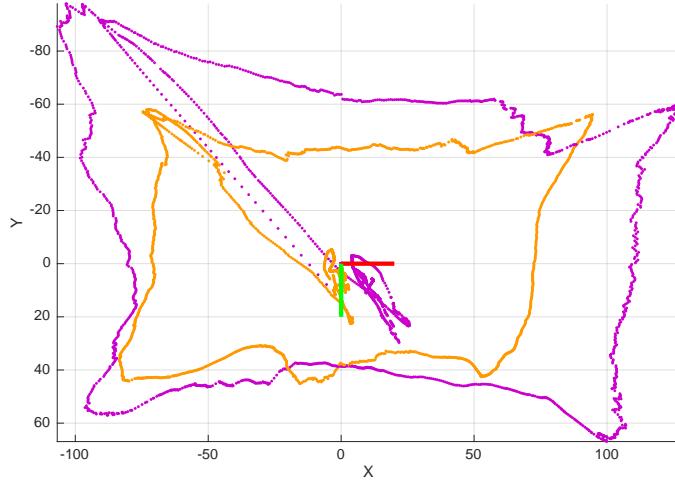


Figure 7.8: Top view of the path described in trial 2.

### 7.1.3 Quantitative evaluation

Performing a quantitative evaluation of the accuracy of the obtained measurements is difficult due to (i) inherent error induced by humans and (ii) the absence of the camera extrinsic parameters. Moreover, since the aim of this framework is to assess the skill of a subject in comparison with some previously stored information, determining the repeatability of the obtained data in different trials of the same movement is of interest.

For this purpose, the yellow guideline shown in Fig. 7.9 was installed in the training centre, allowing the laparoscopic tool to move along the guideline while minimising, although not completely discarding, the human error. Specifically, a movement consisted in sliding the tool along the guideline, from the nearest to the farthest part of the guideline, and back to the initial position. A total of five movements per point of view were performed to evaluate the repeatability of the obtained data. Note that each point of view will be individually analysed.

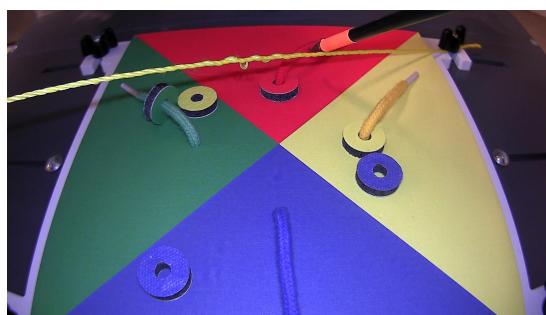
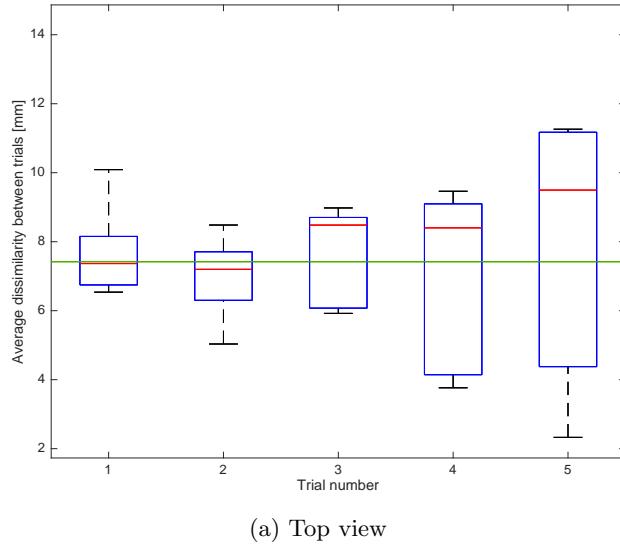


