UNIVERSITÉ DE STRASBOURG

ICube Laboratory (UMR 7357)
Research Group CAMMA on
Computational Analysis and Modeling of Medical Activities

Vision-based Approaches for Surgical Activity Recognition Using Laparoscopic and RBGD Videos

Thesis presented by

Andru Putra Twinanda

on

27 January 2017

Submitted to the

University of Strasbourg

for obtaining the degree of

Doctor of Philosophy

delivered by the doctoral school MSII

Acknowledgment

Is it Andrew with ew or Andru with u? No, it is Andru with Ph.D.
– Satrio Adi Rukmono

First and foremost, I would like to thank The Almighty Allah SWT, for only by His Grace and Blessings, I was able to finish my doctoral studies.

I would like to thank Dr. **Nicolas Padoy**, my thesis supervisor, who has welcomed me 3.5 years ago to do my doctoral studies with him. I am really grateful for his guidance throughout my thesis work and also his brilliance, which can be intimidating yet motivating at the same time. The most important take-away lesson that I have learnt from him is that hard work always pays off. For this lesson, I will be forever indebted to him. This thesis would not be possible without the support of Prof. **Michel de Mathelin**, my thesis director, who has always been nothing but kind and supportive, despite his busy schedule as the chief of the ICube laboratory.

I also would like to thank my thesis committee: Prof. **Jocelyne Troccaz**, Prof. **Gregory Hager**, and Dr. **Gwenolé Quellec** for their constructive comments on my thesis. To hear positive feedbacks from such esteemed members of the community is the best affirmation that I could ever hope for.

I also would like to take this opportunity to thank our medical partners. I would like to thank the surgeons, especially Prof. **Jacques Marescaux** and Prof. **Didier Mutter**, at the University Hospital of Strasbourg who have kindly allowed us to obtain the laparoscopic videos and also guided us during the problem definition stage. I would like to express my gratitute to the audio-visual team, especially **David**, **Thibaut**, and **Melody**, who have been a tremendous help in the process of acquiring the laparoscopic videos. In addition, I also would like to thank the interventional radiology department of the University Hospital of Strasbourg and especially the head of the department, Prof. **Afshin Gangi**, for allowing us to install our recording systems in several operating rooms and to record the videos during real surgeries.

I would like to thank all members of CAMMA research group who have been working closely with me along the course of this thesis. For those countless meetings, invaluable discussions, restless nights, and incredible camaraderie, I will be forever grateful. Special thanks to: Dr. **Rahim** and Dr.-to-be **Nicolas Loy**. Thank you for always being there for me during this amazing 3.5-year roller coaster ride. Thanks to other CAMMA members (former and current): **Laurent**, **Fernando**, **Antonio**, **Ozan**, **Sherif**, **Pramita**, and **Ivan** who have also been important contributors to this thesis.

I would like to also thank my colleagues from the research team AVR for creating a great working environment. Especially, I would like to thank the current habitants of the PhD space: **Carlos**, **Rafael**, **Nicole**, **Ajeethan**, **NLoy**, **Rahim**, **Quentin**, **François**, **KiSoo**, and **Gauthier**; as well as the former habitants: **Laure-Anais**, **Arnaud**, **Nitish**, **Paolo**, **Nadège**, **Cédric**, and **Markus**, for the fun and friendly ambiance in the PhD space.

I would like to thank my high school friends Rexaicel, who are more like a family to me at this point. Special mention to: **Asih** and **Widya**, who have been my trash cans for more times than I would care to admit. I would like to thank all my friends in Strasbourg (PPI Alsace, Paguyuban Bebeb, Bkn.Cintanya.Yg.Terlarang, and Franco-Indo association. You know who you are!) who have kept me sane and have made this journey fun.

Last but not least, I would like to thank my family: my parents, my brother (and his family), and my sister, who always remind me what is important and keep me grounded. They are the only constant and solid support that I can always count on, as there are times that my sole motivation to power through the struggles is just to see them proud.

Okay, I promise this is the last one. For **Bubu-chan**, thank you for helping me in making sense of this journey.

It has been amazing and I could not have done it without all of you!

Abstract

The main objective of this thesis is to address the problem of activity recognition in the operating room (OR). Activity recognition is an essential component in the development of context-aware and data management systems, which will allow various applications, such as automated assistance during difficult procedures and automatic report generation. Here, we focus on vision-based approaches since cameras are a common source of information to observe the OR without disrupting the surgical workflow. Specifically, we propose to use two complementary types of videos: laparoscopic and OR-scene RGBD videos. Laparoscopic videos record in detail the tool-tissue interactions inside the patients during minimally invasive surgeries, while OR-scene RGBD videos are recordings from a multi-view ceiling-mounted camera system which captures the activities occurring in the whole room. Despite the vast literature on activity recognition in computer vision, activity recognition in surgical setups is still far from being solved. The OR is a very particular and challenging environment where the objects are of similar colors and the scene contains a lot of clutter and occlusions. Furthermore, laparoscopic videos capture a scene completely different from the conventional videos used in the computer vision community, which typically contain humans in the scene. The laparoscopic videos also contain inherent visual challenges, such as rapid camera motion and specular reflection.

In this thesis, we investigate how state-of-the-art computer vision approaches perform on these videos and propose novel approaches to overcome some of the aforementioned challenges. First, we establish recognition pipelines to address activity recognition problems on both laparoscopic and OR-scene RGBD videos using the bag-of-word (BOW) approach, Support Vector Machines (SVM), and hidden Markov models (HMM). Second, we propose an extension to the BOW approach used on multi-view RGBD data to retain more spatial and temporal information during the encoding process. Ultimately, to alleviate the difficulties in manually designing the visual features to represent the data, we design deep learning architectures for surgical activity recognition. We also propose an end-to-end approach with an LSTM network to eliminate the need for SVM and HMM. To evaluate our proposed approaches, we generate large datasets of real surgical videos, including either laparoscopic videos or multi-view RGBD videos. The results demonstrate that the proposed approaches outperform the state-of-the-art methods in performing surgical activity recognition on these new datasets.

Table of Contents

List of Figures

List of Tables

1 Introduction

If everyone waited to become an expert before starting, no one would become an expert.
– Richie Norton

Chapter Summary

Surgery is a branch in the field of medicine that deals with diseases which require physical manipulation of a bodily structure to diagnose, prevent, and cure the ailment. Since the industrial revolution, advances in technology have transformed surgery into a scientific discipline capable of treating many diseases. These advances are translated into medical innovations, such as cutting-edge surgical tools, navigation and monitoring systems, and novel imaging technologies. As they improve the patient treatment, these technologies have also altered the operation routines. This can be observed through the differences in how the modern operating rooms (ORs) look compared to the ORs in mid-20th century, shown in Fig. 1.1. In Fig. 1.1-a, a picture taken at the Veterans Administration Hospital in 1953 shows a simple OR which contains only a bed and a few equipment tables. The image in Fig. 1.1-b was taken more than 50 years later at Nouvel Hôpital Civil Strabourg in 2015, in which the hybrid OR is equipped with an

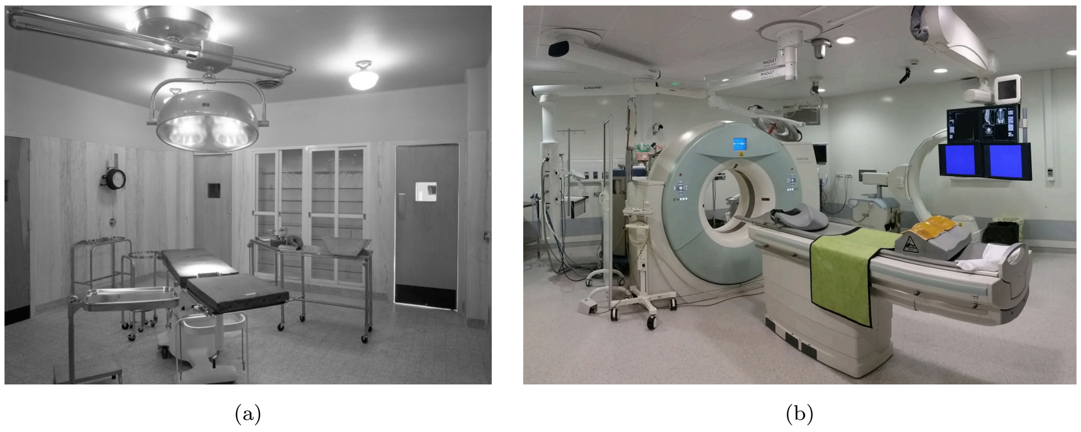


Figure 1.1: Scenes in two operating rooms from different centuries: (a) Veterans Administration Hospital in 1953 and (b) Nouvel Hôpital Civil Strasbourg in 2015.

intraoperative computed tomography system, a mobile C-arm device, and other digital surgical equipment. From these images, one can observe the differences between the ORs from two different ages.

As this transformation has allowed the execution of complex procedures, it has also multiplied the amount of information available in the OR. For example, with the integrated OR display, the surgeon can monitor the patient's vital signs while observing the patient's anatomical structure during a minimally invasive surgery. For certain surgeries, the surgeons also need to simultaneously compare preoperative and intraoperative data. On the one hand, such relevant information is essential for the surgeons and clinicians to execute the complex surgical procedures. On the other hand, information that is not presented at the right place and/or at the right time could also impede the smooth execution of the procedures. In addition, we hypothesize that this rich information available in the OR could be further utilized for various clinical applications. For instance, using the patient's vital signs, one could build a system to predict the requirement of additional anaesthesia for the patient. Therefore, there is a growing interest in the community to construct a context-aware system (CAS) which treats the information available in the OR to provide contextual support to the clinicians [Cleary 2004, Bharathan 2013], ranging from simply displaying the relevant information in the most appropriate manner during surgery to assisting the clinicians in performing difficult surgical tasks by suggesting courses of actions to take. To provide such a support, a CAS carries one main objective which is to understand the physical environment of the OR. In this thesis, we focus on one essential task required to achieve the objective of a CAS, which is to recognize the activities occurring in the OR.

In the following sections, we describe several approaches using various types of signals available in the OR to perform surgical activity recognition. Then, we present clinical applications that can benefit from the automatic surgical activity recognition in the OR. We also present the types of activity recognition tasks that will be addressed in this thesis. Ultimately, we discuss the contributions of this work and conclude by outlining the structure of this thesis.

1.1 Activity Recognition in the Operating Room

In this section, we first discuss the signals available in the OR that have been used for surgical activity recognition. We then focus the discussion on visual information coming from the cameras used during surgical procedures. We describe the advantages as well as the challenges in using visual information to perform the surgical activity recognition task. Finally, we present the types of videos that we use for the task.

1.1.1 Digital Signals for Surgical Activity Recognition

In order to perform surgical activity recognition, one needs to acquire information from the operating room which represents the occurring activities. Thanks to the introduction of digital tools into the OR, various digital signals are available to be utilized. For example, patient's vital signs [Xiao 2005] have been employed to identify the state of the OR in real time (i.e., occupied vs. not occupied). Another type of signals is tool usage information, which has been shown to perform the task very well in several studies in the literature [Padoy 2012, Stauder 2014] thanks to the discriminative information that they contain regarding the surgical activities. However, these signals only provide specific and limited information regarding the procedures. For example, these signals cannot be used to determine activities which involve no digital surgical tools (e.g., patient entering the OR). In addition, most of the time, the tool usage signals are not easy to obtain directly from the OR. They are usually collected through non-trivial processes: either via manual annotation or by using an additional equipment, such as a camera-equipped trocar [Kranzfelder 2009, Toti 2015], to collect the signals.

In a number of studies [Bardram 2004, Meyer 2007, Parlak 2012], radio-frequency identification (RFID) tracking systems were proposed as a solution to track the usage of surgical tools as well as the movements of clinicians. Using a robotic system, such as the da Vinci surgical system, a synchronous stream of data, including the tool usage signals and kinematic data, can be obtained to perform surgical phase segmentation [Malpani 2016]. However, such an advanced robotic system is scarcely available in the ORs across the globe. In another work [Nara 2011], the motion trajectory of the clinicians in the OR obtained from an ultrasonic 3D motion tracking system is used to monitor the surgical activities. Such tracking data provides contextual information related to the activities in the OR. However, the incorporation of such tracking systems in the OR can disrupt the workflow in medical facilities. For example, the ultrasonic motion tracking system requires a high number of ceiling-mounted receivers and obliges all the clinicians to wear a transmitter tag.

In addition to the aforementioned digital tools, the advancement in technology has also introduced the usage of imaging devices to support the surgeons by providing better visualization of the anatomical structure. For example, in cataract surgeries, a microscope is used to provide better visualization of the eye lens. In laparoscopic surgeries, a fiber-optic instrument (i.e., endoscope) is inserted through the abdominal wall to view the organs inside the abdominal cavity. The recordings from the aforementioned imaging devices are particularly interesting to identify surgical activities since they capture in detail the interactions between the surgical tools and the anatomical structures of the patients. Thanks to these properties, numerous studies in the literature have addressed surgical activity recognition using such videos [Blum 2010, Lalys 2012, Quellec 2015].

In addition, there are also other devices installed in the OR which observe the whole room, e.g., ceiling-mounted cameras. These cameras are becoming widely used in healthcare facilities for teaching, legal, safety, and documentation purposes. Contrary to the imaging devices used for laparoscopic and cataract surgeries, these cameras do not capture the tool-tissue interactions, but rather capture generic actions happening in the OR which are typically not apparent in laparoscopic/cataract videos, such as patient entering the OR and surgical tool preparation. In addition, the ceiling-mounted cameras are also useful to record other types of surgeries that do not require particular imaging devices (e.g., open surgeries). Several studies [Bhatia 2007, Chakraborty 2013, Lea 2013] have shown that this type of videos contain discriminative information which can be used for surgical activity recognition.

In this thesis, we focus on using the visual information coming from cameras in the OR to perform surgical activity recognition because of the following reasons. First, cameras are inherent to the execution of surgeries, thus the data acquisition is simpler compared to the acquisition of other signals which typically involves manual annotation and/or additional equipment. Second, the cameras provide rich information which is not always captured by other signals. For example, the recordings of the ceiling-mounted cameras can be used to identify the patient preparation phase, while this might not be captured by patient's vital signs or tool usage signals. Lastly, the videos can be used beyond *intraoperative* surgical activity recognition since they are typically stored in a database after the procedures. Labeled with the automatically recognized activities, these videos could then be exploited for various applications, such as training and postoperative reviews.

1.1.2 Visual Information for Surgical Activity Recognition

In this thesis, we focus on addressing surgical activity recognition on the following video types: laparoscopic videos and OR-scene videos. It is interesting to carry out the task on these videos because of their complementary nature, i.e., the former captures in detail the anatomical structure inside the abdominal cavity of the patient, while the latter captures the general activities occurring in the OR. Ideally, both video types should be used simultaneously to obtain the complete information regarding the activities occurring in the OR. However, in this thesis, the laparoscopic and OR-scene videos could only be recorded in different rooms. Even though such synchronous videos are not currently available, this does not prevent us from carrying out surgical activity recognition independently on each video type.

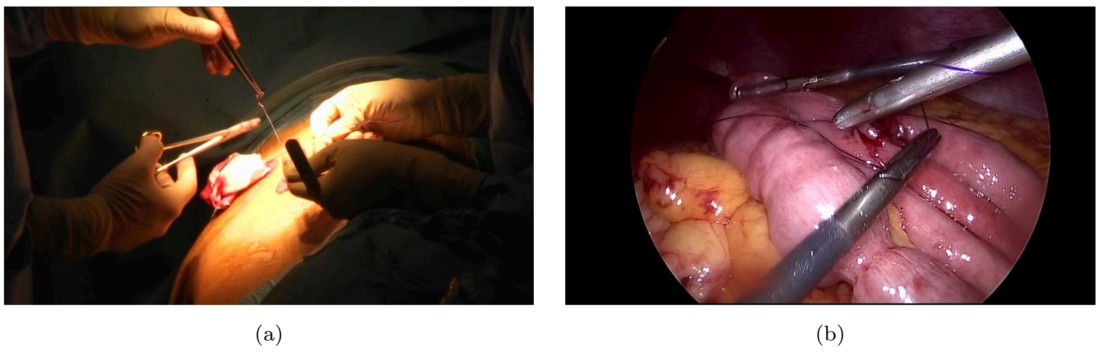


Figure 1.2: Image samples captured during a laparoscopic procedure: (a) image captured by a ceiling-mounted camera showing the abdomen of the patient and (b) laparoscopic image showing the inside of the abdominal cavity of the patient.

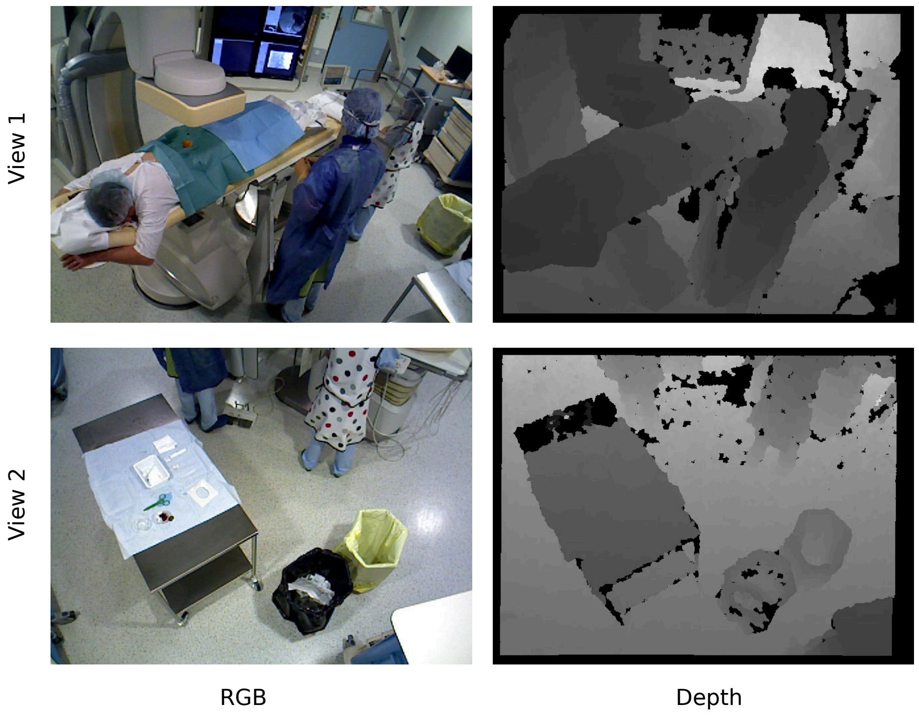


Figure 1.3: Synchronized image samples from the multi-view RGBD camera setup.

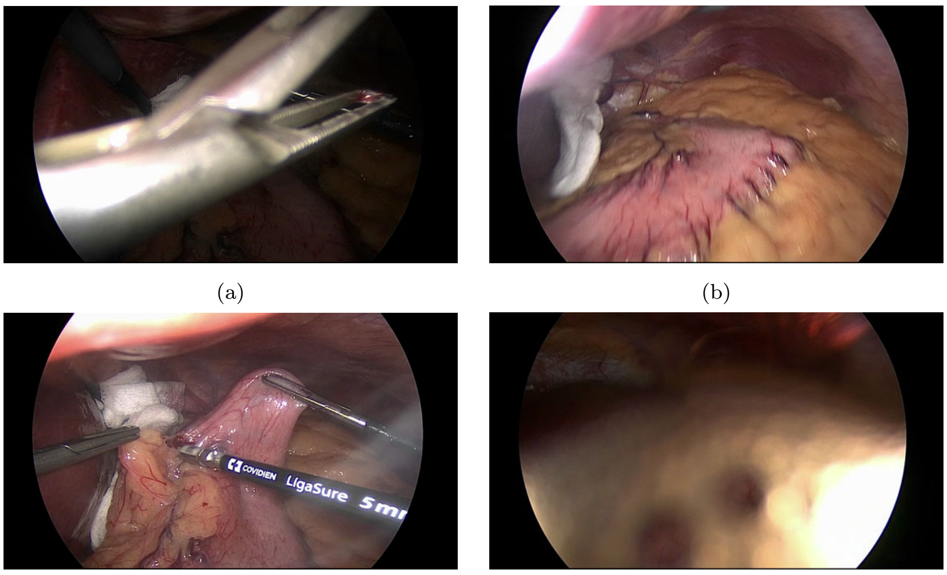


Figure 1.4: Laparoscopic images illustrating several visual challenges: (a) specular reflection, (b) rapid camera motions, (c) the presence of smoke, and (d) the dirtiness of lens.

For the laparoscopic videos, we obtain the recordings from the laparoscopic video database of University Hospital of Strasbourg/IRCAD. In Fig. 1.2, we show image samples captured during a laparoscopic procedure. As for the OR-scene videos, we propose to obtain the recordings using a multi-view RGBD camera system mounted on the ceiling of the OR. RGBD cameras allow to record simultaneously color and depth images of the scene, which provide complementary information: color images capture the visual appearance of objects' surfaces while the depth images capture the distance between an object to the sensor. The depth images provide information which is not captured in the color images, i.e., the 3D structure of the scene. In addition, it is interesting to use such a sensor since the depth sensor works in the infrared light wavelength, thus it is invariant to illumination changes and the lack of textures. In order to cover a larger area in the OR, we also propose to use a multi-view camera system. In Fig. 1.3, we show the image samples captured by our multi-view multi-modal camera system mounted in a hybrid interventional OR. We can see that the usage of a multi-view system allows us to observe two complementary scenes at the same time: the scene around the OR bed and the equipment table.

1.1.3 Challenges for Vision-Based Approaches

Since we are using videos as the source of information for surgical activity recognition, vision-based solutions will be the focus of the following discussions in this thesis. Despite the vast literature on activity recognition in the computer vision community, it is still unclear whether state-of-the-art methods can generalize well to *surgical* activity recognition. This is due to the fact that the OR-scene videos capture a very different scene than the videos typically used in the computer vision literature. Even though the OR-scene videos capture a constrained environment where the activities are occurring mostly around the operating bed and the equipment table, surgical activity recognition on OR-scene videos is not a trivial task due to various challenges. For instance, the performed activities typically consist of fine gestures, such as needle insertion. Furthermore, due to the presence of multiple persons and equipment in the OR, there is a high frequency of occlusions occurring in the videos, which will impede the effectiveness of activity recognition methods. In addition, there is also a high similarity in appearance due to the similar color and textureless nature of various object surfaces in the OR scene.

In the computer vision community, the proposed approaches typically involves identifying actions from videos where humans are observed in the scene. In contrast, the laparoscopic cameras capture a completely different scene from the traditional cameras. The image frames from laparoscopic videos are typically similar to each other, resulting in low inter-class variability. Moreover, visual challenges, such as rapid camera motions, specular reflection, the presence of smoke, and occluded lens, are also inherently present in the laparoscopic videos, as shown in Fig. 1.4.

The aforementioned visual challenges in both video types make it difficult to establish robust methods to perform the surgical activity recognition tasks. Moreover, due to these challenges, it is also difficult to design discriminative visual features to represent the data.

Furthermore, the datasets generated in the course of this thesis contain videos of long durations, totalling up to hundreds of hours. In addition to the efforts required to manage and perform computation on such a large amount of data, it is also not trivial to carry out activity recognition on such videos due to the fact that the activities have a high duration variability. To address this variability, we assume that the activities follow an underlying *surgical workflow*, as it is generally the case in common surgeries. Surgical workflow is a set of rules that governs the activities in a procedure, starting from the entrance of the patient up to their exit. Thus, the activities performed in the OR respect certain rules which impose temporal constraints in their executions.

1.2 Medical Applications

The capability to automatically recognize the surgical activities in the OR plays an important role in the development of a CAS in the OR. Specifically, such a system would open up the possibility for many applications, both intraoperative and postoperative, as discussed in the following sections.

1.2.1 Intraoperative Applications

To assist the surgeons and clinical staff in achieving the objectives of surgeries, it is imperative to **display the information in the most suitable manner and at the right time**. The intraoperative recognition of surgical activities in the OR can be used to determine the information required by the clinicians at a given time during a surgical procedure. For example, during a certain phase of a cholecystectomy (gallbladder removal) procedure, the system could help the surgeon in identifying the calot triangle by overlaying a virtual calot triangle onto the laparoscopic videos. Furthermore, it could also be used to build a **real-time notification system** that sends notifications to senior surgeons when certain key surgical activities are being executed in the OR. These key surgical activities could be defined as surgical activities that are prone to complications, such as *clipping and cutting* in cholecystectomy and *cement injection* in vertebroplasty. This real-time notification system could also be used to **improve OR resource management**. By knowing which surgical phase is currently occurring in the OR, the completion time of the surgery could then be estimated. This real-time estimation could be used to notify the clinical staff to prep the following patient, resulting in minimal waiting time for the patient. It is especially necessary to maximize the OR throughput when hundreds of patients and staff may be flowing through dozens of operating rooms on a daily basis in a single facility [Sutherland 2006].

In addition, by knowing the activities occurring in the OR, the system would also be able to **identify and provide assistance** to the clinicians. For example, through deeper analysis of the surgical workflow, the system would be able to detect rare cases during the execution of surgery by recognizing unexpected variations in the surgical activities. This could be done by comparing the current workflow with the reference surgical workflows. By going through the large video database, the system would also be able to suggest the best actions to handle these rare cases.

1.2.2 Postoperative Applications

Automatic recognition of surgical activities is not only useful during the procedure, but also afterwards when the procedures are finished. For example, key events and their time of occurrence could be identified automatically, **facilitating the writing of the surgical reports**.

The recordings of the procedures are usually stored in a video database for documentation. These videos contain rich information regarding the execution of various surgeries for **training purposes**. For example, let us consider a young surgeon who is interested in learning how *gallbladder dissection* is performed in cholecystectomy procedures. Currently, the only way to access this information is by manually browsing through the database to find the cholecystectomy videos, followed by tediously scrolling through the video to find the execution of *gallbladder dissection*. All this process could be easily reduced to a simple search query if an indexed database, in which the videos are tagged with their key events, is available. Currently, the video labeling process is performed through a laborious process of manual annotation. By being able to automatically identify the key events in a surgery, the manual annotation process would be alleviated and a much larger annotated video database could be generated. In addition to the aforementioned training purposes, this effortless navigation to browse the videos in the database would also be advantageous for **postoperative review** as well as **surgical skill assessment by senior surgeons**.

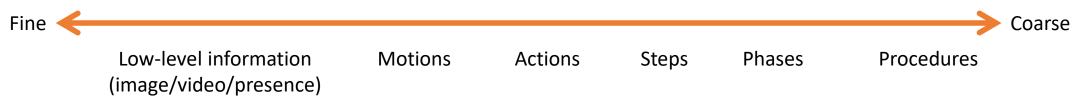


Figure 1.5: Axis showing the granularity of activity in the OR [Lalys 2014] from the finest level (left) to the coarsest level (right). Originally in the granularity axis, the term *"Actions"* is written as *"Activities"*. We replace the term in order to avoid confusions.

By having a database that is fully annotated with the surgical activities, one could also easily generate statistical information regarding these activities. This would be particularly useful for **surgical workflow optimization**. Combined with other signals from the OR, this information could be used by experts to analyze hundreds of surgeries and subsequently improve the existing surgical workflows, when needed.

1.3 Terminology in Surgical Activity Recognition

The definition of activity is a subjective matter and highly depends on the level of abstraction at which the activity is described, i.e., the granularity of the activity. One might define a simple action (e.g., waving hand) as an activity, while another person might assume a more complex action (e.g., preparing breakfast) as an activity. This problem also translates to activities in the OR. Depending on the granularity of the activity, various terms have been used to refer to the addressed activities, such as *surgeme* [Lin 2006], *phase* [Blum 2010, Padoy 2012], and *gesture* [Zappella 2013, Tao 2013].

In order to explicitly clarify these definitions, we adopt the notion of activity granularity presented in [Lalys 2014]. The granularity axis is illustrated in Fig. 1.5, where the coarsest is at the right-most of the axis and the finest is at the left-most of the axis. At the coarsest level, a surgical activity is defined as the surgical procedure itself. Each surgical procedure can be broken down into a list of phases, which define major events occurring during surgery. A step consists of a sequence of physical actions used to achieve a surgical objective. Each action is composed of motion trajectories which do not contain any semantic information. At the finest granularity, low-level information, such as the presence/absence of an object/person, is defined.

In addition to the granularity level of the activity, the surgical activity recognition task can be divided into two categories depending on how the recognition is performed. The first category assumes the task of labeling videos with their corresponding activity class labels (e.g., [Zappella 2013]). The videos could be either video clips extracted from segmented videos or the complete videos themselves. We refer to this task as *pre-segmented classification*. In the second category, the task consists of identifying a sequence of activities performed in a video without any information regarding the beginning/ending of each activity (e.g., [Padoy 2012, Quellec 2015]). To address problems in the second case, we perform the task in a frame-wise manner, where each frame is classified as a certain activity, hence the name *frame-wise classification*. In the literature of surgical activity recognition, the frame-wise classification task is more frequently addressed than the pre-segmented one. This is due to the fact that it corresponds to the real world applications where the videos are not pre-segmented. Typically, frame-wise classification requires a temporal recognition model to enforce the temporal constraints of the surgical workflow.

In this thesis, we model the surgical workflow using the approach proposed in [Padoy 2007b], where the models are constructed using signals obtained from the OR. This means that the workflow is observed by a time-series of multi-dimensional feature vectors obtained from the signals (in our case, the video images). A multidimensional vector of signals at a certain time-step can be referred to as an *observation*. The task of surgical activity recognition using a surgical workflow is defined as *online* when it is performed for each observation of a surgical procedure, using all observations of the procedure acquired up to that particular observation. This type of recognition corresponds to intraoperative surgical activity recognition. In constrast, the task is defined as *offline* recognition when it is performed for each observation, but based on all observations of the procedure up to the last time step. In the latter case, the recognition can only occur after the end of the execution of the procedure and thus corresponds to postoperative surgical activity recognition.

1.4 Contributions

The aim of this thesis is to address the problem of surgical activity recognition by proposing practical solutions to perform the task in a *real and uncontrolled environment*. To this end, we propose to use visual information obtained from the OR. Specifically, we focus on laparoscopic and multi-view RGBD videos due to their interesting complementary properties.

Since the research in vision-based surgical activity recognition is still at its early stage, one of the main objectives of this thesis is to investigate the most suitable visual feature representations to perform surgical activity recognition. The contributions of this thesis mainly revolve around these two points: (1) the extensive experiments using state-of-the-art methods to generate strong baselines; and (2) the proposal of novel approaches to construct robust pipelines for the surgical activity recognition task. The contributions are specified in detail as follows.

The first contribution is the construction of recognition pipelines for pre-segmented and frame-wise classifications. For pre-segmented classification, we propose to use a method based on the bag-of-word (BOW) approach and Support Vector Machine (SVM). The BOW approach is used to encode the visual features from the data and SVM is used to perform the classification. For the frame-wise classification, we propose to reuse the BOW-SVM pipeline and enforce the temporal constraints using a derivation of Hidden Markov Model (HMM). Combined with various handcrafted visual features used in the state-of-the-art methods, we have used these pipelines to generate strong baselines for several activity recognition tasks: surgery classification (pre-segmented) and surgical phase recognition (frame-wise) on laparoscopic videos, and surgical action classification (pre-segmented) and surgical phase recognition (frame-wise) on multi-view RGBD videos.

As second contribution, we propose a feature encoding approach which is based on the BOW approach. Due to the orderless nature of BOW encoding, the spatial and temporal information from the visual features is typically lost during the feature encoding process. Here, we propose a method which retains this information during the encoding process. This is done by clustering the spatio-temporal locations of the features in a sequence and compute the final histograms using the clusters. We have applied this approach for activity recognition tasks on multi-view RGBD videos and shown that this approach is able to retain spatio-temporal information from the features.

Due to the visual challenges inherent to both laparoscopic and multi-view RGBD videos, designing discriminative features is not a trivial task. Thus, as third contribution, we propose to automatically learn the visual features from the dataset using deep learning methods. We propose and compare multiple CNN architectures designed to perform surgical activity recognition on both video types. In addition, we also present an end-to-end deep architecture to perform surgical activity recognition in the OR. The end-to-end architecture is capable of both automatically learning the visual features as well as enforcing the temporal constraints to the recognition pipeline. Thus, it eliminates the need for HMM in the frame-wise recognition pipeline. In practice, this consists of appending the deep architectures with Long-Short Term Memory (LSTM) networks. Our experiments show that the deep architectures can be used to learn discriminative visual features for surgical activity recognition. These learnt features are shown to outperform other features, i.e., handcrafted visual features, tool signals, and the combination thereof. In addition, the LSTM network outperforms the HMM-based approach and the results show that it can be used to properly enforce the temporal constraints of the surgical workflow, even for long videos of complete surgeries.

The last contribution is the generation of large datasets for activity recognition in the OR. We have collected and labelled hundreds of laparoscopic videos and hours of OR-scene videos during this thesis. These videos, containing real surgical procedures, are the results of a close collaboration with our clinical partners: the University Hospital of Strasbourg/IRCAD and the Interventional Radiology Department of Nouvel Hôpital Civil Strasbourg. These datasets permit the extensive evaluation of the proposed approaches for surgical activity recognition on real data. In addition to being used for the evaluation in this thesis, combined with videos recorded at the Hospital Klinikum Rechts der Isar in Munich [Stauder 2016], a part of this dataset has been used to generate two datasets (*m2cai16-workflow* and *m2cai16-tool*) for challenges on surgical phase recognition and tool presence detection at M2CAI 2016[1], held jointly with MICCAI 2016 in Athens. To further encourage research in this domain, we have also released the Cholec80 dataset which is the largest public cholecystectomy video dataset to date, containing 80 cholecystectomy videos annotated with phase and binary tool labels. In Appendix A, we show all the datasets generated in the course of this thesis.

Additionally, we present in Appendix B a published work [Twinanda 2014a] that is

[1]http://camma.u-strasbg.fr/m2cai2016

not discussed in this thesis, which presents an unsupervised approach to retrieve from the laparoscopic video database the occurrence of a task depicted in a query video.

1.5 Outline

This thesis is organized as follows:

- Chapter 2 contains a literature review of methods that are related to the tasks addressed in this thesis.
- Chapter 3 describes the basic principles of some machine learning algorithms for activity recognition used in this thesis, including Support Vector Machine, Hidden Markov Model and deep learning algorithms.
- Chapter 4 includes surgical activity recognition tasks on laparoscopic videos. The methods presented in this chapter have been published in [Twinanda 2014b, Twinanda 2015b].
- Chapter 5 presents surgical activity recognition tasks on RGBD videos. Several results in this chapter have been published in [Twinanda 2015a].
- Chapter 6 contains the proposed deep architectures to perform some of the tasks addressed in this thesis. This chapter also includes the proposed end-to-end architecture using the LSTM network. Some of the results from this chapter have been published in [Twinanda 2017, Twinanda 2016].
- Chapter 7 concludes the thesis with final remarks and introduces several possible directions to improve the methodology.

2 Related Work

Study the past if you would define the future.
– Confucius

Chapter Summary

The need for activity recognition in the operating room (OR) emerged along with the interest of developing context aware systems (CAS) for the OR. Various types of digital signals from the surgical equipment have been used to address the problem. For example, in an early work [Xiao 2005], patient's vital signs are used to identify the state of the OR (i.e., occupied vs. not occupied). In other studies, surgical tool signals have been used to recognize surgical phases in laparoscopic cholecystectomy procedures [Padoy 2012, Stauder 2014]. However, these aforementioned signals only provide specific and limited information about the activities occurring in the OR.

In this thesis, we focus on using video recordings of the surgical procedures to perform surgical activity recognition. Thus, we focus the discussion in the following sections on vision-based approaches. We first review the state-of-the-art methods for activity recognition in the computer vision community. Then, we present vision-based approaches that have been proposed to address surgical activity recognition. Finally, we explain the position of our work with respect to the existing literature at the end of this chapter.

2.1 Activity Recognition in Computer Vision

Activity recognition is one of the most active research fields in the computer vision community. It is used for a wide range of applications, such as video surveillance [Lin 2008, Kushwaha 2012], video retrieval [Bendersky 2014], and human computer interaction [Murthy 2010, Rautaray 2010]. One of the main challenges in performing activity recognition is the fact that images and videos are complex high-dimensional data, which results in difficulties in designing discriminative feature representations of the data. In this section, we discuss how this challenge is addressed along with the development of the activity recognition field in the computer vision community. First, we present the evolution of video modalities used to perform the task. Then, we review visual feature representations used in state-of-the art methods. Ultimately, we discuss the strategies proposed in the literature to address the pre-segmented and frame-wise classification problems.

2.1.1 Video Modalities

The activity recognition task is a major topic in the computer vision community. Various video modalities have been used to address this task, ranging from thermal infrared images [Han 2005] to multi-spectral images [Cheng 2007]. In this thesis, we are interested in intensity images because the cameras used in the OR typically record videos in this modality. These images are captured by visible-light sensors which are photosensitive devices that are sensitive to light signals in the wavelengths visible to humans. In addition to being generally used in the OR, at the early stage of activity recognition, computer vision methods proposed in the literature perform the task using videos of this modality. Most of these methods [Zelnik-Manor 2001, Laptev 2005, Dollár 2005] perform activity recognition tasks using greyscale videos, while only a small number of works addresses the task using color videos, such as [Rapantzikos 2009]. This is due to the fact that working with greyscale images is less complex than working with color images. Color images are represented in a dimensional space that is higher than the greyscale image space, which results in a higher variability in the data representation and does not necessarily lead to more discriminative information for activity recognition. Moreover, because of the data representation, color images are more computationally expensive to process. Furthermore, color images can be represented in various color spaces and it is not trivial to find the most suitable one.

The setback of using a visible-light sensor is that there is an important information that is not directly captured in the recordings obtained from the static monocular video sensor, namely the 3D information of the scene. Because of this reason, 3D sensors, such as time-of-flight sensors [Wallhoff 2007], have also been used to capture the 3D information of the scene for activity recognition purposes. Even though the development of human activity recognition using depth sensors to obtain the 3D information began in the early 1980s [Aggarwal 2014], there had not been many efforts addressed in this direction due to the cost of such sensors. Only after the recent emergence of affordable and reliable depth sensors, more studies have been directed at exploiting depth information for the activity recognition task. Thanks to the complementary information provided by such multi-modal sensors, improvements can be observed in the results of activity recognition methods which combine the information coming from both intensity and depth images [Sung 2012, Ni 2013, Xia 2015].

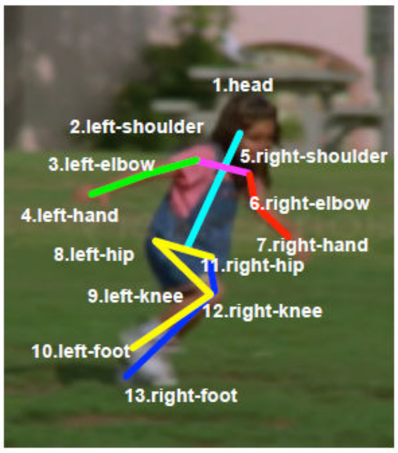
A comprehensive review of the activity recognition approaches developed for RGB and RGBD data can be found in [Aggarwal 2011] and [Aggarwal 2014], respectively.

2.1.2 Visual Feature Representation

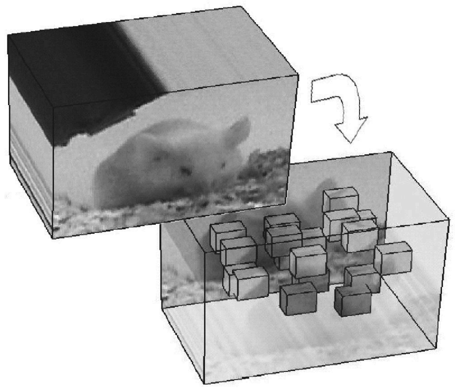
In computer vision, the visual feature representations can be generally grouped into two categories: spatial and spatio-temporal features. The spatial features capture the meaningful information from images at a single frame of time, such as color information [Jain 1996] and texture features [Manjunath 1996]. These features can be regarded as *global descriptors* since they are typically extracted to represent a complete image. In addition to these global descriptors, there exist *local descriptors* which capture the characteristics of small image patches, for example interest point detectors and descriptors (e.g., Harris corner detector [Harris 1988], scale-invariant feature transform (SIFT) [Lowe 2004], speeded-up robust features (SURF) [Bay 2008]), and histogram of gradients (HOG) [Dalal 2005]. They are ideally invariant to background clutter, appearance, occlusions, and also to rotation and scale in some cases. These local descriptors can be aggregated to construct a global descriptor using feature encoding methods, such as the bag-of-word (BOW) approach. The basic idea of BOW approach is to group the local descriptors into “bags" and count the content of all bags to generate a histogram of visual features.

These aforementioned spatial features are considered as low-level visual features since they do not contain any semantic information. To provide more semantically meaningful information, high- level features, such as the results of person and/or object detection (e.g., [Wu 2007]) and human pose estimation (e.g., [Xu 2012]), are then utilized for activity recognition. By using such high-level features, the classification algorithm will have better performance in modeling the activities performed in the scene. For example, in Fig. 2.1-a, an illustration of 13 point skeleton used in [Xu 2012] is shown. Intuitively, the sequence of the body joint locations provides more information regarding the activities than the low-level features.

In contrast to the spatial features, spatio-temporal features, which are captured from image sequences, consider not only the spatial information, but also the temporal information from the sequences. The temporal information typically involves the movements of certain parts of the images. For instance, optical flow-based features [Chaudhry 2009] represent the global pixel movements in the complete image sequence. Similarly to the spatial features, there exist also spatio-temporal features which capture the local characteristics of the sequence. For example, in [Laptev 2005], a spatio-temporal interest



(a) courtesy of [Xu 2012]



(b) courtesy of [Dollár 2005]

Figure 2.1: Examples of visual features: (a) human pose skeletons and (b) extracted cuboid features.

point (STIP) detector, which is an extension of Harris corner detector to space-time dimension, was proposed. The detector was shown to successfully capture significant local structures in spatio-temporal domain and perform very well in detecting events from short image sequences. Inspired by the success of the STIP detector, several derivations of STIP have been proposed in the literature, such as the cuboid detector and descriptor [Dollár 2005]. In Fig. 2.1-b, an illustration of the cuboid extraction [Dollár 2005] from an image sequence is shown. The cuboid detector detects salient points (shown as small cubes) in the spatio-temporal volume. Feature descriptors, such as histogram of optical flow (HOF), are then extracted around these salient points and later collected to represent the sequence. In addition to being applied to intensity image sequence, the STIP detector has also been adapted to extract STIP features on depth images [Xia 2013].

The aforementioned visual features are referred to as *handcrafted* features because they are empirically designed to capture certain characteristics from the data. For example, color histograms are constructed to solely capture the information of color distribution and SIFT features are designed to represent the intensity gradients around salient points. In addition, encoding methods, such as the BOW approach which simply performs counting on the occurrence of similar features, cause further information loss during the feature extraction process. This results in the interest in performing feature learning where the purpose is to automatically learn the feature representation from the raw data.

In the recent years, feature learning methods, especially deep learning algorithms, have gained a lot of interests in the computer vision community because of their success in performing various computer vision tasks, ranging from object detection [Girshick 2014], human pose estimation [Toshev 2014], to activity recognition [Tran 2015]. Such approaches owe their success to the dramatic computational improvement of graphical processing units (GPUs) and the availability of a large amount of data since deep learning algorithms, such as convolutional neural networks (CNNs), typically involve an optimization over a large number of unknowns. For example, in [Krizhevsky 2012], a CNN architecture, referred to as *AlexNet*, containing 60 million unknowns is proposed to solve the image classification task on the ImageNet dataset. The results show that the network can automatically learn discriminative spatial feature representations from the images to perform the task. The presented approach significantly outperforms other state-of-the-art methods which utilize handcrafted visual features, such as Fisher Vector on SIFT [Sanchez 2011] and sparse-coding models on various handcrafted features [Lin 2011]. In addition to learning the spatial features, the deep learning algorithms have also been shown to be successful in learning the spatio-temporal features for activity recognition in [Tran 2015]. More technical details regarding feature learning using neural networks are presented in Section 3.3.

2.1.3 Pre-segmented and Frame-wise Classification

At the early stage of activity recognition in the community, both pre-segmented and frame-wise classification problems are typically addressed by using a two-step pipeline: (1) extracting the visual features from the videos and (2) passing them to a classification algorithm. In Section 2.1.2, we have discussed the various visual feature representations used to perform activity recognition. Due to the nature of pre-segmented classification, the applied classification algorithms are not required to model the temporal information in the recognition pipeline. Thus, classification algorithms, such as instance-based learning (e.g., K-NN [Liu 2015]) and discriminative modeling (e.g., SVM [Xia 2013]), can be used to address the pre-segmented classification task. On the other hand, to properly perform frame-wise classification, the classification algorithms should incorporate a temporal model which models the temporal dependency in the time-series input of variable-length. One of the well-known approaches to address this problem is dynamic time warping (DTW) [Sakoe 1978] which computes the similarity between two sequences. However, in order to perform classification accurately using DTW, a large amount of sequence examples is required, resulting in a high computational cost. To solve this problem, generative models, such as hidden Markov models (HMM) and dynamic Bayesian networks (DBN), are proposed to perform this task [Ryoo 2009, Kuehne 2014]. A DBN is a kind of Bayesian network where the connections between states in the network follow the direction of the forward flow of time, while an HMM is a special kind of DBN where the states are connected under the first-order Markov assumption. Despite the fact that the DBN provides a more general model than the HMM by relaxing the constraints in HMM, it comes with a higher complexity and computational cost as well as the requirement of a lot of training data.

As CNN-based methods gain a lot of interests in the computer vision community, more efforts are directed to the construction of *end-to-end approaches*, where both feature extraction and classification are performed in a one-step solution. Despite the fact that there exists CNN architectures designed to extract spatio-temporal features [Tran 2015], such CNN architectures only work on fixed-length sequences. Thus, an additional step, consisting of feature pooling and classification, is required to perform activity recognition on long sequences, breaking the end-to-end cycle. To this end, recurrent neural networks (RNNs) [Rumelhart 1988], which are a family of neural network that operates in time domain, are proposed to incorporate the temporal information. However, it is shown in [Bengio 1994] that the RNNs are difficult to train because of the so-called "vanishing/exploding gradients" phenomenon, where the weight updates exhibit exponential decay/growth as they are back-propagated through time. To sidestep the difficulty of training RNNs, a modified architecture called Long Short-Term Memory (LSTM) was proposed in [Hochreiter 1997]. Recently in [Donahue 2015], the combination of CNN and LSTM is shown to perform very well in various tasks, such as image and video description, and activity recognition. Since this approach provides an end-to-end alternative for feature learning which also incorporates the temporal information into the learning process, the architecture has been used in several approaches to address activity recognition [Ma 2016, Mahasseni 2016]. Due to memory constraints, most LSTM-based approaches proposed in the literature typically address activity recognition on short sequences. For example, in [Mahasseni 2016] fixed sequences of length 16 frames are used during the LSTM training, and in [Ma 2016] 64 sequences of length 100 frames are utilized.

2.2 Vision-Based Surgical Activity Recognition

Despite the fact that activity recognition is one major topic in the computer vision community, there has only been a few vision-based methods proposed in the literature to perform this task in surgical setups due to several reasons. One reason is the fact that the acquisition of videos in the OR is still difficult either because of the unavailability of the essential equipment or due to the strict regulations for safety and legal reasons. In addition, surgical activity recognition is a relatively new research topic which addresses a particular problem different from the typical activity recognition task. It is a very challenging task to carry out due to various visual challenges. For example, the OR is a complex environment with a crowded scene, which introduces clutter and occlusions in the OR-scene videos. Furthermore, the tool-tissue interaction videos capture a significantly different scene than the ones in traditional computer vision videos. Specifically in the laparoscopic videos, visual challenges, such as specular reflection, rapid camera motions, the presence of smoke, and occluded lens, are inherently present in the videos. These visual challenges make it difficult to construct robust pipelines for surgical activity recognition.

Thanks to the continuous efforts from the community, there are a few surgical activity recognition datasets that have been released. One of the public datasets is the JIGSAWS dataset [Gao 2014] from the Johns Hopkins University. It contains recordings of surgical tasks (i.e., suturing, knot-tying, and needle-passing) performed using the da Vinci Surgical System. The dataset includes the videos and the kinematic data of the robotic arms from the system. This dataset, however, is dedicated to surgical skill assessment and contains short recordings. Recently, EndoVis[1], a dataset for surgical phase recognition, was released. The dataset contains 7 laparoscopic recordings of cholecystectomy procedures, each of which is labelled with the surgical phases and the tool usages.

In the literature, there exist several vision-based methods which address tasks related to surgical activity recognition, such as surgical skill assessment [Islam 2013], surgical tool classification [Speidel 2009], and surgical tool detection and tracking [Voros 2007, Sznitman 2014, Allan 2015]. However, these tasks are not only different from the surgical activity recognition tasks that we address in this thesis, but also they typically deal with still images or short videos, while, in our case, the videos can be hours long.

In this section, we first discuss the existing approaches to model the surgical workflow. Then, we focus the discussions solely on vision-based solutions for surgical activity recognition since this is the focus of the thesis. In Section 2.2.2, we discuss the approaches which address surgical activity recognition on videos which capture the tool-tissue interactions (e.g., laparoscopic videos). In Section 2.2.3, we present several methods performing the tasks on OR-scene videos.

2.2.1 Surgical Workflow Modeling

One way to perform the analysis, visualization, and optimization of surgical workflows is by formally describing the surgical activities in the OR. For example, a unified modeling language (UML-) based approach was presented in [Jannin 2001] to understand and optimize the usage of imaging modalities during a neurological procedure. This model has been shown to be useful for preoperative planning [Jannin 2007]. In a more recent work [Katić 2014], an ontology-based approach is proposed to model the surgical process for three kinds of laparoscopic procedures: cholecystectomy, adrenalectomy, and pancreatic resection. Using rule-based methods on top of the formalization, such an expert-based method is capable of modeling complex surgical workflow for any type of procedure. However, these approaches typically require manually provided high-level information for the modeling. For example, in [Katić 2014], an activity is described by a triplet consisting of the used instrument, the performed action, and the organ acted upon (e.g., <grasper, grasp, liver>). Despite the fact that the method [Katić 2014] has been successfully tested on a dataset containing 19 surgeries of 3 different procedures, such triplet information is still difficult to be obtained reliably using available signals in the OR. This renders recognition methods relying on these modeling approaches impractical for surgical activity recognition.

Instead of using formal definitions of the surgical workflows, several works have addressed the automatic generation of workflow models from training data using graphical models. For example, an approach based on HMM [Blum 2008] was presented to derive the workflow of cholecystectomy procedures using the process logs, containing the tool usage of ten procedures. In other works [Padoy 2009, Blum 2010, Lalys 2012], the workflow is learnt from the dataset annotations using similar graphical models and the appearance

[1]http://grand-challenge.org/site/endovissub-workflow/data/

models are statistically derived using visual features extracted from the videos. Despite the fact that such approaches contain a lack of semantics, they are in practice more advantageous for the surgical activity recognition task than other methods which rely on modeling languages without any direct connection to the low-level signals from the OR.

For a review of surgical workflow modeling, we refer the reader to [Lalys 2014].

2.2.2 Activity Recognition on Tool-tissue Interaction Videos

One of the earliest work of activity recognition on tool-tissue interaction videos is [Lo 2003], where a pipeline to segment laparoscopic videos is proposed. Using a naïve Bayesian network on top of several visual cues related to shape, deformation, changes in light reflection and other low level visual features, the method yields promising results in segmenting five videos into four major events: idle, retraction, cauterization, and suturing.

Following work has used tool usage signals, assuming that this information could be obtained either through further analysis of the videos or through other sensors. They have been shown to yield satisfactory results for surgical activity recognition using DTW or HMM [Ahmadi 2006, Padoy 2007a, Blum 2008]. Later, [Padoy 2008] proposed a pipeline which combines visual features and tool usage signals to perform the task. Passed to an HMM-based pipeline, the feature combination yields promising results in recognizing the phases of 11 cholecystectomy procedures. However, it is still difficult to obtain the tool usage signals at test time, since they are typically obtained either through manual annotation or by using a particular equipment. To alleviate this problem, a method which uses the tool usage signals only at training time is proposed in [Blum 2010]. The tool signals are used to compute the projection matrix to reduce the dimension of the handcrafted visual features (i.e., vertical and horizontal gradient magnitudes, histograms and the pixel values of the resized image) through canonical correlation analysis (CCA) in order to obtain more semantically meaningful and discriminative visual features. Tested on seven cholecystectomy videos, the CCA-based method is shown to perform better in generating discriminative features than the PCA-based method.

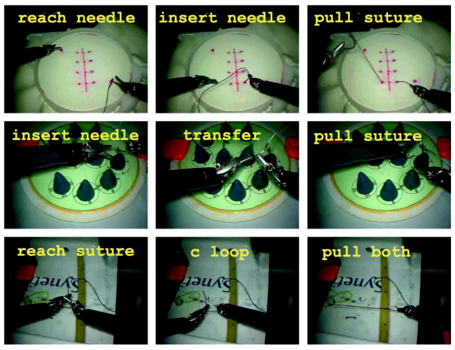
In addition to the tool usage signals, tool motion data can also be used as a complementary information to the visual features. For instance, [Zappella 2013] used tool-tissue interaction videos along with the kinematic data obtained from the robotic console to classify surgical gestures in the suturing action, such as insert needle and pull suture. In Fig. 2.2-a, we show image samples from the dataset, which shows 9 different gestures that are subject to the classification model. From the videos, histograms of oriented gradients (HOG) and histograms of optical flows (HOF) are extracted around the detected STIPs from the videos. Since the videos are recorded on training pods in a laboratory setup, the videos used for the experiments in [Zappella 2013] typically are not as long as videos recording real procedures, i.e., two minutes long in average.

In another work [Lalys 2011], a vision-based method was proposed for the extraction of surgical phases with microscope videos and was validated in the context of neurosurgical procedures. A multitude of visual features, such as color, texture, and shape information, are extracted from the video frames. Using the combination of SVM and HMM on top of the visual features, the pipeline yields promising result for the tasks. Similar features are also used in another work [Lalys 2012], where a vision-based method to segment the surgical phases in cataract surgery (image samples shown in Fig. 2.2-b) was proposed. Combined with DTW, the mixture of these features is shown to perform the task very well.