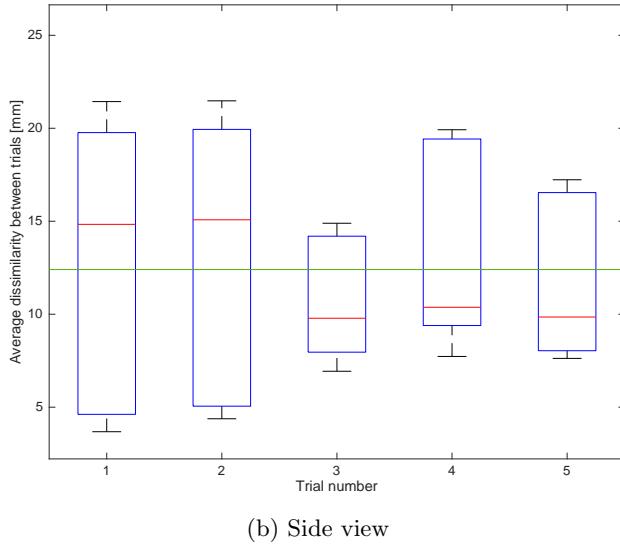
Figure 7.9: Guideline setup for reducing the human error.

For each point of view, the outcome of the previous experiments was five clouds of points, i.e. five paths in the 3D space, which have different numbers of samples. Thus, in order to compare those paths, some post processing was carried out; the 3D line equation was extracted from each cloud of points to determine how good it was at fitting the other clouds of points. The notion of fitness was determined by the average distance in millimetres of each point of the evaluated cloud to the nearest point of the line model.

The obtained 25 fitting coefficients for each point of view were initially placed in a couple of  $5 \times 5$  tables. In order to provide this information in an understandable way, the fitness of each model on the other points of clouds was represented with boxplots. Note that the mean of all this data, which represents the average dissimilarity in millimetres between trials, was plotted in green. Those plots for top and side view are shown in Fig. 7.10.



(a) Top view



(b) Side view

Figure 7.10: Average dissimilarity between trials of the same task.

The average dissimilarity with a top and side view of the scene is of approximately 7 and 12mm, respectively. This discrepancy between trials is due to (a) inaccuracies in the segmentation, (b) inaccuracies in the measurement of the width of the tool and (c) the remaining human error in the experiments. A side view of the scenario tends to be more challenging because of the perspective visualisation of the tools, which not only implies a conical shape but also a wider set of colours for representing the marker.

#### 7.1.4 Performance analysis

As the aforementioned final goal of the framework suggests, the overall process is expected to be carried out in real time. If that feature is achieved, the subjects could be instantaneously evaluated while they perform the training experiments and, thus, the mistakes in which the subject is incurring can be corrected in site. In this sort of ideas, we carried out a performance analysis by averaging the processing time throughout several video sequences. We found that processing a frame takes around 460ms and the contribution of each step is depicted in the pie chart in Fig. 7.11. Two high contributors to the processing rate are pointed out by this plot:

- **Edge detection** presents the highest contribution with a 45%. One workaround to this issue could be to avoid using edge detection for obtaining the boundary of the marker and considering only the outer points intersecting the grid placed in the tip localisation step.
- **Reading from and storing in hard disk** contributes in the 30%. This situation is expected since data transference between RAM and hard disk is computationally costly.

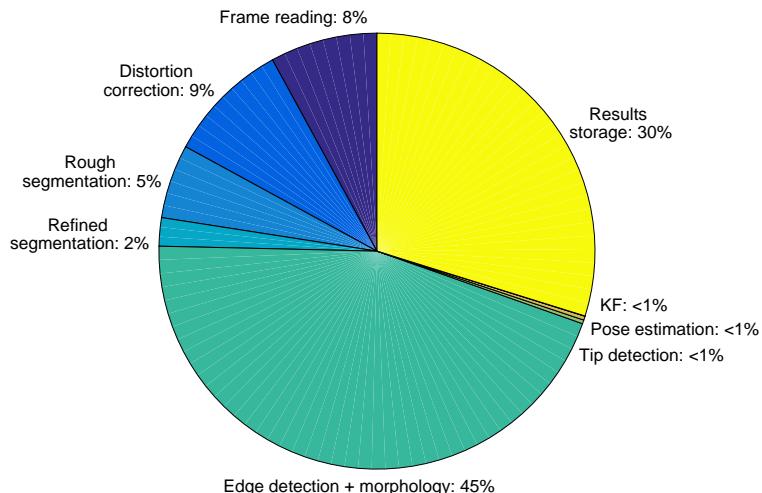


Figure 7.11: Contribution of the different parts of the framework to the overall processing time.