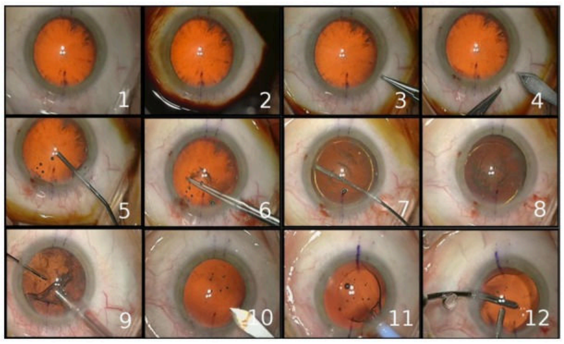
In [Quellec 2014a], an approach based on the content-based video retrieval (CBVR) paradigm to address activity recognition on eye surgery videos is proposed. The method characterizes short subsequences from the video and look up similar annotated video clips in the video database. In order to guarantee the method to run in real time, simple features, such as color, texture, and simple motion features, are used. Evaluated on cataract and retinal surgery videos, the pipeline shows promising results in recognizing the tasks. In [Quellec 2014b], another real-time approach is proposed to perform jointly the segmentation and classification of surgical actions in cataract surgeries. The method first segments the video into clips of idle and action phases, then classifies the clips containing action phases via a CBVR-based approach. Using a conditional random field (CRF) on top of motion features, the method yields satisfactory results. In a more recent work [Quellec 2015], a vision-based approach, consisting of spatio-temporal polynomials to perform the motion analysis, was proposed to address activity recognition on cataract videos. The results show that the method outperforms the approaches proposed in [Quellec 2014a] and [Quellec 2014b]. These real-time approaches are particularly based on motion features which can be reliably extracted from eye surgery videos thanks to the usage of static microscope. However, it is very challenging to extract such motion features from laparoscopic videos due to the rapid motion of the endoscope. Despite the fact that it is interesting to establish real time recognition systems, here, we do not restrict our approaches to be real-time and instead focus on studying various visual feature representation for the surgical activity recognition.

In a more recent work [Charriere 2016], a real-time method to jointly recognize two granularity levels of activities in cataract surgeries was proposed. The experimental results show that the tool usage signals outperform the visual information. This is however expected since the tool usage signals contain more discriminative and semantic information compared to the low-level visual features extracted from the videos (e.g., pixel motion information and STIP). In another recent work [Dergachyova 2016], an HMM-based pipeline addressing surgical phase recognition on laparoscopic videos is presented. The visual features extracted from the videos include color, texture and shape information. Obtained from evaluations on the EndoVis dataset, the results demonstrate a similar trend shown in [Charriere 2016] that the tool usage signals are better than the visual features in performing the task. In addition, in accordance with the results presented in [Padoy 2008], the results in [Dergachyova 2016] show that the combination of the visual features and the tool usage signals yields the best recognition results.

Similarly to the activity recognition task in the computer vision community, one of the main challenges is to design discriminative feature representations from the videos. Classical handcrafted features, such as color, intensity gradients, and spatio-temporal



(a) courtesy of [Zappella 2013]



(b) courtesy of [Lalys 2012]

Figure 2.2: Image samples from tool-tissue interaction videos: (a) suturing pod recordings and (b) cataract procedures.

features, have been used in the aforementioned studies to perform the task. To alleviate the difficulties in designing the feature representations, several studies have also resorted to feature learning approaches. In [Klank 2008], an approach based on genetic algorithm is proposed to learn automatically the feature representations for surgical phase recognition on laparoscopic videos. The objective of the genetic algorithm is to construct the best sequence consisting of basic low-level (e.g., plus, minus, and subtract) and high-level operators (e.g., dilate, gauss, and histogram). Once an optimal sequence is obtained, the features are extracted from the sequences and passed to an SVM model to obtain the recognition results. The experimental results demonstrate that the learnt features are more discriminative than handcrafted features, such as HSV histogram and local binary pattern.

Due to the limitation of large public datasets in the medical community, there has only been a few deep learning approaches proposed to address the surgical activity recognition tasks on tool-tissue interation videos. In [Lea 2016], a CNN-based approach is proposed to perform fine-grained action segmentation, which is evaluated on the JIGSAWS dataset. The proposed approach consists of spatial and temporal convolutional components. The experimental results demonstrate that the proposed approach is better than another deep architecture (i.e., VGG [Simonyan 2014b]) and traditional pipelines which involve handcrafted visual features.

2.2.3 Activity Recognition on OR-Scene Videos

One of the earliest work that uses OR-scene recordings in the OR was presented in [Bhatia 2007], where four OR occupancy states are identified using the videos recorded by a ceiling mounted camera. It was shown that the results obtained using the videos are more accurate and have less delay than the ones obtained using the patient's vital signs [Xiao 2005].

In [Padoy 2009], a system using OR-scene videos is proposed to perform surgical workflow modeling, which consists of identifying eleven states of the OR. A multi-view

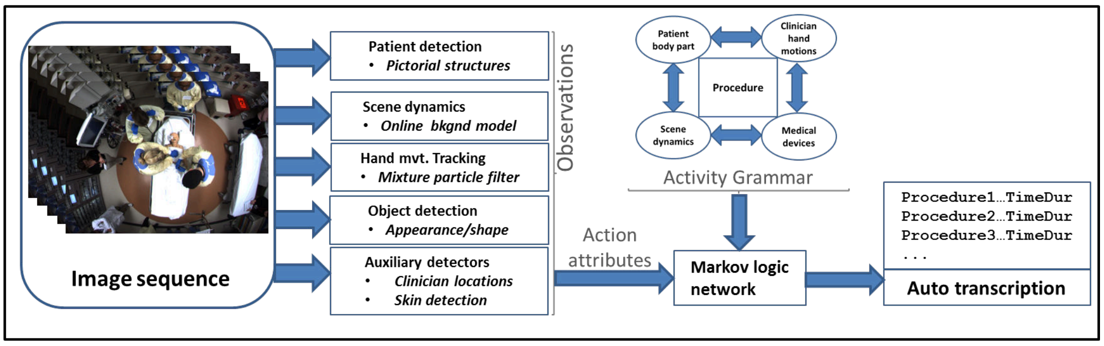


Figure 2.3: The pipeline presented in [Chakraborty 2013] for transcription of trauma resuscitation in the emergency room.

camera system composed of nine cameras is utilized to obtain the sequence of 3D occupation grids of the scene. The proposed method extracts 3D motion features which are computed from the sequence of 3D occupation grids. The features, quantized in 3D, are then passed to hierarchical HMM (HHMM) to perform the recognition task. However, this method has only been tested on a dataset containing recordings in a laboratory setup.

Another work [Chakraborty 2013] presented a pipeline (shown in Fig. 2.3) to perform the automatic transcription of trauma resuscitations in the emergency department. In the pipeline, high-level features, such as the pictorial structure of the patient and hand movement tracking, are extracted. These features are then used to infer the activities using a Markov Logic Network (MLN). Using an RGB camera mounted directly above the bed, a dataset, containing one video of trauma resuscitation simulation, is used to perform the evaluation of the method. The results, albeit on a small dataset, suggest the suitability of the framework to perform the task. However, it is important to note that the method utilizes an MLN where the set of logic rules are handcrafted and the weights are manually specified by an expert. In addition, the aforementioned high-level features can only be extracted from the videos thanks to the strategic camera positioning. This configuration is however not always possible in every OR, especially in a hybrid OR because it could interfere with the motion of articulated arms where the surgical devices are mounted. Thus, it is interesting to see other studies addressing surgical activity recognition using videos captured from other camera configurations.

As RGBD sensors are getting more popular in the computer vision community, several approaches addressing surgical activity recognition also resort to this solution. In [Lea 2013], the problem of pre-segmented activity recognition in an intensive care unit (ICU) using an RGBD sensor was addressed. The method uses the position- and orientation-based features extracted from the segmented persons in the depth sequences to perform the task. The method demonstrated good performance in classifying seven actions on a dataset of 122 action sequences. In a recent work [Reiter 2016], a method using an RGBD camera in the ICU is proposed to perform patient activity tracking. The method consists of the extraction of information, such as the results of patient identification, pose

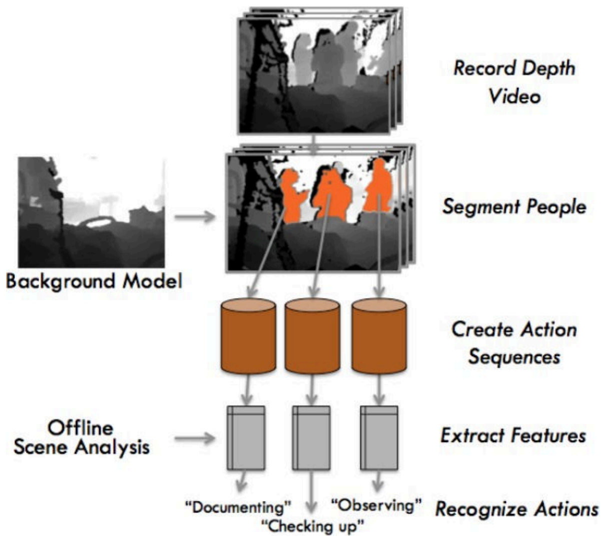


Figure 2.4: The method presented in [Lea 2013] addressing surgical activity recognition in ICU using depth sequences.

estimation and tracking, and object detection. A video clip is then represented by an attribute feature which consists of the concatenation of the high-level information. It is shown that the person and pose classification frameworks yield satisfactory results, but it is not the case for the motion tracking framework. Despite this fact, the pipeline is shown to perform well in classifying the video segments into four coarse activity categories: nothing, in-bed, out-of-bed, and walking. Both studies [Lea 2013, Reiter 2016] show promising results on activity recognition using RGBD videos, however further explorations regarding the visual features might be required to address the problem of recognizing other activities which involve finer movements and gestures.

2.3 Thesis Positioning

In this thesis, we address the problem of surgical activity recognition, specifically we aim to provide *practical* solutions to perform the task in a *real and uncontrolled environment*. Here, we propose to use solely the visual information coming from cameras inside the OR since they are either essential for the execution of many types of surgeries or typically installed in modern ORs. Despite the fact that the tool information is shown to perform better than the visual features in the very recent work [Dergachyova 2016, Charriere 2016], it is still not a trivial task to obtain such tool usage signals during surgeries. This information is typically obtained through either manual annotation or the usage of an additional equipment to capture the data, which renders the approaches impractical. Here, we perform surgical activity recognition tasks on two video types: laparoscopic videos and multi-view RGBD videos.

In addition, to provide practical solutions to perform the task, we propose to model the workflow using discriminative or generative approaches on visual features [Blum 2010, Lalys 2012], instead of the rule-based approach on formal language definitions [Jannin 2007, Katić 2014]. As discussed in 2.2.1, it is still an open question how such formal languages will offer a practical solution to perform the surgical workflow recognition due to the fact that they typically require high level information (e.g., the tool usage and the organ acted upon) which is not yet trivial to obtain from the OR.

In previous sections, we have described the existing methodologies addressing vision-based activity recognition tasks on both medical and non-medical contexts. One can observe that the number of studies addressing surgical activity recognition is still limited and the problem still remains unsolved. In this thesis, we address these limitations in three steps: (1) by generating large datasets containing recordings of real surgeries in two modalities (i.e., laparoscopic and multi-view RGBD videos), (2) by providing thorough analysis on the performance of state-of-the art methods and (3) by proposing novel approaches to perform the surgical activity recognition task.

To investigate the performance of state-of-the-art methods, we first perform extensive experiments on large datasets to establish strong baselines. In contrast to [Padoy 2009, Béjar Haro 2012, Zappella 2013, Chakraborty 2013], we perform the experiments on large datasets containing recordings of real surgeries. For both pre-segmented and frame-wise classification tasks, we construct a pipeline which consists of: (1) the extraction of off-the-shelf visual features [Blum 2010, Lalys 2012, Zappella 2013, Dergachyova 2016], such as color information, intensity gradients, spatial and spatio-temporal interest points; (2) the conventional bag-of-word (BOW) feature encoding method [Béjar Haro 2012, Zappella 2013, Charriere 2016]; and (3) the usage of traditional classification frameworks, such as SVM [Béjar Haro 2012, Zappella 2013] and HMM [Padoy 2009, Blum 2010, Lalys 2012, Dergachyova 2016].

Since the BOW approach only performs simple counting of the occurrence of the visual features, a lot of information can be lost during the feature encoding process. Here, we propose a novel feature encoding method which retains the spatio-temporal information from the features. This is inspired by the spatial pyramid matching (SPM) proposed in [Lazebnik 2006]. Instead of using a rigid grid to divide the space of the feature locations into cells, we propose to learn the cells by clustering on the spatial (or spatio-temporal) locations of the features and compute the final histograms using the obtained clusters.

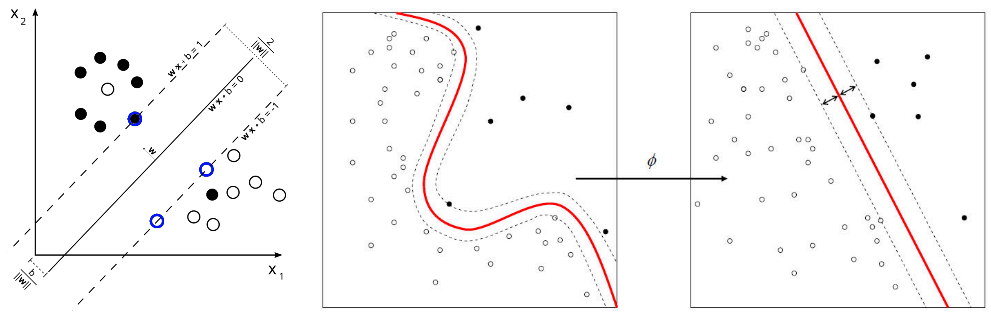
In Section 2.2, it can be observed that most of the visual features used in the literature to perform surgical activity recognition are handcrafted ones. Despite being widely used for computer vision tasks [Krizhevsky 2012, Girshick 2014, Tran 2015], the deep learning approach is yet to be adapted for surgical activity recognition. Here, we propose several novel CNN architectures to perform surgical phase recognition on both laparoscopic and multi-view RGBD videos. To perform the frame-wise classification task, we first propose to use the SVM-HMM pipeline using the features learnt from the CNNs. Recently, in [Dergachyova 2016], it has been shown for phase recognition on a small dataset of tool-tissue interaction videos that (1) the manually obtained tool information outperforms the visual features and (2) the combination of visual features and manually obtained tool information yields the best results to perform surgical activity recognition. Here, we show that this is not the case anymore when learnt visual features are used to perform the task. The visual features learnt by our proposed approaches outperform tool binary signals, handcrafted visual features, and the combination thereof. Ultimately, we design end-to-end networks consisting of CNN and LSTM networks in order to alleviate the need of SVM-HMM pipeline to perform the frame-wise classification tasks. In contrast to [Ma 2016, Mahasseni 2016], here, the LSTM networks are trained on long videos (up to 2 hours long). To the best of our knowledge, this work and [Lea 2016] are the first few studies which propose deep learning approaches to address surgical activity recognition.

3 Classification Frameworks for Activity Recognition

If I have seen further, it is by standing on the shoulders of giants.
– Sir Issac Newton

Chapter Summary

This chapter serves as a support to provide the fundamental knowledge regarding the machine learning methods used in this thesis. This includes Support Vector Machine (SVM), hidden Markov model (HMM), and deep neural networks.



(a) (b)

Figure 3.1: Illustration of: (a) linear SVM and (b) non-linear SVM with a kernel function.

3.1 Support Vector Machines

Support Vector Machines (SVMs) are supervised learning models used for classification and regression analysis. Here, we focus the discussion on using SVM for classification tasks. For *linearly* separable training data, the goal of SVM learning is to find an optimal linear hyperplane between two classes that is maximally far from any point in the training data $\mathbf{x} \in \mathbb{R}^M$ in M-dimensional space. The training samples closest to this hyperplane are called support vectors, hence the name SVM. The illustration of SVM is shown in Fig. 3.1-a. In the graph, the support vectors are shown in blue circles.

3.1.1 Formalization

For a binary classification problem, the SVM separates the data into two classes using two parallel hyperplanes that can be described by the following equations:

$$\mathbf{w} \cdot \mathbf{x}_+ + b = 1, \tag{3.1}$$

$$\mathbf{w} \cdot \mathbf{x}_- + b = -1, \tag{3.2}$$

where $\mathbf{x}_+$ and $\mathbf{x}_-$ are respectively the support vectors from positive and negative classes, and $\mathbf{w}$ and b are respectively the SVM weight vector and bias. These hyperplanes are shown as the dashed lines in Fig. 3.1a.

Since no data point should fall into the margin, the SVM model is also constrained by:

$$\mathbf{w} \cdot \mathbf{x}_+ + b \geq 1, \tag{3.3}$$

$$\mathbf{w} \cdot \mathbf{x}_- + b \leq -1. \tag{3.4}$$

By introducing a variable $y = 1$ for $\mathbf{x}_+$ and $y = -1$ for $\mathbf{x}_-$, for all N data points, Eq. 3.3 and 3.4 can be rewritten as:

$$y_i\left(\mathbf{w} \cdot \mathbf{x}_i + b\right) \geq 1, \quad 1 \leq i \leq N \tag{3.5}$$

for all training samples $\mathbf{x}_i$ with label y_i.

The optimal SVM hyperplanes can be obtained by maximizing the distance between them, which can be geometrically computed by $\frac{2}{\|\mathbf{w}\|}$. Therefore, the optimization problem can be formulated as:

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2, \text{ subject to Eq. 3.5.} \tag{3.6}$$

The factor $\frac{1}{2}$ in Eq. 3.6 is included for convenience in the optimization process. This constrained optimization is solved by introducing Lagrange multipliers to build a Lagrangian function, which boils down to an optimization of a quadratic function. For detailed information regarding the optimization process, we refer the reader to [Bishop 2006].

At test time, to classify a new data $\breve{\mathbf{x}}$, the confidence score ρ is computed from:

$$\rho(\breve{\mathbf{x}}) = \sum_{i=1}^{N} \hat{a}_i y_i K(\breve{\mathbf{x}}, \mathbf{x}_i) + \hat{b}, \tag{3.7}$$

where $\hat{a}_i \geq 0$ are Lagrange multipliers obtained from the optimization process, $\hat{b}$ is the optimal SVM bias parameters, and $K(\breve{\mathbf{x}}, \mathbf{x}_i)$ is the kernel function which will be explained in Section 3.1.3. For a linear kernel, this kernel is defined by the dot product between $\breve{\mathbf{x}}$ and $\mathbf{x}_i$. The data $\breve{\mathbf{x}}$ is classified based on $sign(\rho)$.

3.1.2 Constraint Relaxation

The constraints expressed in Eq. 3.5 is very strict since they assume the training data is completely linearly separable. However, this is very unlikely in real world scenario (as illustrated in Fig. 3.1-a, where a black dot is found in the white region and vice versa). The constraints need to be modified so that the data points are allowed to be on the "wrong side" of the margin boundary, but with a penalty that increases with the distance from that boundary. In [Cortes 1995], this is done by introducing *slack variables* $\xi_i \geq 0$ for each data point. These variables are defined as either $\xi_i = 0$ if the data points are on/inside the correct boundary (i.e., $y_i f(\mathbf{x}_i) \geq 0$) or $\xi_i = |y_i - f(\mathbf{x}_i)|$ otherwise.

The slack variables are introduced to relax the constraint, resulting in:

$$y_i (\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 - \xi_i, \tag{3.8}$$

as the constraint for soft-margin optimization. The optimization goal is now to maximize the margin while penalizing points that lie on the wrong side of the boundary. The minimization problem is formulated as:

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N} \xi_i, \text{ subject to Eq. 3.8.} \tag{3.9}$$

where $C > 0$ controls the trade-off between the slack variable penalty and the margin.

3.1.3 Non-Linear Classification

The goal of SVM is to find the optimal *linear* hyperplane to separate the data points. However, it is often the case that the data points are not linearly separable. In [Boser 1992], a kernel trick approach was introduced to perform non-linear classification using SVM. Instead of using the original feature space of the data points directly, the data points are mapped to another feature space where the data points are linearly separable. This feature mapping is illustrated in Fig. 3.1-b, where the data is mapped to a feature space where it is linearly separable. The kernel trick is an interesting approach because it eliminates the need to compute the actual feature mapping, hence the term "*trick*". By applying this kernel trick, one just needs to replace the kernel function in Eq. 3.7 with another function:

$$K(\breve{\mathbf{x}}, \mathbf{x}_i) = \langle \phi(\breve{\mathbf{x}}), \phi(\mathbf{x}_i) \rangle \tag{3.10}$$

where $\phi(\cdot)$ denoted the feature-space transformation function.

Several commonly used kernels include:

- Polynomial kernel: $K(\breve{\mathbf{x}}, \mathbf{x}_i) = (\langle \breve{\mathbf{x}}, \mathbf{x}_i \rangle + 1)^p, \quad p \in \mathbb{N}$
- Radial basis function (Gaussian) kernel: $K(\breve{\mathbf{x}}, \mathbf{x}_i) = exp\left(\dfrac{\|\breve{\mathbf{x}} - \mathbf{x}_i\|^2}{2\sigma^2}\right)$
- Chi-square (χ^2) kernel: $K(\breve{\mathbf{x}}, \mathbf{x}_i) = \sum_{i=1}^{N} \sum_{m=1}^{M} \dfrac{2\breve{x}^m x_i^m}{(\breve{x}^m + x_i^m)}$
- Histogram intersection kernel: $K(\breve{\mathbf{x}}, \mathbf{x}_i) = \sum_{i=1}^{N} \sum_{m=1}^{M} min(\breve{x}^m, x_i^m)$

In practice, the choice of optimal kernel is not a trivial task. It is essentially a hyper-parameter optimization problem, which is far from being solved. Most of the time, the methodology employed to obtain the optimal kernel is cross-validation [Golub 1979].

3.1.4 Multi-Class SVMs

Fundamentally, SVMs are designed to deal with binary classification problems. Yet, in practice, the problems typically involve $K > 2$ classes. Various SVM extensions have been proposed to address such problems. The most straightforward approach is by reducing the multi-class problem to multiple binary classification problems.

The first approach is the one-versus-all (OVA) approach. For a K-class problem, it constructs K binary SVMs. The k-th SVM model $\mathfrak{C}_k$ is trained with all the examples belonging to class k as positive samples and the rest as negative samples. At test time, for a new example $\breve{\mathbf{x}}$, the confidence ρ_i is computed using each of the SVM models. The approach assigns it to the class that corresponds to the largest confidence.

The second approach is the one-versus-one (OVO) approach. This approach constructs one binary classifier for every pair of distinct classes, resulting in $K\,(K-1)\,/2$ SVM models. The SVM model $\mathfrak{C}_{kl}$ is trained by taking the examples from class k as the positive samples and the examples from class l as the negative samples. At test time, different strategies can be applied for the class decision. The same strategy to the one applied in the OVA approach can also be used for the OVO approach. However, the voting scheme is shown to work better for the OVO approach: if the SVM model $\mathfrak{C}_{kl}$ assigns the data to class k, then the vote for k is added by one. The new data is assigned to the class with the most vote.

3.1.5 Multiple Kernel Learning

Selecting the kernel function $K(\cdot,\cdot)$ and its corresponding parameters is an important issue in training. Generally, the best performing kernel function is chosen by a cross-validation procedure. In recent years, multiple kernel learning (MKL) methods have been proposed where multiple kernels are used instead of using one specific kernel. The main objective is to compute an optimal function to combine multiple kernels. The MKL methods are working under the observations that: (1) each kernel function has its own notion of similarity between the data and (2) different kernels might be using inputs that come from different feature representations. Thus, instead of trying to find which kernel works the best, it is better to pick a collection of kernels and learn the best combination. For a comprehensive review of MKL methods, we refer the reader to [Gönen 2011].

3.2 Hidden Markov Models

Hidden Markov Models (HMMs) are probabilistic graphical state models that have been utilized to model time-series data. The goal of HMM learning is to find the optimal state and state transition parameters which model the temporal information from the training data.

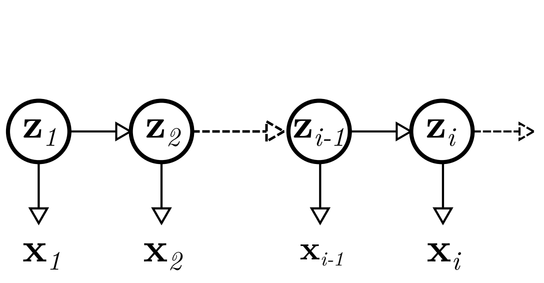
3.2.1 Formalization

Given a sequence of N observations $\mathbf{X} = \mathbf{x}_1, \ldots, \mathbf{x}_N$, an HMM is used to model the joint probability distributions over the underlying state graph. The HMM has three important properties. First, it assumes that the observation $\mathbf{x}_i$ is generated from some process whose state $\mathbf{z}_i$ is *hidden* from the observer. Second, the state of this hidden process satisfies the *first-order markov chain*, such that:

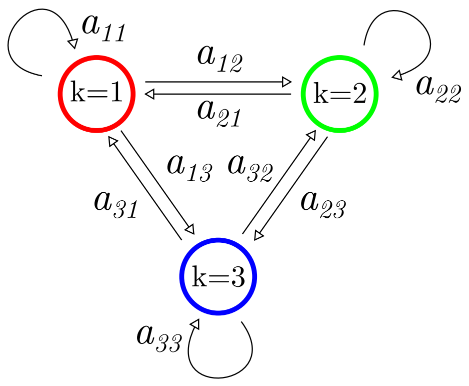
$$p(\mathbf{z}_i|\mathbf{z}_1, \ldots, \mathbf{z}_{i-1}) = p(\mathbf{z}_i|\mathbf{z}_{i-1}). \tag{3.11}$$

This means that the state $\mathbf{z}_i$ only depends on the most recent state $\mathbf{z}_{i-1}$. Third, given the state $\mathbf{z}_i$ at time i, $\mathbf{x}_i$ is independent of the states and the observations at any other time.

Considering all these properties, the joint distribution of a sequence of states and



(a)



(b)

Figure 3.2: Illustration of Markov models: (a) hidden Markov model (HMM) and (b) state transitions in HMM.

observation can be expressed as:

$$p(\mathbf{x}_1, \ldots, \mathbf{x}_N, \mathbf{z}_1, \ldots, \mathbf{z}_N) = p(\mathbf{z}_1)p(\mathbf{x}_1|\mathbf{z}_1)\prod_{i=2}^{N} p(\mathbf{z}_i|\mathbf{z}_{i-1})p(\mathbf{x}_i|\mathbf{z}_i). \tag{3.12}$$

The state graph representation of HMM is shown in Fig. 3.2-a, where the dependencies between the variables are depicted by the arrows.

The HMM models this probability distribution over sequences of observations by specifying a quintuplet $\lambda = (K, \mathcal{O}, \mathbf{A}, \mathbf{B}, \pi)$, where:

- K is the number of possible distinct states in the system, the hidden state variable $\mathbf{z}_i$ is then defined as discrete vector of K elements, in which the values are set to 0, except for $z_i^k = 1$ if $\mathbf{x}_i$ belongs to the k-th state;
- $\mathcal{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_M\}$ is a set of M distinct discrete observation symbols, where $\mathbf{x}_i \in \mathcal{O}$;
- $\mathbf{A} = \{a_{kl}\}_{k,l=1,\ldots,K}$ is the state transition probability matrix of size $K \times K$, where a_{kl} represents the probability of transitioning from state k to state l. This is illustrated in Fig. 3.2-b, where an HMM with $K = 3$ is shown;
- $\mathbf{B} = \{b_{km}\}_{k=1,\ldots,K;m=1,\ldots,M}$ is the observation likelihood matrix of size $K \times M$, where b_{km} represents the probability of observing $\mathbf{o}_m$ at state k.
- $\pi = [\pi_1, \ldots, \pi_K]^T$ is a probability distribution where π_k is the probability that the model starts at state k.

HMM learning involves finding the best HMM parameters $(\mathbf{A}, \mathbf{B}, \pi)$, for a given set of L observation sequences $\mathbb{X} = \{\mathbf{X}_1, \ldots, \mathbf{X}_L\}$ and number of states in the system. For a classification problem, the state space is assumed to be discrete and the number of states K is typically set to be equal to the number of classes in the problem. For discrete observations, it is trivial to construct the observation symbols $\mathcal{O}$ and compute the observation probability $p(\mathbf{x} = \mathbf{o}_m|\mathbf{z})$ from the dataset. As for continuous observations, an additional step to model the observation state space is required (see Section 3.2.4). The optimization process to find the best HMM parameters $(\mathbf{A}, \mathbf{B}, \pi)$ is done by performing expectation maximization (EM) using Baum-Welch algorithm. For a detailed description of the optimization step, we refer the reader to [Bishop 2006].

At test time, given a sequence of observations $\breve{\mathbf{X}}$, an HMM model λ can be used to address two different tasks: inferring the current hidden state and finding the best hidden state path. The former consists of computing the probability that the observations are generated by the model. This is solved by using the forward algorithm [Rabiner 1990], described in Section 3.2.2. The latter task is carried out when the state probabilities for all observations are computed and the objective is to choose the best state path sequence with maximum probability. This is solved using the Viterbi algorithm [Viterbi 1967], described in Section 3.2.3. These aforementioned algorithms are described in the following sections. In terms of surgical activity recognition, the former task corresponds to *online* recognition, while the latter *offline* recognition (see Section 1.3).

3.2.2 Forward Algorithm

Given the HMM model λ, the forward pass recursively computes α_i, defined as the joint probability of $\breve{\mathbf{z}}_i$ and a sequence of observations $\breve{\mathbf{X}} = \breve{\mathbf{x}}_1, \dots, \breve{\mathbf{x}}_i$:

$$\begin{aligned}
\alpha_i(k) &= p(\breve{\mathbf{x}}_1, \dots, \breve{\mathbf{x}}_i, \breve{z}_i^k = 1|\lambda) && (3.13)\\
&= p(\breve{\mathbf{x}}_i|\breve{z}_i^k = 1)\left[\sum_{l=1}^{K} \alpha_{i-1}(k) p(\breve{z}_i^k = 1|\breve{z}_{i-1}^l = 1)\right] && (3.14)\\
\alpha_1(k) &= p(\breve{\mathbf{x}}_1|\breve{z}_1^k = 1) p(\breve{z}_1^k = 1), && (3.15)
\end{aligned}$$

where the term $p(\breve{\mathbf{x}}_1|\breve{z}_1^k = 1)$ can be derived from matrix $\mathbf{B}$, $p(\breve{z}_i^k = 1|\breve{z}_{i-1}^l = 1)$ is defined in matrix $\mathbf{A}$, and $p(\breve{\mathbf{z}}_1)$ is equivalent to π.

The probability of observing the entire sequence is given by the sum over all possible final states:

$$p(\breve{\mathbf{X}}|\lambda) = \sum_{k=1}^{K} \alpha_i(k) \tag{3.16}$$

We thus find that the product of the scaling factors provides us with the total probability for observing the given sequence up to time i and that the scaled probability vector provides us with the probability of being in each state at this time through the equation:

$$p(\breve{z}_i^k = 1|\breve{\mathbf{X}}, \lambda) = \frac{p(\breve{\mathbf{X}}, \breve{z}_i^k = 1|\lambda)}{p(\breve{\mathbf{X}}|\lambda)} \tag{3.17}$$

where the nominator and denominator are given by Eq. 3.14 and 3.16, respectively.

3.2.3 Viterbi Algorithm

The Viterbi algorithm is used to find the best sequence of states $\hat{\mathbf{Z}} = \hat{\mathbf{z}}_1, \ldots, \hat{\mathbf{z}}_i$, given a sequence of observations $\breve{\mathbf{X}} = \breve{\mathbf{x}}_1, \ldots, \breve{\mathbf{x}}_i$ and an HMM model λ, by maximizing $p(\breve{\mathbf{Z}}|\breve{\mathbf{X}}, \lambda)$. It involves the computation of $\delta_i(k)$ which denotes the best score along a single path at time i which accounts for the first i observations and ends at state k. This is defined as:

$$\delta_i(k) = \max_{\breve{\mathbf{Z}}} p(\breve{\mathbf{X}}, \breve{\mathbf{Z}}|\lambda) \tag{3.18}$$

$$= p(\breve{\mathbf{x}}_i|\breve{z}_i^k = 1) \left[\max_{l=1}^{K} \delta_{i-1}(l) p(\breve{z}_i^k = 1|\breve{z}_{i-1}^l = 1) \right] \tag{3.19}$$

$$\delta_1(k) = p(\breve{\mathbf{x}}_1|\breve{z}_1^k = 1) p(\breve{z}_1^k = 1) \tag{3.20}$$

The best path, which ends at state k for the first i observations, is then kept in $\psi_i(k)$ at each step.

Note that the Viterbi algorithm is similar to the forward algorithm, except the maximization over the previous states, instead of a summation. This maximization step makes the Viterbi algorithm return the best path of states.

3.2.4 Continuous Observation

The aforementioned description of HMM assumes that the observations are discrete variables. However, the HMMs can also be extended to accept continuous observation by modeling the distribution of the observed data with a mixture of Gaussians. The observation probability is then given by:

$$p(\mathbf{x}|z^k = 1) = \sum_{m=1}^{M} c_{km} \mathbb{N}(\mathbf{x}, \mu_{km}, \Sigma_{km}), \tag{3.21}$$

where c_{km} and $\mathbb{N}(\mathbf{x}, \mu_{km}, \Sigma_{km})$ are the mixture weight and the probability density function of the m-th Gaussian component, respectively; and μ_{km} and Σ_{km} are the corresponding mean vector and the covariance matrix of the m-th Gaussian component.

3.2.5 Hierarchical HMM

Hierarchical HMMs (HHMMs) are structured multi-level stochastic processed. HHMMs generalize the standard HMMs by making each of the hidden states a probabilistic model on its own, i.e., an HHMM. When a state in an HHMM is activated, the underlying probabilistic model (the sub-HHMM representation) will also then be activated, which in turn will also activate its own probabilistic model (the subsub-HHMM) and so on. This recursive activation process is called the *vertical transition.* The vertical transition is repeated until a special state is reached, called *production state.* In the classical HMM representation, the production state is the one that produces the observations. The states that do not emit observations are called *internal states.* Once the vertical transition is completed, *horizontal transitions* occur between the states in the same level. When

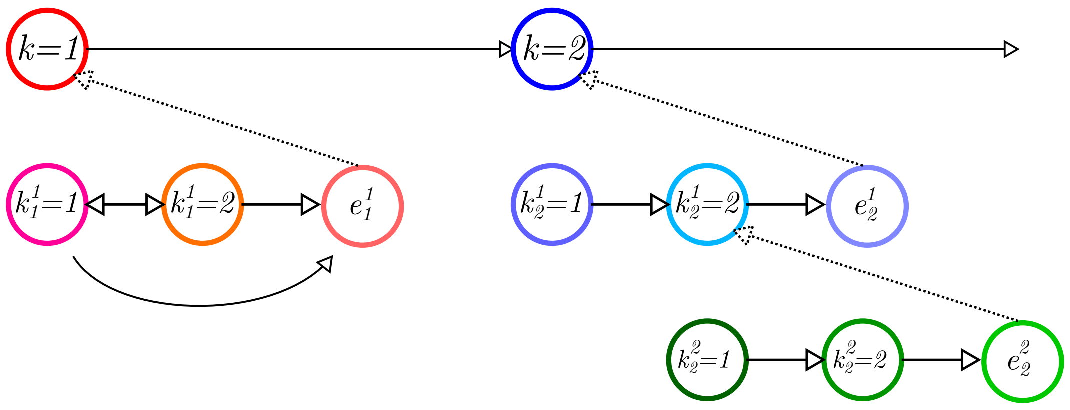


Figure 3.3: Illustration of a three-level hierarchical hidden Markov model. Grey and black edges respectively denote vertical and horizontal transitions. Dashed edged denote returns from the end state of each level to the level's parent state. Note that the production states are not shown in this illustration and only some third-level states are shown, while in practice all the second-level states are represented by a probabilistic model as well.

a horizontal transition leads to a *terminating state*, control returns to the state in the HHMM (in the higher level) that produced the last vertical transition. The graphical model of a three-level HHMM is illustrated in Fig. 3.3.

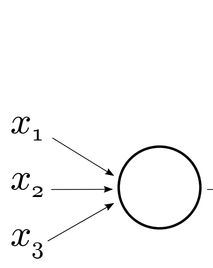
Note that the HMM and HHMM belong to the same class of classifiers, thus they can be used to solve the same set of problems. In fact every HHMM can be represented as a standard single level HMM [Fine 1998]. The states of the HMM are the production states of the corresponding HHMM with a fully-connected structure, meaning that there is a non-zero probability of moving from any of the states to any other state. However, the HHMM utilizes its hierarchical structure to solve a subset of the problems more efficiently. More details of the HHMM can be found in [Fine 1998].

3.3 Deep Neural Networks

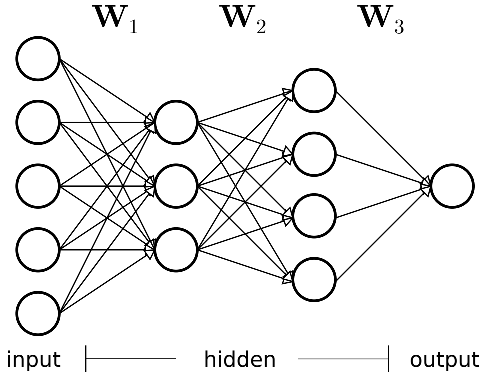
In this section, we discuss neural network techniques. We first describe Artificial Neural Networks (ANNs) in Section 3.3.1. Convolutional neural networks (CNNs), which can be viewed as a special type of ANNs, are later described in Section 3.3.2. In Section 3.3.3, we present a few optimization strategies in order to successfully train the network models. In Section 3.3.4, we show how the neural network approaches can be used as feature learning techniques. Ultimately, we present neural network architectures which are designed to process sequential input in Section 3.3.5.

3.3.1 Artificial Neural Networks

ANNs are inspired by biological neural networks found in brains, where a message is passed from one neuron to another. Similarly, ANNs consist of interconnected neurons (nodes), each of which performs a computation on the input and passes the output to the



(a)



(b)

Figure 3.4: Illustrations of: (a) a perceptron with three inputs and (b) a 4-layered ANN architecture.

next neuron in the system. In Fig. 3.4-a, an illustration of one neuron with three-element input is shown.

3.3.1.1 Formalization

Given a data point $\mathbf{x}$, the computation inside a neuron can be expressed as:

$$z = \mathbf{w}.\mathbf{x} + b \tag{3.22}$$

where $\mathbf{w}$ and b contain the weights and the bias of the neuron, respectively.

In ANNs, these neurons are organized in a layer-wise manner. These layers can be divided into three groups: input, hidden, and output layer. In Fig. 3.4-b, we show an example of ANN architecture, which consists of one input layer, two hidden layers, and one output layer. Such an architecture is typically called a four-layered network. In the literature, ANN architectures containing multiple hidden layers are typically called *deep neural networks* (DNNs), hence the term *deep learning*.

For a network with L layers, the output of the l-th layer of the network, which consists of M_l neurons, denoted as $\mathbf{z}_l \in \mathbb{R}^{M_l}$, is computed by:

$$\mathbf{z}_l = \mathbf{W}_l \cdot h\left(\mathbf{z}_{l-1}\right) + \mathbf{b}_l, \quad 1 \leq i \leq L, \tag{3.23}$$

where $\mathbf{W}_l = [\mathbf{w}_l^1, \ldots, \mathbf{w}_l^{M_l}]^T$ and $\mathbf{b}_l = [b_l^1, \ldots, b_l^{M_l}]^T$ are the weight matrix and the bias vector from the l-th layer, $\mathbf{w}_l^m$ and b_l^m are the weight and bias of the m-th neuron in the l-th layer, $h(\cdot)$ is an activation function (see Section 3.3.1.3), and $\mathbf{z}_0 = \mathbf{x}$ is the input. This computation process is called *forward pass*.

Assuming a binary classification problem and given a finite set of training data $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_N]$ with the labels $\mathbf{y} = [y_1, \ldots, y_N]$, $y_i \in \{0, 1\}$, one can quantify how well the ANN model performs by defining a cost function. A cost function is a non-negative function which becomes smaller as the model performance improves. Once a cost function is defined, the optimization of a network, subsequently, boils down to the minimization of the cost function. There exist several types of cost functions that can be used to quantify the performance of the ANN model; the *cross-entropy* cost function has been shown to converge faster and better than other functions. It is expressed as:

$$C(\mathbf{W}, \mathbf{b}) = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \ln(\sigma(a_i)) + (1-y_i)\ln(1-\sigma(a_i))\right] \tag{3.24}$$

where a_i is the ANN output for data point $\mathbf{x}_i$ and $\sigma(\cdot)$ is a sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}} \tag{3.25}$$

Note that there are two important properties of a cost function: (1) it is non-negative and (2) it tends to zero as the network gets better at computing the desired output. The cross-entropy function is non-negative because all the individual terms in the sum are negative (i.e., $\ln(x) < 0$ for $0 < x < 1$) which are negated by the minus sign out of the sum. As for the second property, the function will be close to zero if the network output $\sigma(a)$ is close to the desired output y. Suppose an example where $y_i = 0$ and $\sigma(a_i) \approx 0$ for some input $\mathbf{x}_i$. We see that the first term in Eq. 3.24 vanishes, while the second term is $\ln(1-\sigma(a_i)) \approx 0$, resulting in the cost for data sample $\mathbf{x}_i$ to be $C_i \approx 0$. A similar analysis holds for $y_i = 1$ and $\sigma(a_i) \approx 1$. Thus, this shows that the cross-entropy function is a proper function to measure the performance of the network.

At test time, to classify a new data $\breve{\mathbf{x}}$, the ANN output $\breve{a}$ is computed. For a binary classification, the new data is classified as the first class if $\sigma(\breve{a}) < d$ and as the second class otherwise, where $d \in (0,1)$ is a threshold dividing the first and second class which can be chosen manually or learnt from the data.

3.3.1.2 Optimization

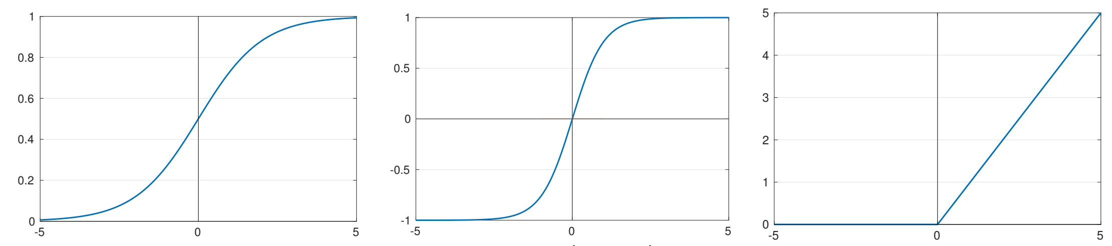
The cost function Eq. 3.24 can be minimized using the gradient descent method which updates the network parameters by computing the Jacobians of the cost function. The update process can be written as:

$$\mathbf{W}' = \mathbf{W} - \eta \cdot \nabla_{\mathbf{W}} C(\mathbf{W}, \mathbf{b}) \tag{3.26}$$

$$\mathbf{b}' = \mathbf{b} - \eta \cdot \nabla_{\mathbf{b}} C(\mathbf{W}, \mathbf{b}) \tag{3.27}$$

where η is the learning rate. The complete detail regarding the Jacobian can be found in [Bishop 2006].

On a network containing L layers, the optimization will start at the last layer ($l = L$) by computing the updates for the weights and biases of the last layer. After $\mathbf{W}_l$ and $\mathbf{b}_l$ are updated, the next step is to compute the update for $\mathbf{W}_{l-1}$ and $\mathbf{b}_{l-1}$ using the same technique. This process is repeated until the very first layer ($l = 1$). This method is called *backpropagation* algorithm because the error is propagated backward through the



$$h(t) = \frac{1}{1 - e^{-t}}$$

(a) logistic sigmoid

$$h(t) = \frac{e^{t} - e^{-t}}{e^{t} + e^{-t}}$$

(b) hyperbolic tangent

$$h(t) = max(0, t)$$

(c) ReLU

Figure 3.5: Illustration of non-linear activation functions.

network during this process.

3.3.1.3 Activation Function

The ANN is capable of modeling any function given enough training points and number of nodes. However, the activation function allows the networks to compute non-trivial functions using fewer number of nodes.

There has been various kinds of non-linear functions proposed as the activation function in ANNs. One of the most common forms of activation function is the sigmoid function, which is a monotonically increasing function that asymptotes at some finite value. The most common examples of sigmoid functions are the standard logistic sigmoid and hyperbolic tangent functions, illustrated in Fig. 3.5-a and 3.5-b, respectively. As can be seen from the figure, the hyperbolic tangent is simply a scaled and shifted version of the logistic sigmoid function. Rectified Linear Unit (ReLU) is also one of the most commonly used activation function, which only allows the dominant neuron outputs to be passed to the next neurons, by basically nullifying all the negative neuron outputs and retaining the positive ones.

3.3.1.4 Multi-Class Problem

For a K-class classification problem, one possible solution is to design the output layer with K nodes, each of which represents the confidence of a data point belonging to the corresponding class. As a result, for data point $\mathbf{x}_i$, the ANN model will give an output $\mathbf{a}_i = [a_i^1, \dots, a_i^K]^T$. The cost function Eq. 3.24 is then modified to:

$$C(\mathbf{W}, \mathbf{b}) = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=1}^{K} y_i^k \ln \left(\sigma \left(a_i^k \right) \right) \tag{3.28}$$

where $y_i^k = 1$ if $\mathbf{x}_i$ belongs to class k, and $y_i^k = 0$, otherwise.

At test time, to classify a new data $\breve{\mathbf{x}}$, the ANN output $\breve{\mathbf{a}}$ is computed. The data is then classified as the k-th class if $\breve{a}^k = max(\breve{\mathbf{a}})$.

3.3.1.5 Multi-Task Problem

In a multi-task problem, the purpose is to learn a model that addresses multiple tasks at the same time by using a shared representation. Multi-task learning is often more advantageous than the independent single-task counterparts because the model is trained using extra information coming from the additional training labels. In addition, the multiple addressed tasks might be correlated to each other, resulting in better parameters in the learnt layers.

To perform multi-task learning for ANNs addressing U tasks, we define a cost function C^u for each task u. Note that each task can be either a single-class or a multi-class problem, thus their corresponding cost function can be expressed by either by Eq. 3.24 or 3.28, respectively. Then the total cost function is then defined by:

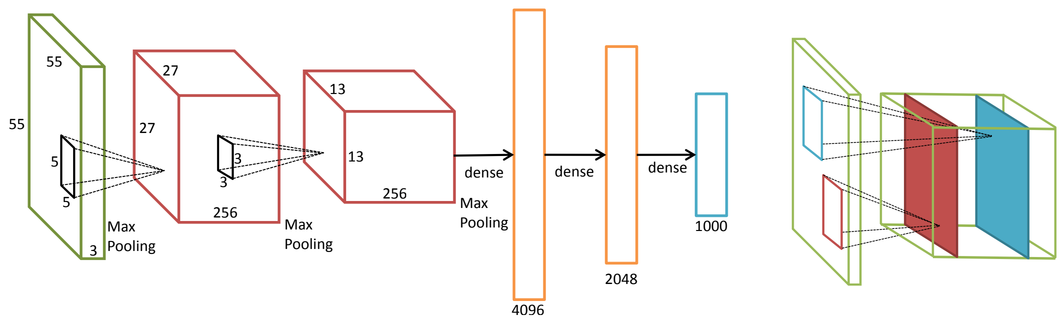
$$C(\mathbf{W}, \mathbf{b}) = \sum_{u=1}^{U} \beta_u C^u(\mathbf{W}, \mathbf{b}), \tag{3.29}$$

where β_u is a scalar multiplier for cost C^u which regulates the trade-off between the cost functions. The back-propagation during training is executed similarly to single-task learning since the Jacobians $\nabla_{\mathbf{W}}C$ and $\nabla_{\mathbf{b}}C$ can be computed straight-forwardly by summing the Jacobians $\nabla_{\mathbf{W}}C^u$ and $\nabla_{\mathbf{b}}C^u$ for all u.

3.3.2 Convolutional Neural Networks

Despite the effectiveness and the interesting properties of ANNs, they are however rather incapable of efficiently handling large inputs, such as images. Assuming the input of the network is small images of size 100×100, the input layer of the ANN will then consist of $10K$ of neurons. The number of parameters that are required to be computed for the first layer is then $M_1 \cdot 10K$. Even with a small number of neurons in the first layer, for instance $M_1 = 1K$, there are already 10 million variables that need to be optimized. This number will add up very quickly as the number of layers increase. In addition, with classical ANNs, the spatial information contained in the image is lost since the networks take the vectorized version of the images. CNNs are proposed to solve these problems that are inherent to ANN models. In the following discussion, we describe in detail the layers which are usually used in CNN architectures.

As a support, in Fig. 3.6-a, we show a CNN architecture example, which takes RGB images of size 55×55 and outputs 1000 values. We can see that at certain layers (the first three), the data is organized in a cube. This particular alignment is essential for the convolution process performed by the convolution layers. Here, we refer the third dimension as *channel*. For instance, the input layer has three channels ($C = 3$) since the network accepts RGB images.



(a) (b)

Figure 3.6: (a) CNN architecture example and (b) illustration of convolution layer.

3.3.2.1 Convolution Layer

The convolution layer is what gives CNNs their interesting properties. Essentially, this layer still performs the same neuron output computation expressed in Eq. 3.22. However, instead of being connected to all the nodes from the input, each of the neuron in this layer is connected only to a local region of the input. For example, in Fig. 3.6-a, the local regions are of size 5×5 and 3×3. Assuming the neurons in the input and output layers are organized in a grid, this operation can be formulated as:

$$\mathbf{z}(u, v) = \Sigma_{c=1}^{C} \left(\mathbf{w}.\mathbf{x}^{c}_{\mathrm{N}_s(u,v)} + b \right), \tag{3.30}$$

where $\mathbf{z}(u, v)$ the output of neuron (u, v) of the layer, $\mathbf{x}^{c}_{\mathrm{N}_s(u,v)}$ is the *receptive field* of the output neuron, i.e., the vectorized values around the neighbourhood of input neuron (u, v) with local region size s at channel c. An interesting property of the convolution layer is that the neuron weight $\mathbf{w}$ is the same for all output neurons, resulting in a huge reduction in the number of unknowns to estimate during the learning process. Conceptually, this is equal to performing convolution operations using the neuron weight as a filter on the input, hence the name *convolution* layer and the weights $\mathbf{w}$ referred to as a *filter*. Typically, one convolution layer contains multiple filters of same size $s \times s$. In Fig. 3.6-b, we show how the results of the convolution operations using multiple filters are stacked along the channel axis (depicted in blue and red rectangles).

One should note that convolution operation is one of the most used operations in image processing techniques, especially in detecting dominant features (e.g, edge and corner detections). The CNNs are performing similar operations on the images to extract discriminative and dominant features from the images. The advantage of using CNNs is the fact that the values inside the filters are automatically learnt from the data; unlike in other feature detectors where the filters are typically pre-defined.

3.3.2.2 Pooling Layer

A pooling layer is a form of non-linear downsampling. Max pooling is the most common function used to execute the pooling in CNN architectures. It partitions the input into subregions and outputs the maximum value for each sub region. The main function of pooling layer is to reduce the amount of parameters in the network, thus speeding up the optimization process. It is common to apply convolution and pooling layers one after another in an alternating manner. Max pooling after the convolution process performs dimensionality reduction while retaining the most dominant features on the image detected by the filters.

To better illustrate how the convolution layer and the max pooling layer works, let us refer to the example shown in Fig. 3.6-a. The input layer takes color images of spatial size 55×55. The image is then passed to a convolutional layer consisting of 256 filters, resulting in an output of size $55 \times 55 \times 256$. The spatial size is retained because the image borders are padded before the convolution operations. Similarly to the ANN counterpart, this output is also typically passed to a non-linear activation function. Then, the output is pooled by the pooling layer of size 3×3 with a stride of 2 pixels, thus reducing the output to $27 \times 27 \times 256$. The second layer performs the same set of operations, i.e., convolution (with 256 filters of size 3×3), non-linear activation, and max-pooling. This layer yields an output of size $13 \times 13 \times 256$.

3.3.2.3 Fully-Connected Layer

Fully-connected layers are the typical neuron connections found in ANN architectures. They are called fully-connected layers since all the input are connected to all the neurons inside the output layer. The fully connected layers are shown in orange and blue in Fig. 3.6-a. Similarly to the ANN architectures, the classification process in CNN architectures is also performed using the fully-connected layers. The multi-class problem is addressed in CNNs using the same method presented in Section 3.3.1.4.

3.3.3 Optimization Strategy

Learning an ANN or a CNN model can become a tedious process since there are a high number of unknown variables subject to the optimization process. In this section, we will discuss a few of optimization strategies that are applied in our methods. For a complete list and detailed description of the strategies, we refer the reader to [LeCun 1998] and [Krizhevsky 2012].

3.3.3.1 Batch vs. Stochastic vs. Mini-Batch Learning

Notice that the cost functions (Eq. 3.24 and 3.28) have the form of $C = \frac{1}{N}\sum_{i=1}^{N} C_i$, which is an average over costs C_i for each training sample $\mathbf{x}_i$. This means that to update the parameters, one needs to compute the Jacobians $\nabla_{\mathbf{W}} C$ and $\nabla_{\mathbf{b}} C$ over the complete training data. Such an approach is called *batch learning*. Unfortunately, when the number of training samples is high, the Jacobians can take a long time to compute, therefore slowing down the learning process.

In contrast, *stochastic gradient descent* (SGD) approach performs the parameter update for *each* training sample $\mathbf{x}_i$, resulting in $C = C_i$. Thanks to this mechanism, it is frequent that SGD converges much faster than batch learning. Since the Jacobians are computed only using one data sample at each parameter update, this makes the process to be highly sensitive to noise and outliers. As a result, the cost function is usually observed to fluctuate during the optimization process. On the one hand, these fluctuations might allow jumps to new and potentially better local minima. On the other hand, the fluctuations might also cause SGD to overshoot and thus subsequently miss the optimal solution.