The performance analysis above has been carried out on an Intel Core i7 CPU (2.5 GHz) under OS X El Capitan with 16 GB of RAM. Note that the implementation in MATLAB might use more than one core in some parts of the framework.

## 7.2 Theoretical part of the framework

Due to the problems described in Section 4.3, i.e. the lack of test data, the skill assessment classifier could not be implemented. Being that this was the motivation of the tracking the laparoscopic tools, simulations were done as a proof of concept of the extraction of features from the filtered path. Without loss of generality, the test paths were created in two dimensions.

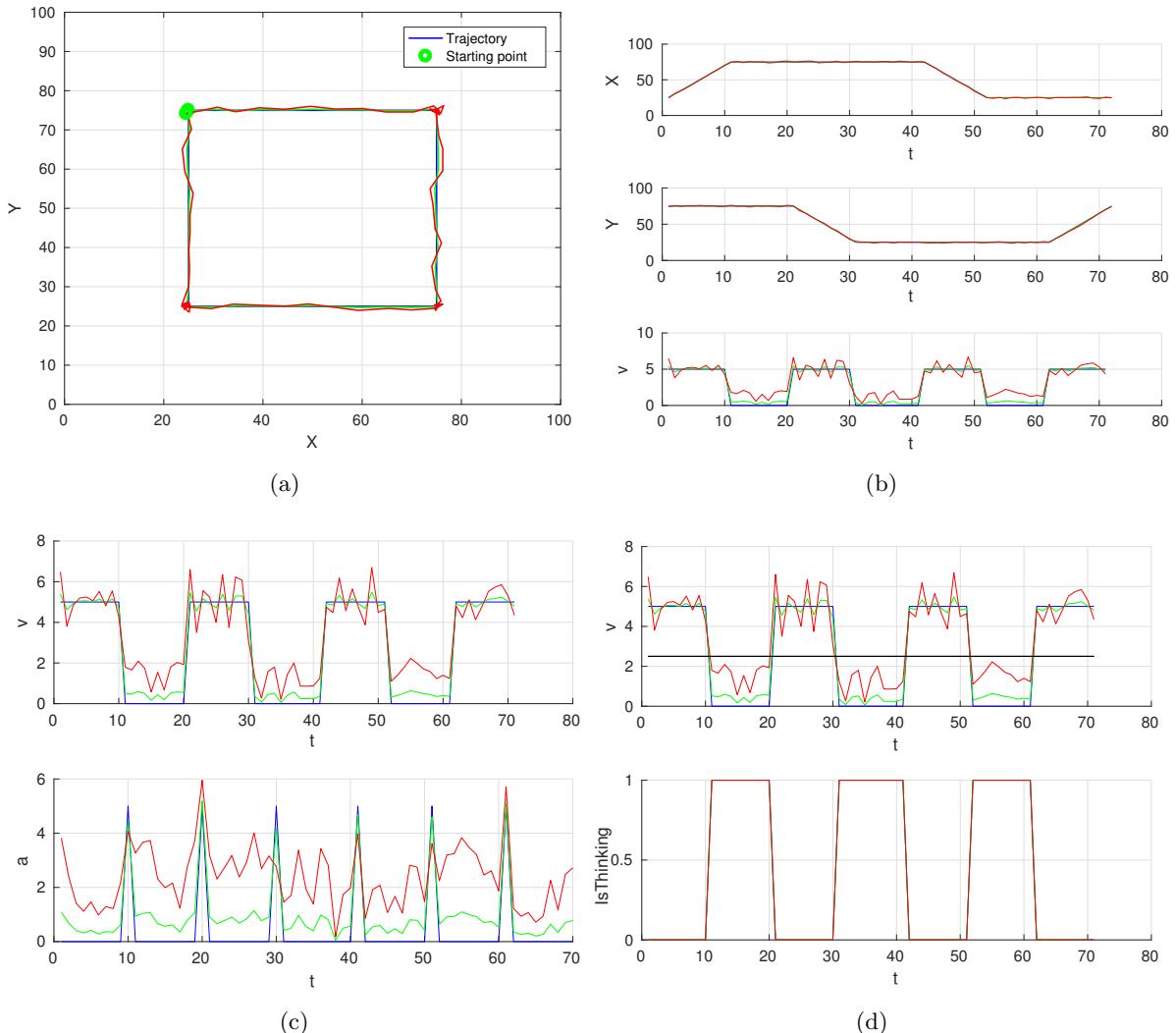


Figure 7.12: Simulated paths and some of the extracted features. (a) Simulated paths with different amounts of noise, (b) coordinates and velocity over time for each path, (c) velocity and acceleration over time for each path, (d) thinking time extracted from velocity magnitude.