In order to take advantage of the benefits from both the batch learning and SGD approach, *mini-batch learning* performs the parameter update using a subset of the training data, i.e., n randomly chosen training samples, referred to as *batch*. The cost can then be written as the average over these random samples: $C = \frac{1}{n}\sum_{i=1}^{n} C_i$. Provided the sample size n is large enough, it is assumed that the average value from the mini-batch will be roughly equal to the value given by the batch learning approach. By doing so, the approach can perform as well as the batch learning approach and comparably as fast as the SGD approach.

3.3.3.2 Shuffling Samples

Networks learn faster with the most unexpected samples because the gradient will be higher for data that generates high error. A large error indicates that the input has not been learned by the network, thus it contains more valuable information than other data. There is, however, no simple way to know which inputs contain more information than the other. A very simple trick is to shuffle the training samples, so that the successive batches come from different classes. This mechanism makes sure that the network observes enough variation in each batch, resulting in faster learning and less sensitivity to overfitting.

3.3.3.3 Choice of Activation Function

In terms of sigmoid functions, it is often the case that the hyperbolic tangent converges faster than the logistic sigmoid function [LeCun 1998]. In addition, the hyperbolic tangent is also preferable since it is symmetric on the origin (see Fig. 3.5-b), resulting in outputs that are more likely to have a 0-mean, unlike the logistic sigmoid function which always returns positive outputs.

In a recent work [Krizhevsky 2012], the rectified linear unit (ReLU) is shown to perform very well for a CNN model, resulting in a much faster optimization process compared to the sigmoid type functions. Fast learning is particularly important while learning a large model using a large dataset. This function is shown in Fig. 3.5-c.

As for the pooling layers, it generally divides the input into non-overlapping subregions. However, this leads to an aggressive reduction in the size of the input, which might cause information loss during the learning process. Thus, overlapping pooling layer is used to prevent such loss. In [Krizhevsky 2012], it has been shown that using an overlapping max-pooling layer yields a significant performance improvement. The CNN illustration presented in Fig. 3.6-a also uses the overlapping max-pooling layer.

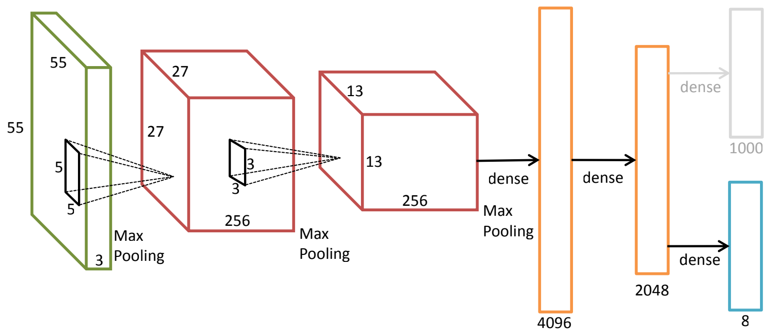


Figure 3.7: Illustration of a network fine-tuned using the network shown in Fig. 3.6-a. The last layer is replaced with a fully-connected layer containing 8 nodes.

3.3.3.4 Dropout

A large neural network is prone to overfitting, especially when it is trained on a small dataset. In [Hinton 2012, Srivastava 2014], a method, called *dropout*, was introduced to address this problem. It consists of setting the output of each hidden neuron to zero for each iteration, with 50% probability. Therefore, these dropped-out neurons do not contribute in both forward pass and backpropagation. In [Krizhevsky 2012, Szegedy 2015], dropout is applied after the fully-connected layers. Despite slowing down the convergence of the network training, it is shown to eliminate overfitting from the learnt model.

3.3.4 Transfer Learning

Up to this point, we have presented one of the approaches used to address a classification task using neural network solutions, which involves training the model from scratch and perform the task using the learnt model. However, it is very hard to optimize neural network models which contain a lot of of unknowns, for example the *AlexNet* architecture introduced in [Krizhevsky 2012] to perform image classification contains roughly 60 millions parameters. Despite the aforementioned strategies and the success of previous works, a lot of labeled data and a high computational power are still required to optimize deep network models. Therefore, training a network from scratch is not always a feasible solution for all classification tasks, especially when labeled data is scarce. In order to still be able to benefit from the advantages of neural network approaches, one can resort to *transfer learning* techniques. There are two possible transfer learning approaches and both of them rely on *pre-trained* networks that have been shown to successfully perform other tasks.

The idea of the **first** solution stems from the notion that considering the last layer in a neural network model is typically designed to perform the classification task, the outputs of the second last layer can then be perceived as features extracted from the data. It has been shown in [Yosinski 2015] that as the network goes deeper, higher level features are learnt by the neural network. In deep CNN models, the first few convolution layers typically detect low-level features, such as edges, while the last layers detect more

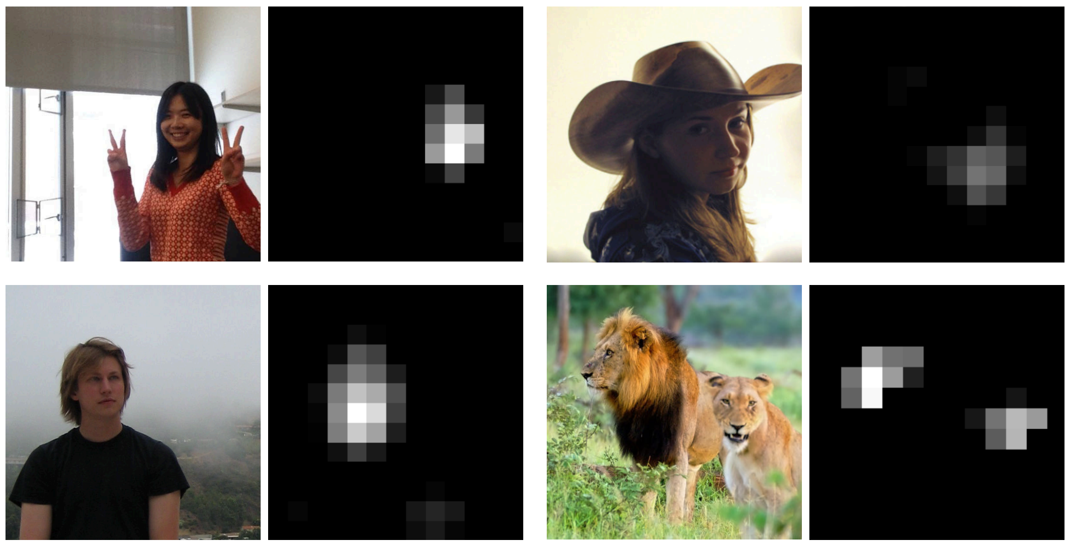


Figure 3.8: Visualization of the activations from a channel of the last convolution layer of the AlexNet network. Courtesy of [Yosinski 2015].

semantically meaningful structures. For example, shown in Fig. 3.8, the last convolution layer of the AlexNet network shows high activations on the image parts which contain faces, despite the fact that the training dataset does not contain a class labeled as "face" or any explicit notion of "face". This shows that in addition to performing classification, the pre-trained deep neural network models can also be used to extract visual features.

Since a neural network can be perceived as a feature extractor, any neural network models pre-trained using a *source* dataset can then be used to extract features from any data in a *target* dataset (given the data is of the same format as the one in the source dataset). Once the features are extracted, any classification methods (e.g., SVM and HMM) can be used on top of the features to perform the task. Since the way how the features are computed is automatically learnt during the training process of the network, this process is often referred to as *feature learning*. For example, it has been shown in [Donahue 2013] that the visual features extracted using the AlexNet network can be used to perform various computer vision tasks, such as scene classification and domain adaptation.

In the first solution, the network is not explicitly trained to extract features from the target dataset. Intuitively, a network should generate better features for a dataset if the network has been trained on the dataset. Hence in the **second** solution, one can perform *finetuning* on the pre-trained network if some labeled data is available in the target dataset. The finetuning process involves slight modifications to the parameters in the pre-trained network, so that it extracts more suitable features from the target dataset and thus performs the current task at hand better than the pre-trained network. First, the network architecture needs to be modified to accommodate the current task. For example, if one is to use the pre-trained network shown in Fig. 3.6-a and the current

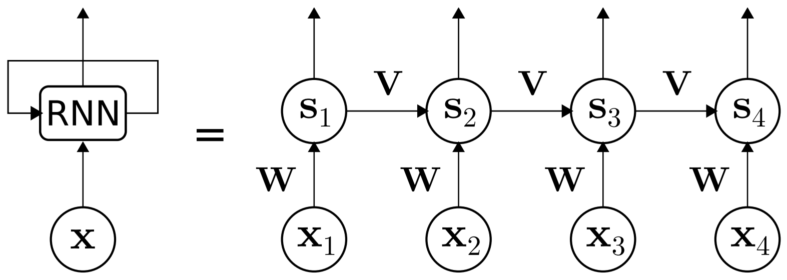


Figure 3.9: Unrolled recurrent neural network (RNN). The number of unrolled elements corresponds to the length of the input sequence. In this case, we assume a sequence of length 4.

task is an 8-class problem, the last layer of the network has to be replaced with a new fully-connected layer containing 8 nodes. This results in a new CNN architecture shown in Fig. 3.7. Later, in the training process, instead of using random initializations, the parameters in the network are initialized with the weights of the pre-trained network, rendering the optimization process easier than training the network from scratch. Since the initialization is assumed to be close to the desired solution, the network parameters should not be changed drastically during the training process. Thus in a fine-tuning process, the learning rate η is usually set at a lower value.

3.3.5 Neural Networks for Sequential Data

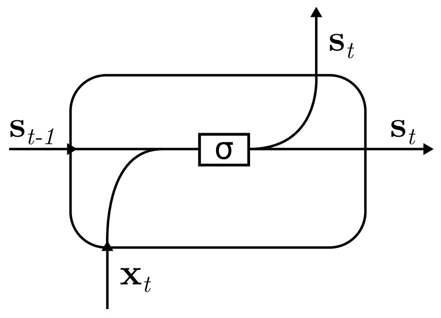
Neural networks have been shown to be a powerful tool in solving computer vision problems. However, the traditional neural network approaches assume that the image inputs (and classification outputs) are independent of each other. This leads to a shortcoming of the inability to model sequential information in the network. In the following, we discuss neural networks which process sequential data as input: Recurrent Neural Networks (RNNs) and Long Short Term Memory (LSTM) architectures.

3.3.5.1 Recurrent Neural Networks

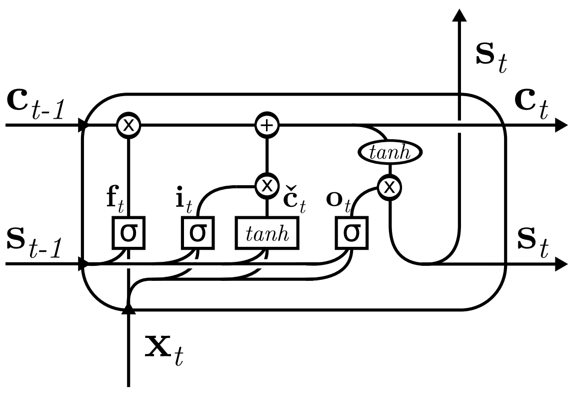
Recurrent Neural Networks (RNNs) are networks that contain loops which allow information to pass along the sequence. This information is designed to contain the information of what the network has seen so far. In Fig. 3.9, we show an example of an unrolled RNN for an input sequence of length 4, where:

- $\mathbf{x}_t$ is the input at time t;
- $\mathbf{s}_t = h\left(\mathbf{W} \cdot \mathbf{x}_t + \mathbf{V} \cdot \mathbf{s}_{t-1} + \mathbf{b}\right)$ is the output at time t: h is a non-linear activation function (e.g., logistic sigmoid σ), $\mathbf{W}$ and $\mathbf{V}$ are the parameters of the RNN, and the initial state $\mathbf{s}_0$ is typically initialized to zero. Here, the output $\mathbf{s}_t$ is also the information passed along the sequence.

Training RNNs boils down to finding the optimal $\mathbf{W}$ and $\mathbf{V}$, which can be solved using backpropagation.



(a)



(b)

Figure 3.10: Internal structures of: (a) RNN and (b) LSTM units.

In theory, RNNs should be able to retain information that has been seen along the sequence. However, as the length of the sequence grows, RNNs tend to lose long-term context. In [Bengio 1994], it is shown that this is due to the fact that as the sequence grows, the gradients tend to vanish/explode. As a result, it is very difficult for RNNs to retain long-term context. This is however not the case for long-short term memory (LSTM) networks [Hochreiter 1997].

3.3.5.2 Long Short Term Memory

LSTM networks are a special kind of RNNs designed to avoid long-term dependence problems by incorporating memory units that explicitly allow the network to learn when to forget and keep the information in the state. In Fig. 3.10, we show the difference between the internal structures of simple RNN and LSTM units, where cell state and output at time t are denoted respectively by $\mathbf{c}_t$ and $\mathbf{s}_t$. Generally, a LSTM unit can be defined by the following set of equations:

- forget gate $\mathbf{f}_t = \sigma\left(\mathbf{W}_f \cdot \mathbf{x}_t + \mathbf{V}_f \cdot \mathbf{s}_{t-1} + \mathbf{b}_f\right)$. The forget gate is responsible for deciding which information is going to be thrown away from memory;

- input gate $\mathbf{i}_t = \sigma\left(\mathbf{W}_i \cdot \mathbf{x}_t + \mathbf{V}_i \cdot \mathbf{s}_{t-1} + \mathbf{b}_i\right)$. In contrast to the forget gate, the input gate decides which values will be updated;

- input modulation gate $\check{\mathbf{c}}_t = \mathit{tanh}\left(\mathbf{W}_c \cdot \mathbf{x}_t + \mathbf{V}_c \cdot \mathbf{s}_{t-1} + \mathbf{b}_c\right)$. This gate creates a vector of new candidate values to be added to the cell state;

- cell state update $\mathbf{c}_t = \mathbf{f}_t \circ \mathbf{c}_{t-1} + \mathbf{i}_t \circ \check{\mathbf{c}}_t$, where $\circ$ is an element-wise vector multiplication operator. This is done by using the previous cell state values and the values computed from the input and output gates;

- output gate $\mathbf{o}_t = \sigma\left(\mathbf{W}_o \cdot \mathbf{x}_t + \mathbf{V}_o \cdot \mathbf{s}_{t-1} + \mathbf{b}_o\right)$. The output gate decides which values need to be passed as output; and

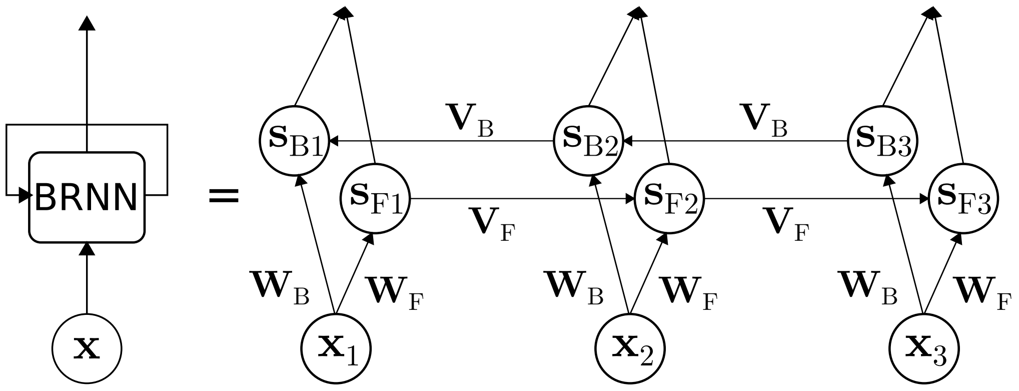

BRNN
=
$\mathbf{x}$
$\mathbf{s}_{\mathrm{B}1}$
$\mathbf{s}_{\mathrm{B}2}$
$\mathbf{s}_{\mathrm{B}3}$
$\mathbf{s}_{\mathrm{F}1}$
$\mathbf{s}_{\mathrm{F}2}$
$\mathbf{s}_{\mathrm{F}3}$
$\mathbf{V}_{\mathrm{B}}$
$\mathbf{V}_{\mathrm{B}}$
$\mathbf{V}_{\mathrm{F}}$
$\mathbf{V}_{\mathrm{F}}$
$\mathbf{W}_{\mathrm{B}}$
$\mathbf{W}_{\mathrm{F}}$
$\mathbf{W}_{\mathrm{B}}$
$\mathbf{W}_{\mathrm{F}}$
$\mathbf{W}_{\mathrm{B}}$
$\mathbf{W}_{\mathrm{F}}$
$\mathbf{x}_1$
$\mathbf{x}_2$
$\mathbf{x}_3$

Figure 3.11: Unrolled bidirectional recurrent neural network (Bi-RNN). The number of unrolled elements corresponds to the length of the input sequence. In this case, we assume a sequence of length 3.

- output computation $\mathbf{s}_t = \mathbf{o}_t \circ tanh(\mathbf{c}_t)$. Through a $tanh$ function, the values stored in the current cell state are filtered to the output.

Training LSTM networks boils down to finding the optimal $\mathbf{W}_{f,i,o}$ and $\mathbf{V}_{f,i,o}$, which can be solved using backpropagation.

3.3.5.3 Bidirectional Network

In Fig. 3.9, we can see that the state passage in RNN follows the direction of the forward flow of time. This indicates that the future signals, which also contain discriminative information regarding the current state, are not exploited in this type of network. To incorporate the information from the future signals, a bidirectional RNN (Bi-RNN) is proposed in [Schuster 1997].

In Fig. 3.11, we show an example of an unrolled Bi-RNN architecture for an input sequence of length 3, which consists of:

- $\mathbf{x}_t$ is the input at time t;
- forward flow computation $\mathbf{s}_{Ft} = h\left(\mathbf{W}_F \cdot \mathbf{x}_t + \mathbf{V}_F \cdot \mathbf{s}_{t-1} + \mathbf{b}_F\right)$ and backward flow computation $\mathbf{s}_{Bt} = h\left(\mathbf{W}_B \cdot \mathbf{x}_t + \mathbf{V}_B \cdot \mathbf{s}_{t-1} + \mathbf{b}_B\right)$ at time t. Ultimately, the final output $\mathbf{s}_t$ is acquired by combining combination of forward and backward computation, which can be obtained through various methods, such as average pooling and concatenation.

Using the same principle, the bidirectional architecture can also be applied to LSTM networks, resulting in bidirectional LSTM (Bi-LSTM) networks.

4 Surgical Activity Recognition on Laparoscopic Videos

The problems are solved, not by giving new information,
but by arranging what we have known since long.
– Ludwig Wittgenstein

Chapter Summary

In this chapter, we present recognition pipelines using handcrafted visual features to address surgical activity recognition tasks on laparoscopic videos at two different granularity levels. First, we introduce in Section 4.2 a pre-segmented surgical activity recognition task at the coarsest level of granularity, i.e., identifying the surgical procedure performed in a laparoscopic video. We perform this task as an initial experiment to investigate the discriminative characteristics of handcrafted visual features for surgical activity recognition on laparoscopic videos. Then, we present in Section 4.3 a frame-wise activity recognition task, i.e., recognizing surgical phases in cholecystectomy procedures.

We perform extensive experiments for both tasks to generate strong baselines that will be used in Chapter 6.

4.1 Dataset Generation

Thanks to a close collaboration with the surgeons at the University Hospital of Strasbourg/IRCAD, we are able to access a video database which stores the recordings of various laparoscopic procedures. In this database, all videos are recorded at 25 fps.

For surgery classification, we generate a dataset of 208 videos labeled with their surgery type. For surgical phase recognition, we generate a dataset of 80 cholecystectomy videos labeled with the surgical phases as well as the tool presence information. The annotation process was facilitated by an annotation software, developed by research group CAMMA[1]. Further details about the datasets can be found in the following sections and in Appendix A.

4.2 Surgery Classification

Automatic surgical video database indexing is one of the applications that benefits from surgical activity recognition. Using the database indices, users can easily navigate inside the database without having to browse the videos one by one. We take here the first step toward this direction by performing laparoscopic surgery classification, which consists of automatically identifying the type of surgery performed in the video. The automatic surgery classification is particularly important to deal with a large amount of laparoscopic videos that are not labeled.

To the best of our knowledge, this is the first work to address such a task in the literature. The surgery classification task is however not a trivial task. In addition to the visual challenges inherent to laparoscopic videos (e.g., rapid camera motion and specular reflection), the videos from different procedures look highly similar to one another. This results in a low inter-class variability which is challenging for any classification problem. Here, we study the feasibility of performing the surgery classification task on laparoscopic videos using handcrafted visual features.

Ideally, the surgery classification task is performed on a complete recording of the procedure. However, it is frequent that the complete recordings of surgeries are not easily accessible, but video clips (partial recordings) are typically publicly available. Thus, in addition to performing surgery classification on complete videos, we also perform *video clip classification* to analyze the feasibility of doing surgery classification on video clips. More specifically, we subsample the videos by taking the beginning, the middle and the end parts of the videos. Then, we perform the surgery classification using these sub-parts.

The length of laparoscopic videos vary significantly depending on multiple factors, such as the expertise of the surgeon and the surgical team, and the pathology of the patients. At times, the laparoscopic videos can be more than one hour long, which can

[1]http://camma.u-strasbg.fr/

impede the scalability of the method. To study the scalability, we perform *downsampled classification* by gradually downsampling the video frames down to 1% of the whole video (i.e. at 0.2 fps). Considering that the laparoscopic video database contains hundreds of videos, this is a crucial point to observe in order to reduce the cost of feature computation without impeding the classification performance.

4.2.1 Methodology

We propose a classification pipeline to perform surgery classification, which consists of four steps: frame rejection, visual feature extraction, feature encoding, and classification.

4.2.1.1 Irrelevant Frame Rejection

During laparoscopic procedures, the endoscope is often taken out of the patient's body, typically to clean the lens of the endoscope. When this happens, the endoscope points to arbitrary directions, resulting in frames which contain irrelevant, if not misleading, information about the surgery. In addition, the action of taking the camera out of the body is recurring along the procedure. The videos also often contain blank or static frames when the camera is disconnected from the recording device. As the amount of such irrelevant frames in a video increases, the feature representation of the video becomes less meaningful. Thus, a method to compute the relevance of video frames has to be incorporated in the pipeline.

In [Atasoy 2012], an approach to label informative frames for gastro-intestinal endoscopy is presented. The semi-supervised method is based on the K-means clustering on the energy histogram from the frequency domain and requires an expert to label the clusters as informative or non-informative frames. In a more recent work [Muenzer 2013], a supervised method using color-based features to identify relevant frames from laparoscopic videos is presented. However, both of these methods are supervised and thus require annotations from experts.

To avoid the requirements of labeling relevant and irrelevant frames, here, we propose an unsupervised method to reliably reject the irrelevant frames from the videos using a straightforward RGB histogram thresholding. It can observed that when the laparoscope is inside the patient, the red color channel is particularly more dominant compared to the other color channels due to the nature of the anatomical structures. With that observation, a scalar value is computed using a base encoding approach to represent each color channel histogram. This is written as a dot product $S_c = [1, 2, \ldots, 256]^T \cdot \mathbf{h}_c$ where S_c is the scalar representation and $\mathbf{h}_c$ is the 256-bin histogram for each color channel c. Ultimately, we obtain S_R, S_G, and S_B from red, green, and blue channels of the video frames. Three thresholds are then set to carry out the frame rejection task: the minimum threshold $\underline{S_R}$ for the level of "redness", and the lower bound $\underline{I}$ and the upper bound $\bar{I}$ of the brightness level. The frame rejection is defined as:

$$f = \begin{cases} 1 & \text{, if } S_R > \underline{S}_R, \text{ and } \underline{I} < S_R, S_G, S_B < \bar{I}, \\ 0 & \text{, otherwise,} \end{cases} \tag{4.1}$$

which returns 1 for relevant frames. These thresholds are set empirically based on preliminary observations on a few videos.

4.2.1.2 Visual Feature Extraction

To obtain the feature representation of the videos, we propose to extract spatial visual features from the video frames. We do not use spatio-temporal features, such as spatio-temporal interest points, since the laparoscopic videos are recorded with a rapidly moving endoscope. The visual features belong to three feature groups: color information, salient points and image gradients.
Color Information. We pick color histograms since they have been used in the literature to perform surgical activity recognition [Lalys 2012, Dergachyova 2016]. We extract histograms from two color channels: (1) Red-Green-Blue histograms from the RGB color space with 3×16 bin size and (2) Hue-Saturation histograms from the HSV color space with 2×36 bin size.
Salient Points. We extract the scale-invariant feature transform (SIFT) [Lowe 2004] because of its scale- and rotation-invariant characteristics. Each of the detected SIFT key points are described with 128-dimensional feature vector.
Image Gradients. We use the Histogram of Oriented Gradients (HOG) [Dalal 2005], which counts occurrences of gradient orientation in a dense grid of uniformly spaced cells over the image. In our setup, a HOG feature vector of dimension 288 is extracted from each video frame.

4.2.1.3 Feature Encoding

The aforementioned visual features are extracted from the video frames. Since the task consists of labeling the videos with surgery labels, a feature encoding method is required to aggregate the visual features coming from the frames to ultimately represent the laparoscopic videos. Here, we use the Bag-of-Word (BOW) approach to perform the feature encoding.

The main idea of the BOW approach is to treat the video as a histogram of visual words. The visual words are defined by the K-means centers obtained during the construction of the visual dictionaries using K-means clustering. Every visual feature vector $\mathbf{f} \in \mathbb{R}^M$ will be expressed in a sparse representation $\mathbf{s} \in \mathbb{R}^K$ that is computed using the learnt dictionary $\mathbf{D} \in \mathbb{R}^{M \times K}$ by solving

$$\mathbf{s}^* = \underset{\mathbf{s}}{\operatorname{argmin}} \|\mathbf{f} - \mathbf{D}\mathbf{s}\|_2 \quad \text{subject to} \quad \|\mathbf{s}\|_0 = 1, \|\mathbf{s}\|_1 = 1, \tag{4.2}$$

where $\|.\|_p$ computes the l_p-norm of the vector. The constraints basically suggest that

	Surgery	#Vid	Avg. length (min.)
1	Sigmoidectomy	26	113 ± 41
2	Eventration	28	73 ± 52
3	Bypass	36	119 ± 29
4	Hernia	34	56 ± 28
5	Cholecystectomy	36	65 ± 36
6	Nissen Gerd	22	94 ± 33
7	Adrenalectomy	11	108 ± 346
8	Sleeve Gastrectomy	15	91 ± 31

Table 4.1: List of the abdominal surgery classes in our dataset along with their corresponding number of videos and average video length (± std) in minutes.

the vector $\mathbf{s}$ can only contain one non-zero element which is equal to 1. For each feature type, a video is then represented by a normalized histogram $\mathbf{x} = \frac{\sum_{i=1}^{N} \mathbf{s}_i}{\|\sum_{i=1}^{N} \mathbf{s}_i\|_2}$ of size K.

4.2.1.4 Classification

After the feature encoding process, every laparoscopic video V_i will be represented with a final feature vector $\mathbf{x}_i$. A collection of feature vectors $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ of N laparoscopic videos from the dataset is used to train and test the classification model. Here, we use one-versus-all Support Vector Machines (SVMs) with three different kernels: linear, chi-square, and histogram intersection.

Later, we use the Generalized Multiple Kernel Learning (GMKL) [Varma 2009] to combine the visual features to build a better classification model. From each feature, we take the kernel setup yielding the best performance. The GMKL will then learn the best possible kernel combination. GMKL is particularly superior to other multiple kernel learning approaches because it can learn more general kernel combinations, in contrast to the traditional MKL approaches that focus on learning a linear combination of the given base kernels.

4.2.2 Experimental Setup

4.2.2.1 Laparo208 Dataset

To evaluate the classification pipeline, we generate a large dataset containing 208 video recordings of 8 different types of surgeries, referred to as *Laparo208*. These surgeries are chosen because they are the 8-most frequent laparoscopic procedures in our partner hospital. These videos are recorded in the department of abdominal surgery at the University Hospital of Strasbourg/IRCAD. All videos are recorded at 25 fps. The surgical procedures were performed by 10 different surgeons. In total, they sum up to more than 300 hours or approximately $25M$ frames. The statistics of the dataset are shown in Table 4.1.

Feature	Dim.	# Words
SIFT	128	$500, 1\text{K}, 1.5\text{K}$
HOG	288	
RGB	48	$100, 300, 500$
HS	72	

Table 4.2: The configuration of dictionary learning for every feature.

4.2.2.2 Feature Encoding

Before the feature extraction process, the videos are downsampled to 1 fps due to redundant information and for faster processing. We take one video randomly from every class and extract the features from these videos to build the visual dictionaries. The parameters of the visual dictionaries are shown in Table 4.2.

4.2.2.3 Classification

In our experiments, we use linear and non-linear classifiers for K-means and K-SVD feature encodings. The non-linear kernels are Chi-Square (χ^2) and histogram intersection (HI). We compute the kernels and learn the classification models using the VLFeat toolbox [Vedaldi 2010]. We observe the accuracy of our method by using the leave-one-out evaluation.

4.2.2.4 Downsampled Classification

The downsampled classification is performed to investigate the scalability of the approach in performing the surgery classification task. For every video V_i which has been downsampled to 1 fps, we have N_i frames. To carry out the downsampled classification, we assign a new number of frames $\breve{N}_i \in \left\{\frac{1}{5}N_i, \frac{2}{5}N_i, \frac{3}{5}N_i, \frac{4}{5}N_i, N_i\right\}$ and downsample the frames uniformly along the videos. For instance, for $\breve{N}_i = \frac{2}{5}N_i$, we only take two frames out of every five consecutive frames from which the visual features are extracted. We train a new classification model and run a leave-one-out validation for every $\breve{N}_i$. Ultimately, we will have visual features which are extracted at 0.2, 0.4, 0.6, 0.8, and 1 fps. The illustration of the downsampling for the downsampled classification is shown in Figure 4.1.

4.2.2.5 Video Clip Classification

The video clip classification is performed to see which part of the videos is the most discriminative. Similarly to the downsampled classification, we have a set of decreasing numbers of frames $\breve{N}_i \in \left\{\frac{1}{5}N_i, \frac{2}{5}N_i, \frac{3}{5}N_i, \frac{4}{5}N_i, N_i\right\}$. The difference is that, in this setup, we take the video frames based on their position, i.e. beginning, middle, and end. For example, in the setup of classifying the beginning of the videos with $\breve{N}_i = \frac{4}{5}N_i$, we take the first 80% of the frames to represent V_i. Similar process also applies when we use the end of the videos, i.e. we take the last 80% of the frames to represent V_i. For the middle configuration, we take randomly the middle part of the videos. The illustration of the

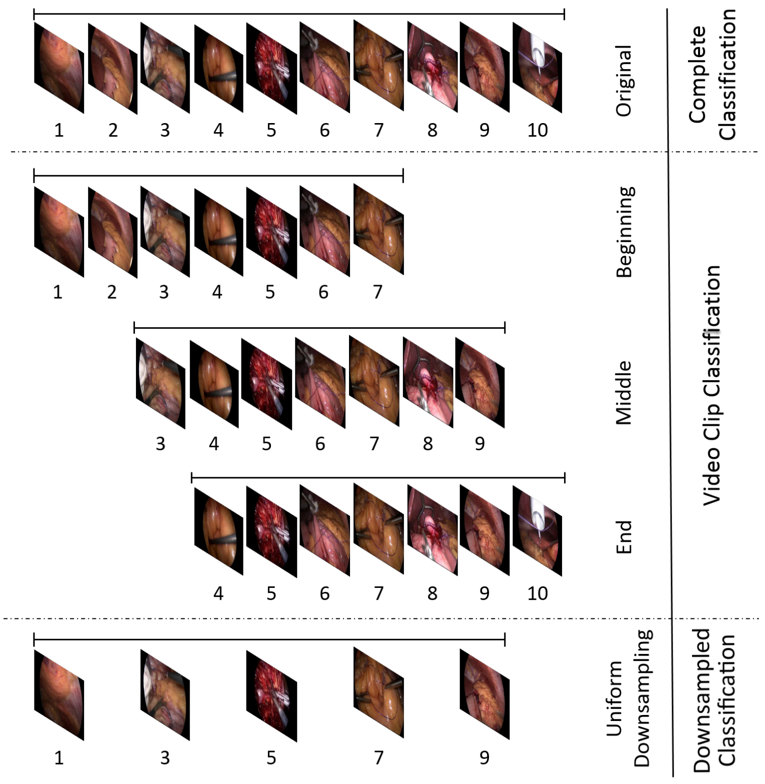


Figure 4.1: Illustration of the subsampling for the video clip and the downsampled classifications.

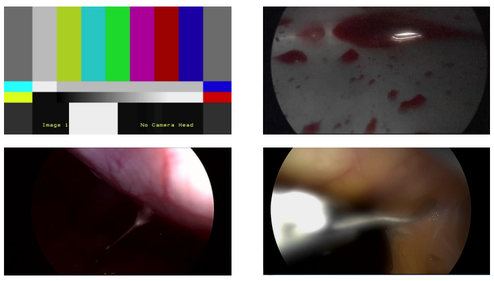
video clip extraction is shown in Figure 4.1.

4.2.3 Experimental Results

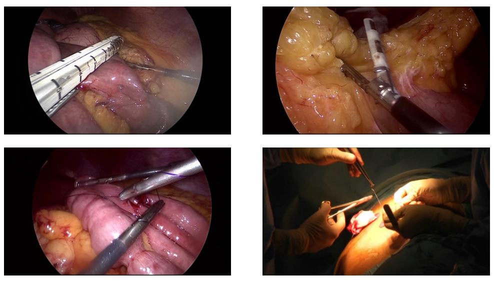
4.2.3.1 Irrelevant Frame Rejection

Before presenting the surgery classification results, we first analyze the effectiveness of the frame rejection method. In Fig. 4.2-a, we show image samples which are identified as irrelevant. Despite the fact that they are captured when the camera is inside the patient's body, the frames at the bottom row of Fig. 4.2-a are either too dark or too bright and do not contain any significant information regarding the surgery. Thus, the rejection of such frames should not result in any information loss in the final feature representation of the videos. In addition, this also shows that the threshold can also be used to reject such frames that have high or low brightness.

In Fig. 4.2-b, we show the frames that are identified as relevant by our method. We can see that the frame is being classified as relevant even though it captures the scene outside of the body. Most probably, this is because of the "redness" level of the frame which resembles to the redness level of the relevant frames (shown by other images in Fig. 4.2-b). However, such frames are only included in videos which are recorded for training

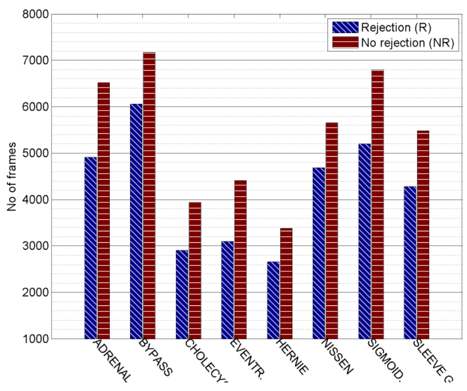


(a)

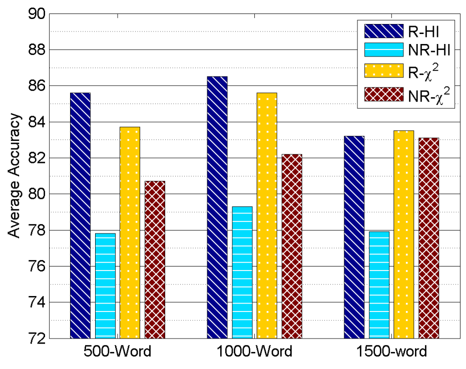


(b)

Figure 4.2: Frame rejection results: frames that are classified as (a) irrelevant and (b) relevant.



(a)

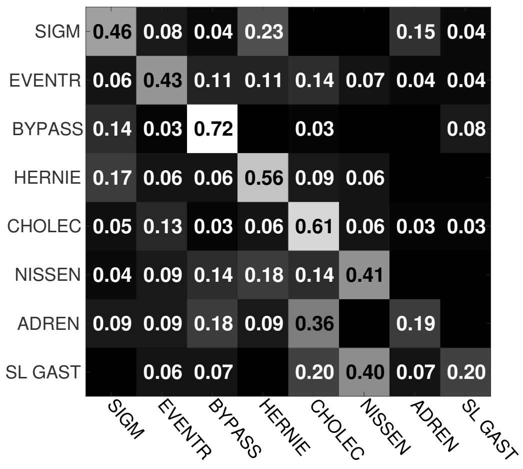


(b)

Figure 4.3: Graphs showing the comparisons of: (a) the average numbers of video frames when the frame rejection step is used (R) and unused (NR), and (b) accuracies of non-linear classifiers (HI and χ^2) built with and without frame rejection on SIFT.

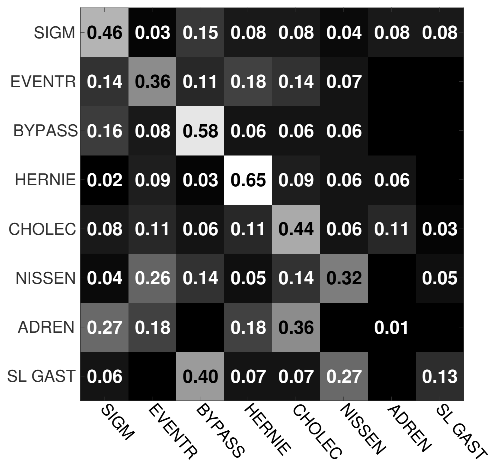
purposes. These videos rarely appear in the database, thus the information representation of the model should remain meaningful.

In Fig. 4.3-a, we show the number of frames processed by the pipeline with and without the frame rejection step. It can be seen that the frame rejection step decreases the number of frames that need to be processed by 1.2K frames on average, which is equivalent to approximately 20 minutes per recording. Subsequently, this results in a much faster feature computation process and reduces the accumulated memory footprint. Furthermore, the frame rejection step also improves the classification results. In Figure 4.3-b, we show the accuracy comparison of classifications without and with frame rejection. These results are obtained using the non-linear classifiers on the SIFT features. By rejecting the irrelevant frames from the feature extraction process, the classification accuracies are increased by up to 7%. Seeing the improvement, we then carry out the rest of the experiments using the frame rejection method.



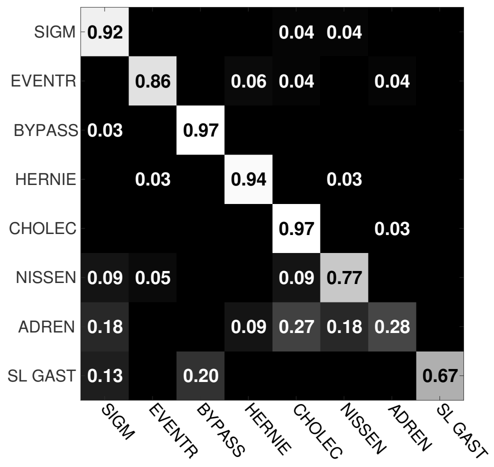
	SIGM	EVENTR	BYPASS	HERNIE	CHOLEC	NISSEN	ADREN	SL GAST
SIGM	0.46	0.08	0.04	0.23			0.15	0.04
EVENTR	0.06	0.43	0.11	0.11	0.14	0.07	0.04	0.04
BYPASS	0.14	0.03	0.72		0.03			0.08
HERNIE	0.17	0.06	0.06	0.56	0.09	0.06		
CHOLEC	0.05	0.13	0.03	0.06	0.61	0.06	0.03	0.03
NISSEN	0.04	0.09	0.14	0.18	0.14	0.41		
ADREN	0.09	0.09	0.18	0.09	0.36		0.19	
SL GAST		0.06	0.07		0.20	0.40	0.07	0.20

(a) RGB



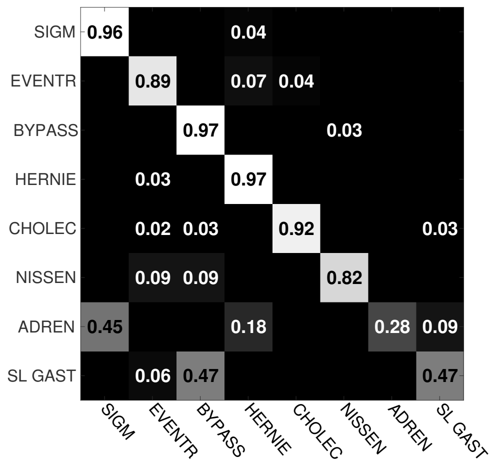
	SIGM	EVENTR	BYPASS	HERNIE	CHOLEC	NISSEN	ADREN	SL GAST
SIGM	0.46	0.03	0.15	0.08	0.08	0.04	0.08	0.08
EVENTR	0.14	0.36	0.11	0.18	0.14	0.07		
BYPASS	0.16	0.08	0.58	0.06	0.06	0.06		
HERNIE	0.02	0.09	0.03	0.65	0.09	0.06	0.06	
CHOLEC	0.08	0.11	0.06	0.11	0.44	0.06	0.11	0.03
NISSEN	0.04	0.26	0.14	0.05	0.14	0.32		0.05
ADREN	0.27	0.18		0.18	0.36		0.01	
SL GAST	0.06		0.40	0.07	0.07	0.27		0.13

(b) HS



	SIGM	EVENTR	BYPASS	HERNIE	CHOLEC	NISSEN	ADREN	SL GAST
SIGM	0.92				0.04	0.04		
EVENTR		0.86		0.06	0.04		0.04	
BYPASS	0.03		0.97					
HERNIE		0.03		0.94		0.03		
CHOLEC					0.97		0.03	
NISSEN	0.09	0.05			0.09	0.77		
ADREN	0.18			0.09	0.27	0.18	0.28	
SL GAST	0.13		0.20					0.67

(c) SIFT



	SIGM	EVENTR	BYPASS	HERNIE	CHOLEC	NISSEN	ADREN	SL GAST
SIGM	0.96			0.04				
EVENTR		0.89		0.07	0.04			
BYPASS			0.97			0.03		
HERNIE		0.03		0.97				
CHOLEC		0.02	0.03		0.92			0.03
NISSEN		0.09	0.09			0.82		
ADREN	0.45			0.18			0.28	0.09
SL GAST		0.06	0.47					0.47

(d) HOG

Figure 4.4: Confusion matrices from the best configuration (written in bold in Table 4.3), namely giving the highest accuracy for: (a) RGB, (b) HS, (c) SIFT, and (d) HOG features.

4.2.3.2 Feature Comparison

In Table 4.3, we show the classification accuracies of all the kernel-feature configurations from our experiments, where the best accuracy for each feature is written in bold. We can observe that the performances of classification using color information from RGB and HS histograms are lower than the ones of SIFT and HOG features. Such results are expected since RGB and HS histograms only consist of a simple count of pixel values contained in the video frames and the videos are visually similar to one another. However, these features still contain discriminative information considering that the classification accuracies obtained from these features are higher than chance. In contrast, the SIFT and HOG features yield good results with 86.5% and 86.1% accuracies respectively on their best configurations. These features successfully perform the task thanks to the

Feature	# words	Kernel		
		Lin.	χ^2	HI
RGB	100	39.4	37.5	33.2
	300	43.3	**50.5**	46.2
	500	37.0	49.5	47.1
HS	100	34.1	37.5	33.7
	300	36.1	41.8	37.5
	500	38.0	**43.3**	42.3
SIFT	500	68.8	83.7	85.6
	1000	67.3	85.6	**86.5**
	1500	62.5	82.2	83.2
HOG	500	71.2	84.1	82.2
	1000	72.6	85.6	82.7
	1500	73.6	**86.1**	85.6

Table 4.3: Classification accuracies for all configurations. The accuracies shown in bold are the best accuracies for each feature.

higher level information contained in them, leading to better representation of the videos compared to the color histograms.

In Fig. 4.4, we show the confusion matrices from the best configuration of each visual feature. As previously discussed, we can see that the SIFT and HOG features perform significantly better than RGB and HS features. The confusions on classification using the SIFT and HOG features come from the least represented classes, i.e., Adrenalectomy and Sleeve Gastrectomy. This problem is solved by incorporating the features using GMKL (presented in Section 4.2.3.5)

Intuitively, for each combination of visual feature and encoding method, there exists an optimal dictionary size. If the dictionary is too small, the feature encoding method loses its discriminative power, i.e., the inter-class variation is high. On the other hand, if it is too big, the feature representations of videos belonging to the same surgery will not match, i.e., the intra-class variation is high. This number however highly depends on the feature itself and the feature encoding method, i.e., an optimal feature encoding configuration for a feature type might not generalize to other features. For example, the optimal dictionary size for SIFT is 1000.

4.2.3.3 Downsampled Classifications

In the previous subsections, we show the results from experiments which use the complete video frames at 1 fps. Here, we study the evolution of the classification accuracy with respect to the number of frames included in the process.

We take the best configuration of each visual feature (shown in bold in Table 4.3) and run the pipeline for the downsampled classification tasks. The accuracies obtained from this experimental setup are shown in Figure 4.5. As expected, the accuracy increases along with the increase of the number of frames. However, in this setup, the RGB and

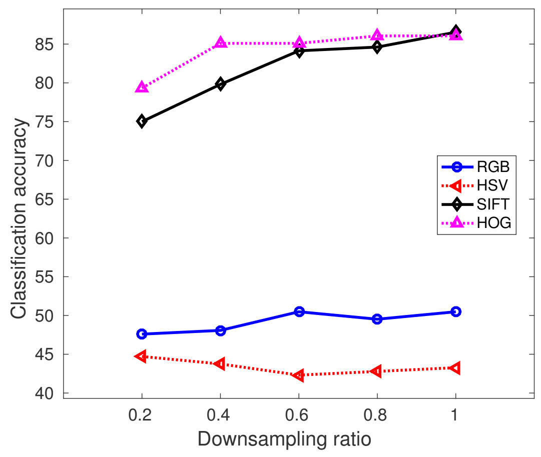


Figure 4.5: Classification accuracies for downsampled classification on the best configuration of each feature.

HSV features are not affected significantly by the number of frames. This could be due to the visual similarity of the video frames which causes these features to be similar across the videos, reducing the importance of the amount of images used to represent the videos. It is interesting to note that by reducing the video frames to 60%, all the classification accuracies only decrease by less than 3.4%.

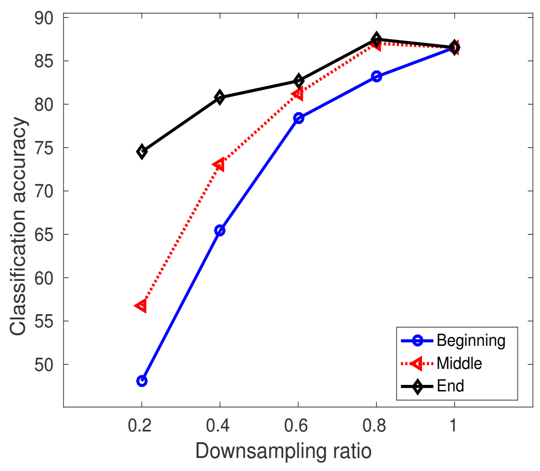
4.2.3.4 Video Clip Classification

In contrast with the previous setup, here, we study the evolution of the classification accuracy with respect to the temporal locations of the frames included in the process. The visual features used in this experiment are SIFT and HOG since they perform significantly better than RGB and HS histograms. We test them using one configuration for each feature encoding method to show the effect of feature encoding on the results.

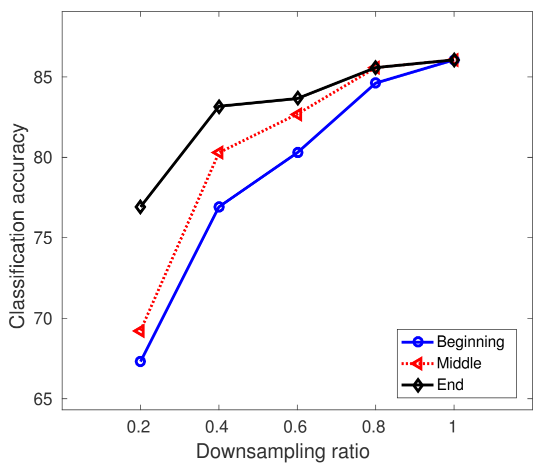
In Figure 4.6, we show the video clip classification results. We can see that similarly to the downsampled classification setup, the classification accuracies for video clip classification also increase along with the increase of number of frames. From the graphs, one can clearly observe that the beginning is the least discriminative part of the video. This is quite intuitive since the beginning of the videos usually includes similar scenes, i.e. trocar and tool insertions. Thus, this part of the surgery is the least discriminative among the three.

As for the middle part, since it contains more procedure-specific actions, it gives better results than the beginning part. It does not, however, outperform the video clip classification using the end part of the videos. This could be due to the fact that the middle part are subsampled randomly from the videos, which subsequently creates large intra-class variations.

As a result, the end part of the video is the most discriminative part of the videos. It can be seen that just by taking the last 20% of the video frames, the classifiers manage to obtain accuracies above 70%. Despite the similar actions contained at the end of all



(a) SIFT



(b) HOG

Figure 4.6: Classification accuracies for video clip classification on the SIFT and HOG features using the best configuration shown in Table 4.3

laparoscopic videos (i.e. trocar removal and incision stitching), these actions take less time than the actions performed at the beginning of the videos. Thus, the procedure-specific actions which occur at the end of the videos are still captured even after the subsampling process.

4.2.3.5 Feature Combination

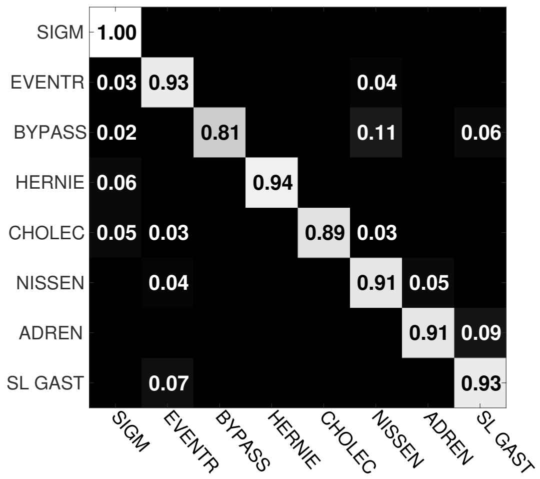
In this experiment, we take the best configuration for each feature (shown in bold in Table 4.3), and input them as the base kernels in the GMKL approach. To summarize, we show the best encoding-kernel configuration for each feature as well as the classification result using GMKL in Fig.4.7a. As can be seen in Figure 4.7-b, though it slightly decreases the recall for the bypass surgery, GMKL increases the recalls for all other surgery types, thus improving the overall classifier performance. It gives the best classification model yielding 90.9% of accuracy. It can also be seen that GMKL solves the problem of low recall on classes with less instances, e.g., Adrenalectomy and Sleeve Gastrectomy.

4.2.4 Conclusions

Here, we have presented a pipeline to tackle a pre-segmented surgery activity recognition task which involves the identification of the surgery being performed in a video. To the best of our knowledge, there has not been any studies in the literature addressing this problem, especially on laparoscopic videos. Despite the various visual challenges inherent to this type of videos, our proposed pipeline is able to carry out the task with high performance. Obtained using the Laparo208 dataset which contains 208 laparoscopic videos, the experimental results show that satisfactory classification results can be achieved by using only a temporal resolution of 0.6 fps. In addition, it is demonstrated that the end part of the laparoscopic videos contains the most discriminative information regarding

Feat.	Best Setup		Acc. (%)
	#words	Kernel	
RGB	300	χ^2	50.5
HS	500	χ^2	43.3
SIFT	1000	HI	86.5
HOG	1500	χ^2	86.1
GMKL All Features			**90.9**

(a)



(b)

Figure 4.7: Surgery classification results using GMKL: (a) the best encoding-kernel configurations for all features and (b) the confusion matrix. Note that the result using feature combination is obtained using the features extracted at 1 fps.

the surgeries, compared to the beginning and middle parts of the videos. Ultimately, we have shown that the combination of the visual features results in the best classification model, yielding an accuracy of 90.9%.

Having observed that the proposed pipeline works very well for *pre-segmented* surgical activity recognition on laparoscopic videos at the coarsest level, we are interested in investigating how the handcrafted visual features used in this pipeline perform in carrying out a *frame-wise* surgical activity recognition task at a finer level of granularity.

4.3 Surgical Phase Recognition

The recognition of surgical phases is an important topic because it is an essential component to develop a context aware system (CAS) which could be used to monitor the surgical processes, optimize OR and staff scheduling, and provide automated assistance to the clinical staff. Similarly to the surgery classification, the ability to automatically recognize surgical phases on laparoscopic videos would also facilitate the process of video database indexing. By identifying the key frames which indicate the transition between surgical phases, it would provide an effortless navigation for video browsing. This is particularly interesting for training purposes and postoperative review. For instance, a young surgeon, who needs to learn how a certain phase is executed, is just one click away from finding the phase without the need to scroll through the complete video.

Here, the addressed task falls into the *frame-wise* classification category, where the method assumes that there is a sequence of activities (in this case, phases) in each video. Specifically, we address the problem of surgical phase recognition on cholecystectomy videos. The cholecystectomy procedure is chosen because it is one of the most common procedures executed laparoscopically and there have been several studies in the literature that address surgical phase recognition on cholecystectomy [Blum 2010, Padoy 2012, Stauder 2014]. In [Padoy 2012, Stauder 2014], it has been shown that the tool usage signals yield promising results in performing the surgical phase recognition task on such videos. Here, we present a recognition pipeline to carry out the task and compare the performance of tool binary information and handcrafted visual features. Ultimately, we show how the combination of tool and handcrafted features can further improve the phase recognition results.