The simulated path consisted of the tool following the four sides of a square and taking a pause at each corner. Noise was then added to this path to simulate different levels of expertise of the operator. Three levels of noise were added: no noise, small noise, and large noise. Each trial is represented by a different colour in Fig. 7.12:

- **Large noise (beginner subject):** Red
- **Small noise (trained subject):** Green
- **No noise (expert subject):** Blue

For these simulated cases, the total duration of the activity and the thinking time were fixed for all three paths, and therefore we should expect equal values for these extracted features. The simulated paths served mainly the purpose of example paths for feature extraction and should not be seen as accurate representations of skilled or unskilled operators but as a simplification of what we expect to see.

From the  $(x(t), y(t))$  coordinates at each time-step we calculated five features for each path: mean velocity, jitter, thinking time, duration, and smoothness. The results obtained for each feature of each path can be seen in Table 7.2. The equations that were used to calculate the features in Table 7.2 have been described in Section 4.4.

| Feature       | Beginner subject | Trained subject | Expert subject |
|---------------|------------------|-----------------|----------------|
| Mean velocity | 3.4504           | 2.9867          | 2.8169         |
| Jitter        | 2.4107           | 0.9854          | 0.4286         |
| Thinking time | 0.4366           | 0.4366          | 0.4366         |
| Duration      | 72.0000          | 72.0000         | 72.0000        |
| Smoothness    | 0.0246           | 0.0035          | 0.0000         |

Table 7.2: Extracted features from the simulated trajectories. Fixing the thinking time and the duration of the simulated data, the effect of the mean velocity, the jitter and the smoothness can be analysed.

In these simulated results we can see that the task duration and thinking time do not change between paths, as was expected since the difference between these paths is the amount of added noise. It can be observed that noisier paths lead to larger calculated values of jitter, smoothness and mean velocity. Also, it can be seen in Fig. 7.12a and Fig. 7.12c that while small amounts

of noise are barely noticeable by observing the paths themselves, but they become clear when analysing the velocity and the acceleration at each time-step.

These results indicate that these features are a good simple set of features that can be quickly calculated from the filtered path and these numbers can be used by the chosen classification algorithm to assess the skill level of operators that have their activities recorded.

# Chapter 8

## Final remarks

In this project, we propose a framework for tracking MIS tools on laparoscopy training environments and assessing the level of skill of subjects based on the executed paths.

From a general view, the framework consists of four main parts. Given a video sequence, (i) the frames are corrected to discard distortions coming from the acquisition system, (ii) the tools are segmented using colour segmentation, the width of the same is measured, (iii) based on this information the corresponding pose in the real world of the tools is estimated and, finally, (iv) once all the frames are processed, features are extracted and compared to data from skilled and unskilled operators. Due to the unavailability of data suiting the requirements of the final framework, the final part was addressed theoretically.

The implemented part of the framework was evaluated from three different aspects. Initially, the assessment considered each component separately so that weaknesses, as well as strengths of them, could be highlighted. At this scale, we observed that the segmentation results are highly dependent on the discriminating power of the features used for differentiating the tools from the background. In the case that colour is the considered feature, we noted that maximising the distance from the colours in the scenario and the marker is a good option. Secondly, the components of the framework were evaluated altogether in two different scenarios in which the subject was asked to trace specific patterns using one or two tools. With these two experiments, we corroborated that the traced figures and the estimated trajectories looked alike. Nevertheless, we were also able to spot that the pose estimation method is highly sensitive to the orientation of the camera and, hence, an additional transformation of the resulting trajectory may be required. Finally, the third test aimed to determine whether the approach was able to produce similar

results given similar inputs. The outcome regarding this evaluation element suggested that if an experiment is carried out several times, the framework is able to describe similar paths with an average error of 12mm.