4.3.1 Formalization

For a laparoscopic video $\mathbb{V} = I_1, \dots, I_N$ containing N video frames, the problem of surgical phase recognition is formulated as finding the corresponding phase p_i for video frame I_i at time i. The function f which performs the task typically accepts features $\mathbf{x}_i$ extracted from frame I_i. Thus, it can be written as:

$$f : \mathbf{x}_i \rightarrow p_i, \quad 1 \leq i \leq N \text{ and } 1 \leq p_i \leq K \tag{4.3}$$

where K is the defined number of phases in the procedure. However, since there are temporal constraints that need to be enforced in surgical phase recognition, the estimation of phase p_i does not only depend on the features $\mathbf{x}_i$. In offline recognition, the pipeline can access the full sequence to determine the phase at time i; while in online recognition, the pipeline cannot access the future information. Assuming the task of classifying an image I_i, the offline and online recognition problems can then be rewritten respectively as $f : \check{\mathbf{X}} \rightarrow p_i$ and $f : \mathbf{X} \rightarrow p_i$, where $\check{\mathbf{X}} = [\mathbf{x}_1, \dots, \mathbf{x}_i, \dots, \mathbf{x}_N]$ and $\mathbf{X} = [\mathbf{x}_1, \dots, \mathbf{x}_i]$ are the extracted features.

4.3.2 Methodology

The first step of surgical phase recognition is to extract features from the video frames. Here, we use tool binary information as well as visual features to represent the video frames. The features are then passed to a classification pipeline, which consists of SVM and Hierarchical HMM (HHMM). The SVM is used to compute the values which represent the confidences of a frame belonging to the phases and the HHMM enforces the temporal constraints of the surgical workflow into the recognition pipeline. Note that, since the task involves identifying the corresponding surgical phase for all video frames, there is no irrelevant frame rejection step in this pipeline.

4.3.2.1 Feature Representation

In this work, each frame can be represented by two different features: binary tool and handcrafted visual features. The binary tool feature contains the information regarding the tools being used at a defined time. Formally, for each video frame, a binary vector is defined where each element of the vector represents the usage of a surgical tool in the frame.

ID	Phase	Avg. duration (min)
P1	Preparation	1.8±1.7
P2	Calot triangle dissection	15.6±11.1
P3	Clipping and cutting	2.9±2.1
P4	Gallbladder dissection	12.2±8.9
P5	Gallbladder packaging	1.6±0.8
P6	Cleaning and coagulation	3.0±2.6
P7	Gallbladder retraction	1.4±1.2
Complete surgery		38.4±17.1

Table 4.4: List of phases in the Cholec80 dataset, including the mean ± std of the duration of each phase in seconds.

For the handcrafted visual features, we extract the same visual features from each video frame as the ones used for the surgery classification task, as described in Section 4.2.1.2. Later, we encode each visual feature using their best feature encoding configuration shown in bold in Table 4.3. The video frame is then represented by a feature vector which is the concatenation of all the visual features extracted from that frame.

4.3.2.2 Phase Recognition

The classification step consists of two classification algorithms: a Support Vector Machine (SVM) and a Hierarchical Hidden Markov Model (HHMM). The multi-class SVM is designed to take the feature representation of the video frames as input and to compute the confidence values representing the probabilities of a video frame belonging to the phases. To enforce the temporal constraints, we use use a two-level Hierarchical HMM. The top-level contains states that model the surgical phases and their transitions, while the bottom-level nodes model the intra-phase dependencies. All these parameters (except the number of top-level states which is set to the number of classes) are data-driven and automatically learnt from the data. The HHMM observations are given by the confidence values obtained from the SVM.

4.3.3 Experimental Setup

4.3.3.1 Cholec80 Dataset

The *Cholec80*[2] dataset contains 80 videos of cholecystectomy procedures performed by 13 different surgeons at the University Hospital of Strasbourg. The videos are captured at 25 fps. For faster processing and to reduce redundancy, we downsample the videos to 1 fps by taking the first frame from every 25 frames. After a discussion with our senior surgeon, we defined seven surgical cholecystectomy phases. The list of the phases along with the statistics can be found in Table 4.4. In Fig. 4.8, we show frame samples from each phase and illustrate the transitions between phases generated from our dataset.

[2]http://camma.u-strasbg.fr/datasets/

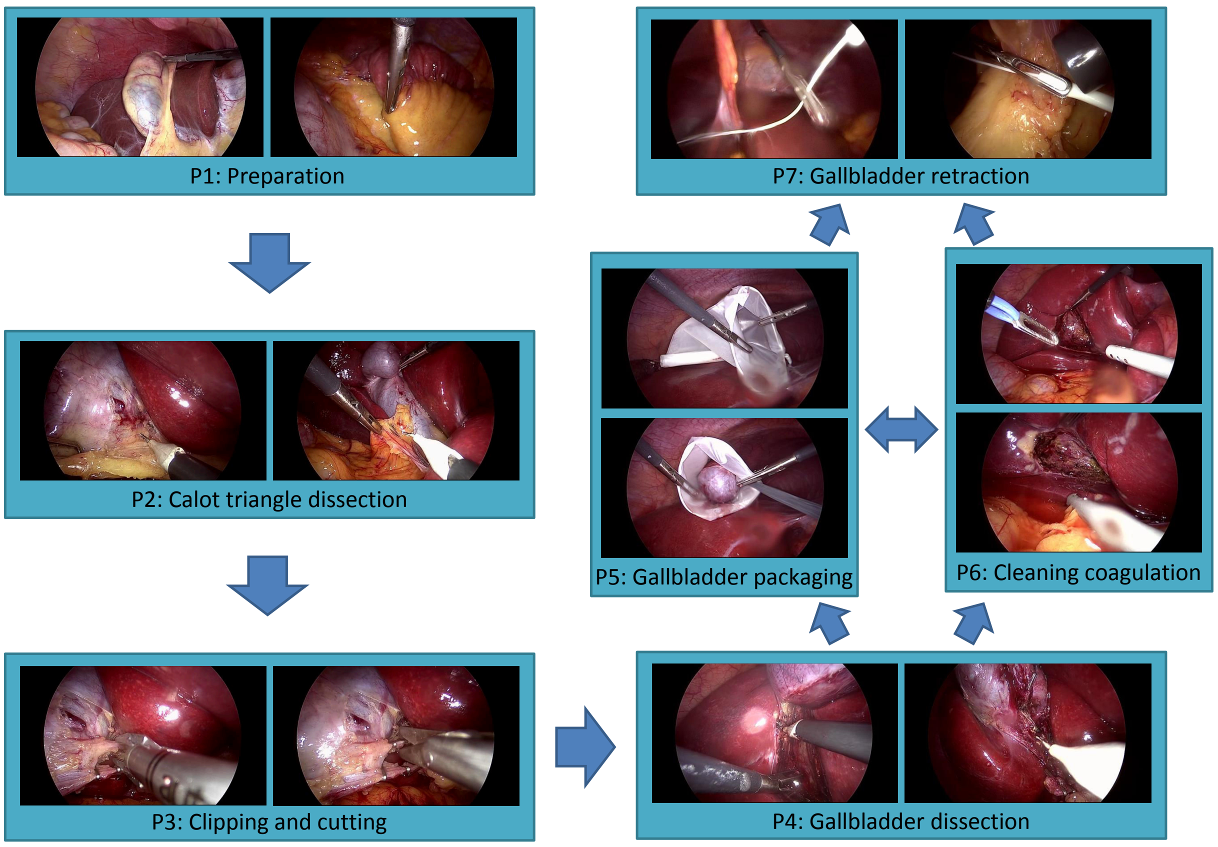

P1: Preparation
P2: Calot triangle dissection
P3: Clipping and cutting
P4: Gallbladder dissection
P5: Gallbladder packaging
P6: Cleaning coagulation
P7: Gallbladder retraction

Figure 4.8: Surgical phase transitions of cholecystectomy procedures in the Cholec80 dataset.

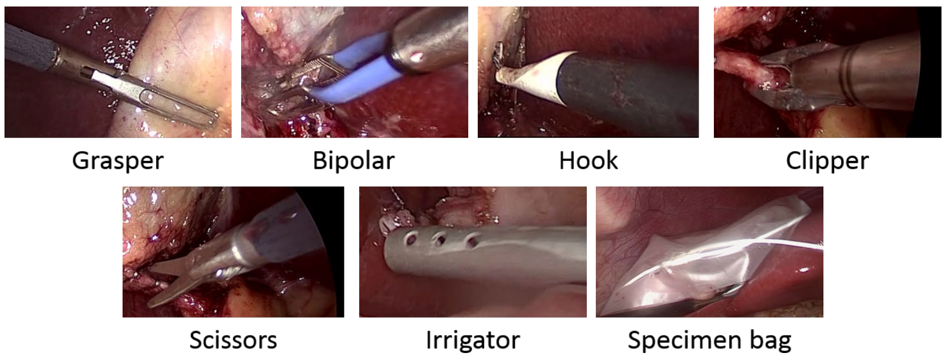


Figure 4.9: List of the seven surgical tools used in the Cholec80 dataset.

ID	Phase	Avg. Duration (min.)
P0	Placement trocars	3.0±2.0
P12	Preparation	7.0±3.6
P3	Clipping and cutting	6.5±3.2
P4	Gallbladder dissection	9.4±7.3
P5	Retrieving gallbladder	6.5±4.1
P6	Hemostasis	5.6±1.0
P7	Drainage and closing	2.9±2.1
Complete surgery		40.8±10.4

Table 4.5: List of phases in the EndoVis datasets, including the mean ± std of the duration of each phase in seconds.

The Cholec80 dataset is also annotated with surgical tool presence at 1 fps. A surgical tool is defined as present if at least 50% of the tip is visible in the video frame. This annotation is used for the binary tool feature representation in the experiments. The list of the surgical tools is shown in Fig. 4.9.

4.3.3.2 EndoVis Dataset

The second dataset is a public dataset released for the EndoVis workflow challenge at MICCAI 2015[3]. The dataset contains seven cholecystectomy videos collected at Klinikum Rechts der Isar [Stauder 2014] that are captured at 25 fps and downsampled to 1 fps for processing. Even though the challenge had been cancelled due to insufficient participation, the data is still useful to demonstrate the generalization of our approach.

The list of phases in the EndoVis dataset is shown in Table 4.5. It can be seen that phase P3 is longer in Endovis than in Cholec80 (compare to Table 4.4). This is due to the fact that in Cholec80, P3 is typically started when the calot triangle is clearly exposed. Yet, this is not the case in EndoVis. As a result, extra dissection steps are included in P3, leading to a longer P3 in EndoVis.

Some phases in EndoVis have been defined differently from the phases in Cholec80. For instance, a phase *placement trocars* is defined in the EndoVis dataset, even though it should be noted that this phase is not always visible from the laparoscopic videos. Additional sources of information (e.g., external videos), which are not available in the dataset, are required to label this phase correctly. Another difference is in the definition of the *preparation* phase. In the EndoVis dataset, the *preparation* phase includes the *calot triangle dissection* phase (hence the phase ID *P12* in Table 4.5). The other phases are defined similarly to the phases in Cholec80.

In Fig. 4.10, we show the frame samples from each phase and the phase transitions generated from the EndoVis dataset. As can be seen, the phase transitions in the EndoVis dataset are not the same as the ones in the Cholec80 dataset (see Fig. 4.8). In the EndoVis dataset, the transition is almost straightforward, except for the transition between P5

[3]http://grand-challenge.org/site/endovissub-workflow/data/

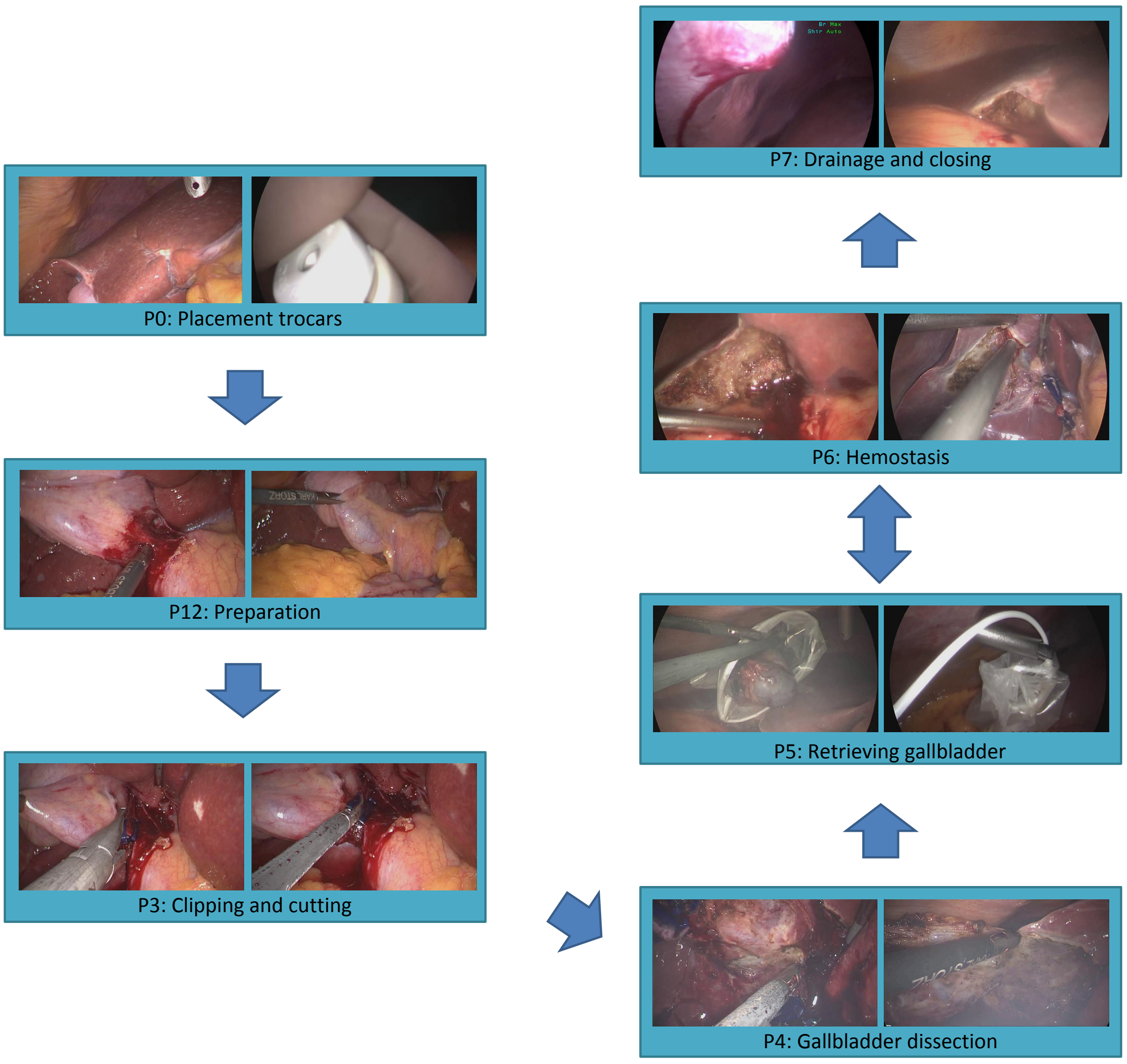


Figure 4.10: Surgical phase transition of cholecystectomy procedures in the EndoVis dataset.

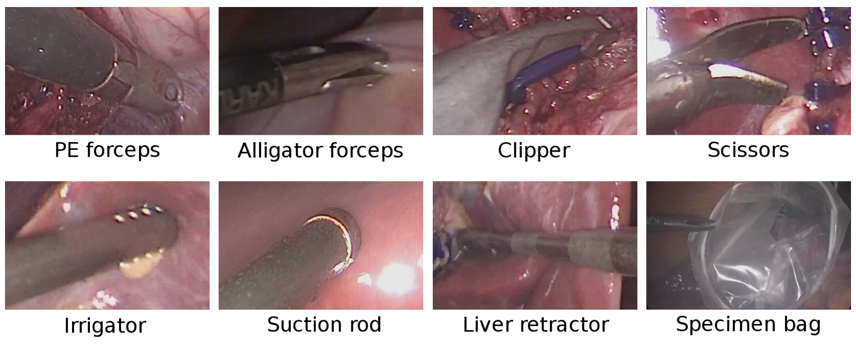


Figure 4.11: List of the surgical tools used in the EndoVis dataset.

and P6. This is, however, to be expected since the datasets come from two different hospitals where there might be slight differences in the surgical workflows. In addition, the Cholec80 dataset contains higher variability since it contains more recordings of surgeries performed by 13 different surgeons.

As for the surgical tools, the tools annotated in the EndoVis are also slightly different than the ones in the Cholec80 dataset. The annotation provides tool usage information of ten surgical tools. However, one tool is never utilized in the dataset. Furthermore, two types of clipper are defined, but the difference is not in the tool itself, but in the type of clips being used (metal or plastic). Ultimately, there are 8 distinct surgical tools present in the EndoVis dataset. The list of the tools is presented in Fig. 4.11.

The tool binary annotations in the EndoVis dataset represent the usage of tools. Note that the tool *usage* annotation does not provide the same information as the tool *presence* annotation in the Cholec80 dataset. In tool presence annotations, the binary flag is only switched on when the corresponding tool is present in the image (at least 50% of the tip is visible, see Section 4.3.3.1). While in tool usage annotation, the binary flag is turned on for a tool if it is being used in the body even though it is not visible in the image. Subsequently, the tool usage annotation process cannot be performed accurately solely by observing the laparoscopic videos because the tools being used might not appear in the frames. Additional sensors, such as magnetic trackers or external cameras, are required to provide this additional information.

4.3.3.3 Features

Here, we will compare the performance of various features for phase recognition. We first show the performance using the binary tool information. Then, we present the results of phase recognition using a combination of visual features. Finally, we make use of both the handcrafted visual features and the tool information. We do this in two ways. In the first approach, we perform feature selection on the visual features through canonical correlation analysis (CCA) using the binary tool information in order to obtain more semantically meaningful and discriminative features. This approach was presented in [Blum 2010] and is shown to yield improved performance for phase recognition on cholecystectomy videos. We refer to this feature as CCA feature. Note that this feature only requires the tool information at training time. In the second approach, we concatenate the visual features with the tool information to represent the video frames. The significant difference between the concatenation approach is that the CCA feature does not require the binary tool information at test time since the feature selection transformation is already computed using CCA at training time.

4.3.3.4 Classification

Following the results shown in Table 4.3, we also use non-linear kernels for the visual features and also the concatenation of visual and binary tool features, specifically the histogram intersection kernel because we see that it performs slightly better than the χ^2

kernel. For the binary tool and CCA features, we use linear SVM classifiers because we did not observe any improvement in the preliminary experiments when non-linear kernels are applied.

As for the HHMM, we use a mixture of five Gaussians as observation model, except for the binary tool and CCA features, where one Gaussian is used instead. These are the optimal number of Gaussians obtained from preliminary experiments. The type of covariance is diagonal. We use a two-level HHMM, where the number of top-level states is set to be equal to the number of phases, i.e., seven for both Cholec80 and EndoVis. The number of bottom-level states is computed based on the dataset.

4.3.3.5 Evaluation

To measure the method's performance for phase recognition, several evaluation metrics are used , i.e., precision, recall, and accuracy as defined in [Padoy 2012]. Precision and recall show the quality of the recognition results for each phase. For a video $\mathbb{V}_v$, the precision and recall for phase p is defined by:

$$\mathrm{prec}_p^v = \frac{\mathrm{tp}_p^v}{\mathrm{tp}_p^v + \mathrm{fp}_p^v} \tag{4.4}$$

$$\mathrm{rec}_p^v = \frac{\mathrm{tp}_p^v}{\mathrm{tp}_p^v + \mathrm{fn}_p^v} \tag{4.5}$$

where tp_p^v, fp_p^v, and fn_p^v are respectively the true positive, false positive, and false negative rates for phase p in video $\mathbb{V}_v$.

In contrast, accuracy represents the percentage of correct detections in the complete surgery. For a video $\mathbb{V}_v$ of length N_v, it is expressed as:

$$\mathrm{acc}^v = \frac{\mathrm{tp}^v}{N_v} \tag{4.6}$$

where tp^v is the number of correctly classified frames in video $\mathbb{V}_v$. We use three evaluation metrics in order to obtain more comprehensive results since short phases will not be well represented in accuracy due to the fact that it is computed over the complete video. In other words, low precision/recall with high accuracy means that the short phases are not recognized well and high precision/recall with low accuracy means that the long phases are not recognized as well as the short ones. Thus, it is informative to observe the precision and recall for such phases in addition to the overall accuracy.

All these metrics are then averaged over the surgeries:

$$\text{prec}_p = \frac{1}{V}\sum_{v=1}^{V}\text{prec}_p^v \tag{4.7}$$

$$\text{rec}_p = \frac{1}{V}\sum_{v=1}^{V}\text{rec}_p^v \tag{4.8}$$

$$\text{acc} = \frac{1}{V}\sum_{v=1}^{V}\text{acc}^v \tag{4.9}$$

Finally, for precision and recall, they are averaged over the phases.

We run a 4-fold cross-validation on the Cholec80 dataset and a leave-one-out cross validation on the EndoVis dataset. Note that the training of the classification models in the pipeline (SVM and HHMM) contains random initializations, which can result in different outcomes for each experimental run. Thus, for each validation fold, we perform five experimental runs and average the evaluation metrics over the experimental runs. To perform an ablative analysis on the visual features and the temporal model, we show the recognition results before applying the HHMM (after the SVM) and after the HHMM. It is to study how important the temporal model is for the performance of the system.

4.3.4 Experimental Results

4.3.4.1 Results on Cholec80 Dataset

In Table 4.6-a, we show the results of phase recognition on the Cholec80 dataset before applying the HHMM. For all evaluation metrics, the binary tool features outperform the handcrafted visual features. This outcome is however expected since the binary tool features are semantically more meaningful than the handcrafted visual features. Nevertheless, the handcrafted visual features contains discriminative information for the surgical phase recognition task since the obtained classification performance is much higher than chance. Thus, it is interesting to see how the combination of both features perform for the task.

When we apply the binary tool to perform CCA on the handcrafted visual features, we do not observe any improvement in the recognition results. This might be due to the fact that there is not a high correlation between the visual features extracted from the system and the binary tool feature. On the other hand, the concatenation of binary tool and visual features outperforms other features, yielding 51.3% accuracy.

One might notice that the recognition results shown in Table 4.6 are not very good, considering the best features can correctly classify only roughly half of the images in the dataset. However, it is important to note that these results are obtained through frame-wise classification that only rely on handcrafted visual and binary tool features extracted from laparoscopic video frames which have high inter-class similarities. More importantly, these results are obtained without incorporating any temporal constraint,

Feature	(a) Cholec80			(b) EndoVis		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	39.7±35.3	40.2±38.3	50.0±3.4	44.3±32.5	48.5±39.3	49.0±9.7
Visual	31.0±19.8	18.2±31.0	45.8±4.4	35.7±6.6	33.2±10.5	36.1±2.6
CCA	23.8±12.5	19.7±27.1	40.0±3.1	31.1±4.6	31.6±22.6	32.6±5.3
Visual + Tool	**50.7±19.2**	**43.1±29.6**	**51.3±4.5**	**56.7±9.8**	**55.3±32.6**	**56.4±9.0**

Table 4.6: Phase recognition results before applying the HHMM (mean ± std) on: (a) Cholec80 and (b) EndoVis datasets. The best result for each evaluation metric is written in bold.

Feature	Overall-Offline (%)			Overall-Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	62.8±23.8	72.8±14.8	68.7±7.3	55.3±28.7	61.7±22.0	53.0±4.6
Visual	47.2±21.8	56.2±13.6	53.9±1.5	41.7±24.8	48.0±13.2	48.1±2.5
CCA	58.2±21.6	62.6±15.5	69.7±3.1	43.3±31.2	45.5±20.5	50.7±3.8
Visual + Tool	**72.2±19.1**	**78.0±11.6**	**79.3±6.8**	**61.5±26.5**	**66.4±16.9**	**58.2±5.3**

(a) Cholec80

Feature	Overall-Offline (%)			Overall-Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	81.4±16.1	79.5±12.3	73.0±21.5	80.3±18.1	77.5±18.8	69.8±21.7
Visual	49.7±15.6	33.2±21.5	46.5±24.6	46.6±16.2	48.0±18.5	43.4±21.6
CCA	66.1±22.3	64.7±22.1	61.1±17.3	52.3±22.2	49.4±21.5	44.0±22.3
Visual + Tool	**87.4±13.5**	**85.5±9.4**	**82.3±8.9**	**80.6±13.6**	**80.1±13.8**	**77.1±9.3**

(b) EndoVis

Table 4.7: Phase recognition results after applying the HHMM (mean ± std) on: (a) Cholec80 and (b) EndoVis. The best result for each evaluation metric is written in bold.

thus they are not the final results from the pipeline.

In Table 4.7-a, we show the recognition results on the Cholec80 dataset after applying the HHMM which enforces the temporal constraint into the pipeline. It can be seen that the offline results are always better than the online counterparts. This is however expected due to the nature of how the offline and online recognitions are performed. In offline mode, the algorithm recognize the phases in a video using the complete video, while it is not the case in online mode. Therefore, the predictions in online mode are not as smooth as the ones in offline mode. The results also show that there is a significant improvement obtained by applying the temporal information to the recognition pipeline. For all features, the recognition results in both offline and online modes are better than the ones before applying the HHMM.

Despite the fact that the CCA features yield the worst recognition results before applying the HHMM, the results after applying the HHMM are better than the ones using solely the visual features: the accuracies are 69.7% vs. 53.9% (offline) and 50.7% and 48.1% (online). This shows that a better temporal model is learnt using the SVM confidences computed using the CCA feature, which indicates that the confidences are easier to smooth temporally compared to the confidences computed using the visual features.

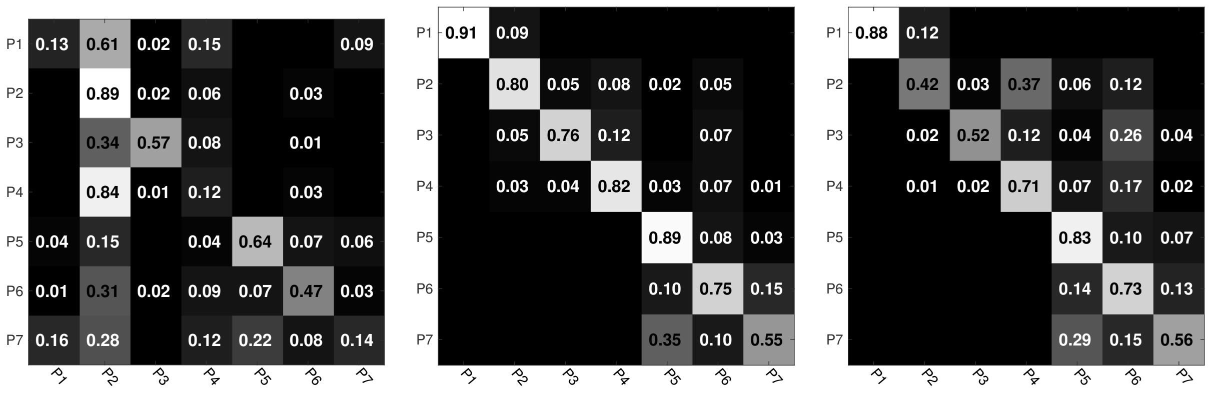
Similarly to the recognition before applying the HHMM, the best results are given by the concatenation of the handcrafted visual and binary tool features. The fact that the concatenation outperforms the CCA feature indicates that the extracted visual feature and the binary tool feature are not heavily correlated. However, it should be recalled that the CCA feature does not require the binary tool feature to be available at test time, while the concatenation does. Thus, in terms of practicality, the CCA feature is more favorable than the concatenation feature when the binary tool feature is not typically immediately available for the phase recognition task.