For checking the theoretical part of the framework, i.e. the classification of the trainee based on features extracted from the executed path, synthetic data was generated. The obtained results suggest that the considered features may be able to discriminate unskilled from skilled subjects. However, additional testing on real data is required.

Due to the way the framework is implemented, it can be extended to support additional functionalities and, also, scaled to bigger applications without requiring considerable changes.

As future work, the authors would like to acknowledge that the framework should take advantage of the prior knowledge of the geometry of the tool. Hence, using a Computer-Aided Design (CAD) of the tool and shape-matching techniques may be useful for overcoming some of the detailed problems with the current framework. Also, since shape-based matching may require performing registration operations, the knowledge coming from the current approach could be used for initialising the matching algorithm and, hence, for decreasing the overall computational time.

# Bibliography

- Bouguet, J. Y. (2010). Camera calibration toolbox for matlab. Technical report, Computational Vision at the California Institute of Technology.
- Cuevas, E., Zaldivar, D., and Rojas, R. (2005). *Kalman Filter for Vision Tracking*. Freie Universität Berlin, Fachbereich Mathematik und Informatik / B: Fachbereich Mathematik und Informatik. Freie Univ., Fachbereich Mathematik und Informatik.
- Designs, K.-C. (2008). Rgb tint colour wheel. [http://www.major-confusion.co.uk/tutorials/webplus/rgb\\_tint\\_colwheel.html](http://www.major-confusion.co.uk/tutorials/webplus/rgb_tint_colwheel.html). Online; [accessed 16 Dec 2016].
- Forsman, M. (2011). Point cloud densification. Technical report, UMEA University.
- Friedman, N. and Russell, S. (1997). Image segmentation in video sequences: A probabilistic approach. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence, UAI'97*, pages 175–181, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.
- Gonzalez, R. C. and Woods, R. E. (2006). *Digital Image Processing (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.
- Heikkila, J. and Silven, O. (1997). A four-step camera calibration procedure with implicit image correction. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 1106–, Washington, DC, USA. IEEE Computer Society.
- Hulke, U. and Gupta, A. (2014). Single camera based motion tracking for minimally invasive surgery. In *22nd Mediterranean Conference of Control and Automation (MED)*, pages 356–361. IEEE.
- Koller, D., Weber, J., Huang, T., Malik, J., Ogasawara, G., Rao, B., and Russell, S. (1994). Towards robust automatic traffic scene analysis in real-time. In *Decision and Control, 1994., Proceedings of the 33rd IEEE Conference on*, volume 4, pages 3776–3781 vol.4.

- MathWorks (2016a). Supervised learning workflow and algorithms. <https://www.mathworks.com/help/stats/supervised-learning-machine-learning-workflow-and-algorithms.html>. Online; [accessed 16 Dec 2016].
- MathWorks (2016b). What is camera calibration? <http://uk.mathworks.com/help/vision/ug/camera-calibration.html>. Online; [accessed 10 Dec 2016].
- Moll, M., Sucan, I. A., and Kavraki, L. E. (2014). An extensible benchmarking infrastructure for motion planning algorithms. *arXiv preprint arXiv:1412.6673*.
- Ridder, C., Munkelt, O., and Kirchner, H. (1995). Adaptive background estimation and foreground detection using kalman-filtering. In *Proceedings of International Conference on recent Advances in Mechatronics (ICRAM)*, volume 1.
- Shin, S., Kim, Y., Cho, H., Lee, D., Park, S., Kim, G. J., and Kim, L. (2014). A single camera tracking system for 3d position, grasper angle, and rolling angle of laparoscopic instruments. *International Journal of Precision Engineering and Manufacturing*, 15(10):2155–2160.
- Stauffer, C. and Grimson, W. E. L. (1999). Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE.
- Tonet, O., Thoranaghatte, R. U., Megali, G., and Dario, P. (2007). Tracking endoscopic instruments without a localizer: A shape-analysis-based approach. *Computer Aided Surgery*, 12(1):35–42.
- Yang, J., Shi, M., and Yi, Q. (2012). A new method for motion target detection by background subtraction and update. *Physics Procedia*, 33:1768 – 1775.
- Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(11):1330–1334.
- Zhou, J. and Payandeh, S. (2014). Visual tracking of laparoscopic instruments. *Journal of Automation and Control Engineering Vol*, 2(3).