To better visualize the recognition results on the Cholec80 dataset, we show the confusion matrices in Fig. 4.12-a. The matrices are generated on one random experimental run on Cholec80 using the best feature, i.e., the concatenation of handcrafted visual features and the binary tool information. The confusion matrix generated from the results before applying the HHMM demonstrates that many images are confused for most of the phases. However, as previously stated, this is expected since there is no temporal constraint enforced in these results.

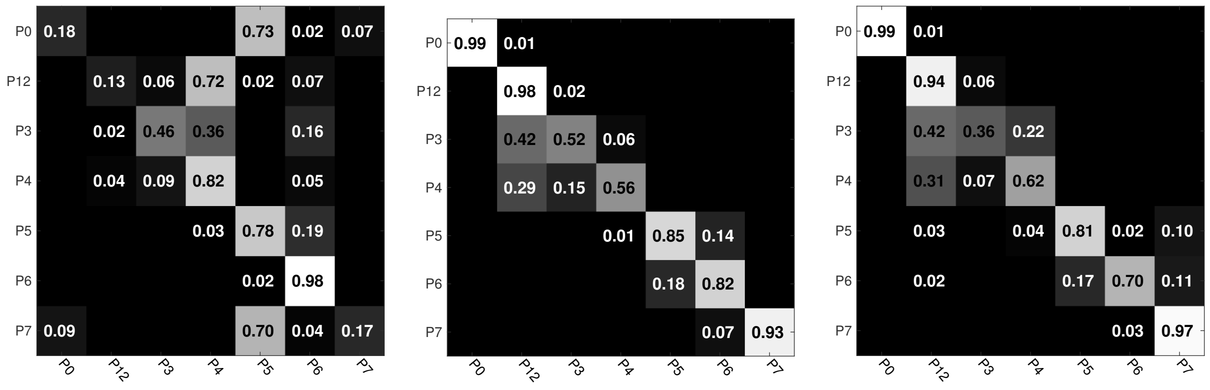
Once the temporal model is applied, the recognition results are improved, especially in offline mode where most of the images are correctly classified. As for the results in online mode, we can see that P2 is often confused with P4. This could be due to the fact that the two phases look very similar to one another. In addition, the tools used in these phases are typically the same: grasper and hook. Since the HHMM does not observe the complete video in online mode, this might cause the HHMM to move to P4 too quickly during the procedure. Despite this fact, the overall recognition results in both offline and online modes are better than the ones before applying the HHMM.

4.3.4.2 Results on EndoVis Dataset

We show the recognition results on the EndoVis dataset before applying the HHMM in Table 4.6-b. We can see that the results demonstrate a trend similar to the one of the Cholec80 results. As for the results after applying the HHMM, they are shown in Table 4.7-b. One can observe that the results using the binary tool are better than the ones of the Cholec80 dataset (Table 4.7-a). This might be due to the fact that the EndoVis dataset contains less videos and thus less variability than the Cholec80 dataset. Furthermore, the tool information from the EndoVis dataset is more semantically meaningful for the surgical phase recognition task since it represents *tool usage* while the tool features in the Cholec80 dataset represents *tool presence*. This might result in SVM confidences capable of constructing better temporal models. However, the handcrafted visual features perform



(a) Cholec80



(b) EndoVis

Figure 4.12: Confusion matrices generated from the recognition results of one experimental run using the concatenation of handcrafted visual and binary tool features for: (a) Cholec80 and (b) EndoVis. They are generated from recognitions (left to right) before applying the HHMM, after applying the HHMM in offline mode, and after applying the HHMM in online mode.

slightly worse on the EndoVis dataset than on the Cholec80 dataset. We hypothesize that this might be due to the fact that the image quality of the videos in Cholec80 is better than the quality in EndoVis. By comparing the images in Fig. 4.8 and 4.10, we can see that the images from the EndoVis dataset are less sharp and cloudier than the ones from Cholec80. Ultimately, it is also shown here that the combination of the visual and the binary tool features yield the best results for the surgical phase recognition task.

In Fig. 4.12-b, we show the confusion matrices generated from the recognition results using the combination of visual and tool features. It can be seen that before enforcing the temporal constraints, the feature is not discriminative for the task. This result is expected due to the low inter-class variance. After applying the HHMM, we can observe much improvement in the results. However, it can be seen that phase P3 are particularly difficult to recognize as it is often confused with its neighbouring phases. This might be due to the fact that the phase P3 is sometimes started when the calot triangle is not yet clearly exposed. As a result, the phase P3 contains extra dissection steps (which looks similar to P12 and P4).

4.3.5 Conclusions

We have presented a pipeline to perform surgery activity recognition on laparoscopic videos which involves recognizing the surgical phases in cholecystectomy procedures. To evaluate the pipeline, we generate a large dataset containing 80 cholecystectomy videos. We also perform the task on a public dataset, namely EndoVis, containing seven videos. We show comparisons between various features that are used for surgical phase recognition in the literature, including the binary tool information and the handcrafted visual features. The results show that the tool information is indeed more discriminative than these types of handcrafted visual features. We have also shown that the combination of both tool and handcrafted features yields the best results for the surgical phase recognition task.

4.4 Summary

In this chapter, we have presented two pipelines to address two surgical activity recognition tasks on laparoscopic videos using handcrafted visual features. The first pipeline addresses a pre-segmented classification task, i.e., surgery classification. To the best of our knowledge, this is the first work which addresses such a task on laparoscopic videos. The second pipeline tackles a frame-wise classification task, namely surgical phase recognition. To evaluate these pipelines, we generated two large laparoscopic video datasets, containing 208 and 80 laparoscopic videos, respectively. Despite the visual challenges inherent in the laparoscopic videos (e.g., motion blur, presence of smoke, and specular reflection), it is shown that handcrafted visual features contain discriminative characteristics to carry out the tasks. In the following chapter, we will apply the pipelines proposed in this chapter to investigate their robustness in performing pre-segmented and frame-wise recognition tasks on OR-scene videos using handcrafted visual features.

In Section 4.3.4, we have shown that the tool information outperforms the *handcrafted* visual features in carrying out surgical phase recognition. Intuitively, one can improve the performance of these handcrafted features by designing features that extract more discriminative characteristics from the images. It is however tedious to incorporate experts' knowledge to the manual engineering process of visual features. Instead of designing the features by hand, in Chapter 6, we propose to use deep learning algorithms to automatically learn the visual features from the data. To this end, we design several deep learning architectures to perform surgical phase recognition on laparoscopic videos. We show that the visual features learnt using our proposed deep architectures outperform the handcrafted visual features, tool information, and the combination thereof.

5 Surgical Activity Recognition on OR-Scene RGBD Videos

We are drowning in information, but starved for knowledge.
– John Naisbitt

Chapter Summary

In contrast to the laparoscopic videos, the OR-scene videos do not capture the anatomical structure inside the abdominal cavity of the patient, but rather the general activities occurring in the OR performed by the surgeons and clinicians. The OR-scene videos are closely related to the videos typically used for activity recognition in the computer vision community because they capture a scene with human actors performing the activities. However, it is unclear how state-of-the-art approaches presented in the literature perform for activity recognition on the OR scene videos. Despite the fact that the OR scene is a constrained environment where the activities are occurring mostly around the operating bed and the equipment table, surgical activity recognition on the OR-scene videos poses various challenges. For instance, due to the presence of multiple persons and equipment in the OR, there is a high frequency of occlusions occurring in the videos, which will impede the effectiveness of activity recognition methods. In addition, there is also a high similarity in appearance due to the similar color and textureless nature of various object surfaces in the OR scene.

Here, we build a ceiling-mounted multi-view RGBD camera system which records the activities in the OR. We resort to a multi-view system in order to capture a larger area of the OR. In addition, we propose to use RGBD cameras because they record simultaneously color and depth images of the scene, which provide complementary information: color images capture the visual appearance of objects' surfaces while the depth images capture the distances between objects to the sensor. The depth images provide an information which is not captured in the color images, i.e., the 3D structure of the scene. In addition, it is interesting to use such a sensor since the depth sensor works in the infrared light wavelength, thus it is invariant to illumination changes and the lack of textures. It is particularly interesting for activity recognition on OR-scene videos because the OR lights are sometimes dimmed or turned off during certain interventional surgeries so that the images on the screens appear with better contrast.

In this chapter, we first describe in Section 5.1 the multi-view RGBD acquisition system built to capture the OR scene. In order to improve the feature representation for the OR-scene videos, we propose a novel feature encoding method which is based on the BOW approach. This method retains spatio-temporal information during the feature encoding process. Then, we incorporate the encoding method into the recognition pipelines presented in Section 4.2 and 4.3 to tackle two activity recognition tasks on the OR-scene videos. In Section 5.2, we present a pre-segmented recognition task, i.e., surgical action classification, which consists of identifying the action performed in a video clip. In Section 5.3, we present the surgical phase recognition task which is similar to the task presented in Section 4.3.

5.1 Dataset Acquisition

In the following sections, we present the multi-view RGBD camera system built to record the activities in the OR. We also introduce the interventional facilities where we mount the recording system. Ultimately, we describe the datasets generated from the recordings.

5.1.1 RGBD Sensor

In this work, we chose to use Asus Xtion Pro-Live cameras as the RGBD sensors because of its light and small build with a single cable for power and data transfer. This type of RGBD sensor provides the depth information by using an infra-red (IR) structured light system. It consists of an IR projector and an IR camera. The IR projector is used to project a spatially-coded IR patterns which fall on objects in the scene and are

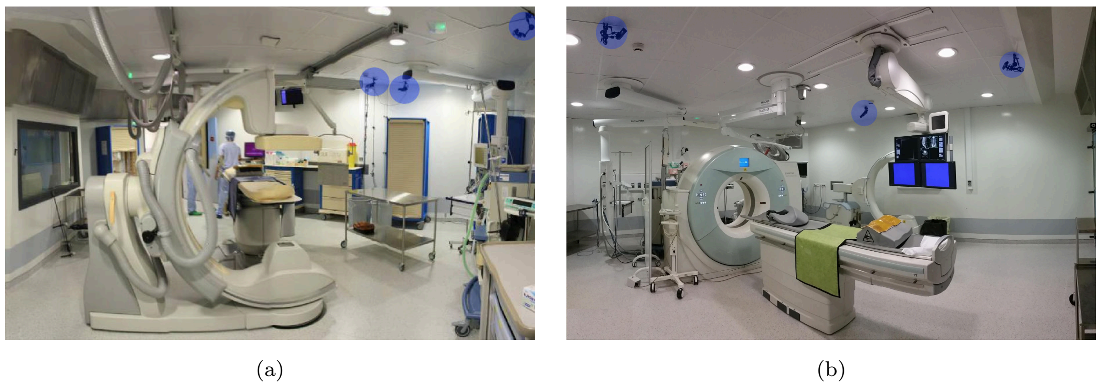


Figure 5.1: Multi-view RGBD camera system in two interventional ORs: (a) C-arm and (b) CT rooms, at the Interventional Radiology Department of University Hospital of Strasbourg. The RGBD cameras mounted on the ceiling are highlighted in blue.

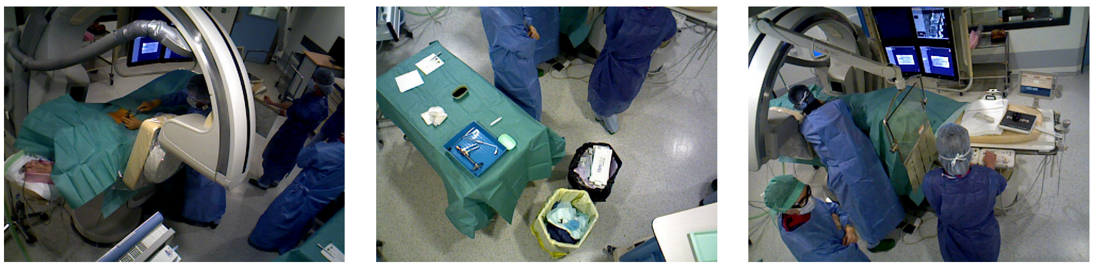
subsequently captured by the IR camera. The patterns are then decoded to determine the distance of the objects to the camera, resulting in the depth images. The fact that it is using infra-red light instead of visible light makes it invariant to illumination changes. In addition, the IR structured light patterns do not perturb the color information captured by the RGB cameras.

5.1.2 Multi-View RGBD Camera System

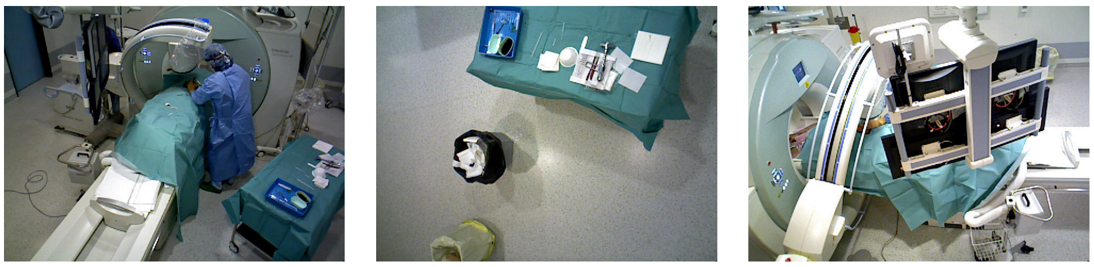
Due to the limited field of view of the sensor and the constraints of the camera placement in the OR, multiple RGBD cameras are used in order to provide a wider coverage of the scene. In order to merge the 3D scene information coming from the multiple cameras, the camera system has to be synchronized and calibrated. To automatically manage the synchronization between the cameras, a recording system built using the OpenNI2 framework was used. To calibrate the multi-view camera system, we obtain the intrinsic camera parameters via the classical calibration method using a checkerboard pattern. As for the extrinsic parameters, we use a calibration method that utilizes laser pointers. This is done by recording the laser pointers moving in the scene in the dark (without any illumination). By doing so, one can easily obtain the location of the laser pointers in the 2D images from the cameras and subsequently the 3D coordinates by using the corresponding depth images. Using this information, the transformation between the three cameras can be derived, achieving a fully-calibrated multi-view RGBD system.

The multi-view RGBD system is mounted in two interventional ORs, shown in Fig. 5.1, referred to as C-arm and CT rooms. In Fig. 5.1, one can notice that the multi-view camera system consists of three RGBD cameras. The frames captured by the three cameras are shown in Fig. 5.2.

Here, we aim to capture the scene around the operating bed and the equipment table. Therefore we configure two cameras to capture the scene around the bed and one camera to observe the equipment table. Upon further analysis, we observe that with this



(a) C-arm room



(b) CT room

Figure 5.2: Synchronized frame samples obtained by the three cameras of the multi-view RGBD system in both (a) C-arm and (b) CT rooms.

specific setup, the surgical activities are sufficiently captured by two cameras (the first two columns in Fig. 5.2) and the scene captured by the third camera typically contains a high frequency of occlusions. For instance, in the C-arm frame samples, we can only see the back of the clinicians which occlude the essential movements performed in the actions, such as the clinicians' hand gestures; while in the CT room, the view is occluded by the mobile C-arm and the screen. Because of this reason and also to reduce computational cost, we perform the surgical activity recognition tasks on OR-scene RGBD videos using the information given by the recordings of the first two cameras.

5.1.3 Vertebroplasty Datasets

Thanks to a close collaboration with the clinical staff and surgeons at the Interventional Radiology Department of University Hospital of Strasbourg, we are able to install the recording system and to record the activities in the two ORs. In both C-arm and CT rooms, various types of interventions are executed everyday. Here, we focus on recognizing activities in vertebroplasty procedures since it is one of the most frequent procedures performed at the department. The procedure consists of the injection of special cement into a fractured vertebra, with the goal of relieving spinal pain and restoring mobility. From the C-arm room, we generate two datasets which are used for the evaluation in this chapter. The first dataset is used to evaluate the pipeline for surgical action recognition and the second dataset is used for surgical phase recognition. From the CT room, we generate one dataset which is used to evaluate the methods presented in Chapter 6. Further details about these datasets can be found in the following sections and in

Appendix A.

5.2 Surgical Action Classification

The surgical action classification task consists of the identification of surgical actions performed in pre-segmented OR-scene video clips. Similarly to the surgery classification task presented in Section 4.2, this is the first step to investigate whether off-the-shelf computer vision solutions can be used to address activity recognition on such a unique dataset, which consists of video clips of activities recorded in the OR.

In Chapter 4, we proposed to use *spatial* handcrafted visual features. Due to the nature of laparoscopic videos where rapid camera motions are inevitable, it is not straight-forward to use *spatio-temporal* features. This is however not the case on the OR-scene videos since they are obtained from static ceiling-mounted cameras. Since the spatio-temporal features contain not only spatial, but also temporal information, we propose here to utilize the spatio-temporal features to address the surgical action classification task. In particular, we are using spatio-temporal interest point (STIP) features to be extracted from the video clips.

To obtain the final representation of the video clips, we encode the visual features using the BOW approach which consists of counting the occurrence of visual words contained in a dictionary. Due to the orderless nature of the BoW approach, the spatial and temporal information from the features is lost during the feature encoding process. However, this information loss might discard some discriminative characteristics from the spatio-temporal visual features. It is thus crucial to retain the spatio-temporal information during the feature encoding process.

In the literature, there has been several studies proposing methods to alleviate the information loss during the feature encoding process. For example, in [Lazebnik 2006], an extension of the BoW approach, called Spatial Pyramid Matching (SPM), was proposed for an image classification problem. The approach retains the spatial information during the encoding process by dividing the image into subregions using rigid grids. In addition to computing the histogram of word occurrence in the whole image, SPM also computes the histogram for each subregion. To obtain the final representation, the histograms coming from the image subregions are then concatenated to the histogram obtained from the whole image. This holistic approach was shown to perform better compared to regular histogram binning for recognizing natural scene categories, especially when the image is divided into finer subregions (i.e., using 2×2 and 4×4 grids). The SPM approach has also been applied to retain 3D spatio-temporal information (2D+t) in [Laptev 2008, Choi 2008] by applying 3D rigid grids on the spatio-temporal volume and yields satisfactory results for action recognition tasks. One drawback of SPM is that the layout, which defines how the space is divided, is a rigid grid layout. Such layout neglects the information regarding the actual locations of the interest points. Especially in a large spatio-temporal space, if the rigid layout is not defined properly, all the feature points might fall into the same subregion. In such cases, there is no new information introduced by the spatio-temporal layout. Finer grids can be introduced to solve this problem. However, increasing the number of cells in the grid will lead to a dimensionality problem, since it will also increase the dimension of the final feature representation.

One of the possible solutions is to learn the best layout to divide the working space (i.e., spatial space for image and spatio-temporal space for videos), instead of using a rigid grid. In [Krapac 2011], an approach was proposed to learn the appearance model (i.e. the visual features) and the spatial model (i.e. the location of the features) using Mixture of Gaussians (MoG). However, this approach might lead to the problem of high dimensionality due to the joint encoding of visual and spatial features using MoG.

Here, we propose a feature encoding method to retain the spatio-temporal information by learning a visual dictionary to model the visual features and a 4D spatio-temporal dictionary which defines how the spatio-temporal volume is divided. By using a spatio-temporal dictionary learnt from the data, the spatio-temporal space is then divided based on the actual locations of feature points in the data, retaining more spatio-temporal information during the feature encoding process. In addition, to alleviate the high dimensionality problem, we propose to use K-means algorithm to obtain both the visual and spatio-temporal dictionaries.

In the following, we explain the recognition pipeline used to perform the surgical action classification task. Then, we describe the experimental setup used to evaluate the proposed method. After presenting the experimental results, we present the conclusions of performing the task using the proposed pipeline.

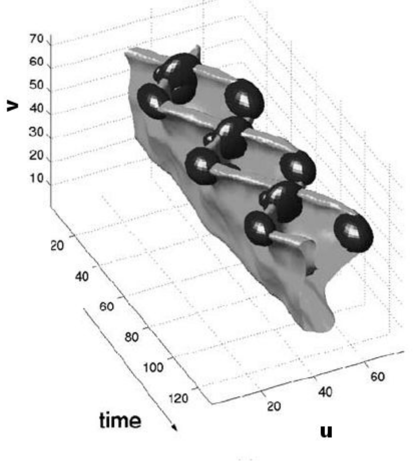
5.2.1 Methodology

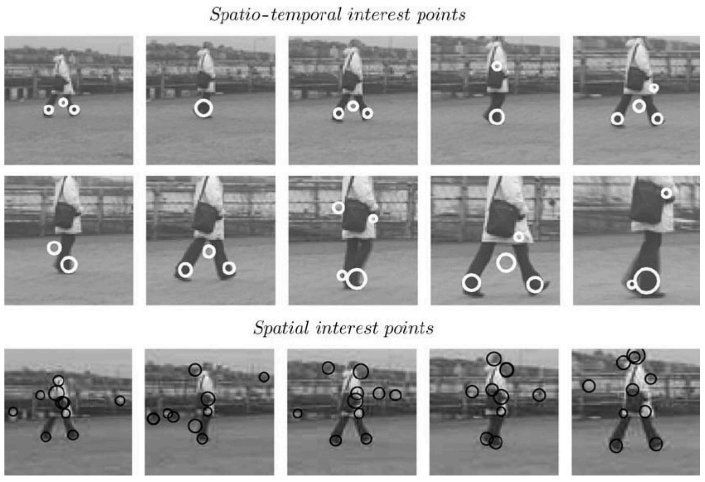
The complete pipeline to perform the surgical action classification task is as follows: (1) detecting interest points, in our case spatio-temporal interest points on intensity data and on depth data, (2) extracting features around the interest points, (3) encoding the features into histograms using our proposed feature encoding approach, and (4) classifying the histograms into the actions they belong to.

5.2.1.1 Spatio-Temporal Interest Points

Spatio-temporal interest points (STIPs) can be regarded as an extension of the 2D spatial interest points (SIPs) to 3D spatio-temporal space, consisting of (u, v) image and t time coordinates. The SIPs are defined as local image structures that are rich in terms of information contents and have high degree of variations in all directions. This typically boils down to *corner* detection. The STIPs also share the same characteristics, i.e., they can be regarded as corners in spatio-temporal domain. An illustration of the SIPs and STIPs detected in a sequence of a person walking is shown in Fig. 5.3.

Numerically, the SIPs are obtained by finding local maxima of a response function of separable linear filters applied on the image. The STIPs are computed similarly, but the filters are applied on the spatio-temporal image volume, obtained by stacking the images on top of each other.





(a) (b)

Figure 5.3: Illustration of interest points. (a) 3-D plot of detected interest points from the motion the legs of a walking person; (b) the comparison of spatial and spatio-temporal interest points detected on a sequence of a person walking. Courtesy of [Laptev 2005].

Here, we perform the detection of STIPs on color and depth images using the methods presented in [Dollár 2005] and [Xia 2013], respectively.

5.2.1.2 Interest Point Description

The STIPs and depth STIPs (DSTIPs) are described using handcrafted features which are extracted from a cuboid containing the pixel values around an interest point. The cuboid size ($\Delta_u \times \Delta_v \times \Delta_t$) is defined proportionally to the spatial and temporal scales (i.e., σ and τ) at which the interest points are detected. They are obtained by computing $\Delta_u = \Delta_v = \mu\sigma$ and $\Delta_t = \nu\tau$ where μ and ν are scalar multipliers. Here, we extract the same visual features extracted from the cuboids of STIPs and DSTIPs as the ones described in [Dollár 2005] and [Xia 2013], respectively.

From the cuboids around STIPs, we extract the histograms of optical flow (HOF). This is obtained by calculating Lucas-Kanade optical flow [Lucas 1981] between each pair of consecutive frame in the temporal domain inside the cuboid. Then, a histogram of the optical flows are computed to represent each cuboid.

From the cuboids around DSTIPs, we extract the depth cuboid similarity feature (DCSF). To extract the feature, the first step is to divide the cuboid into $s_{uv} \times s_{uv} \times s_t$ voxels, whose borders are configured in such a way that each voxel contains an integer number of depth pixels. We then define blocks whose sizes range from $1 \times 1 \times 1$ to $s_{uv} \times s_{uv} \times s_t$ voxels. From each block, a histogram of depth values is computed and normalized. Let blocks p and q have histograms $\mathbf{h}_p$ and $\mathbf{h}_q$, respectively, the Bhattacharyya distance is then computed to obtain the similarity between the two histograms:

$$d(\mathbf{h}_p, \mathbf{h}_q) = \sum_{i=1}^{M} \sqrt{h_p^i h_q^i} \tag{5.1}$$

where M denotes the number of histogram bins. The DCSF feature is then obtained by concatenating all the Bhattacharyya coefficients from all block pairs in a cuboid. Note that the dimension of this feature is consistent across different DSTIP cuboids because it does not depend on the cuboid size $(\Delta_u \times \Delta_v \times \Delta_t)$, but rather on the defined number of voxels per cuboid $(s_{uv} \times s_{uv} \times s_t)$, which is the same for all cuboids.

5.2.1.3 Data-Driven Spatio-Temporal Feature Encoding

The BoW approach typically uses the K-means algorithm to encode the feature descriptors, so that for each feature $\mathbf{x}$, a value $k \in \{1, \cdots, K\}$ is assigned to denote the index of the K-mean center which is the closest to feature $\mathbf{x}$. The collection of the computed K-mean centers is typically called a visual dictionary, in this case, we refer to them as a visual dictionary (of K_a centers) since it is computed using the visual feature descriptors. In this paper, we propose to compute another type of dictionary in order to retain the spatial and the temporal information of the visual features, i.e., a spatio-temporal dictionary (of K_{st} centers).

While the visual dictionary is used to encode the visual features (i.e., the HOF for STIPs and the DCSF for DSTIPs), the spatio-temporal dictionary can be perceived as a replacement of the rigid grid used in the SPM approach [Lazebnik 2006], used to divide the spatio-temporal space. By using a spatio-temporal dictionary learnt from the data, the spatio-temporal space is then divided based on the actual locations of feature points in the data. The space will also be divided by a non-rigid layout, instead of a rigid one. In Fig. 5.4, we show an illustration of how a spatial dictionary is learnt using the feature points. It can be seen that the spatial dictionary divides the image space into "cells", thus the spatial words contained in the dictionary can be regarded as a collection of "cells". Learning the spatio-temporal dictionary is essentially an extension of this illustration to a spatio-temporal space.

We construct the spatio-temporal dictionary by clustering the locations of the interest points (x, y, z, t), where $\mathbf{p} = (x, y, z)^\top$ and t are the 3D coordinates in the world reference frame and the temporal location of the interest point, respectively. Since the video clips vary in length, we normalize t with respect to the length of the video clip.

In [Lazebnik 2006], a better classification result is obtained when the image space is divided with a multi-resolution grids (i.e., divided by 2×2 and also 4×4 grids). Our proposed layout encoding method can also be extended to multi-resolution, by defining the size of the spatio-temporal dictionary as:

$$K_{st} = \sum_{l=1}^{L} K_{st}^l, \tag{5.2}$$

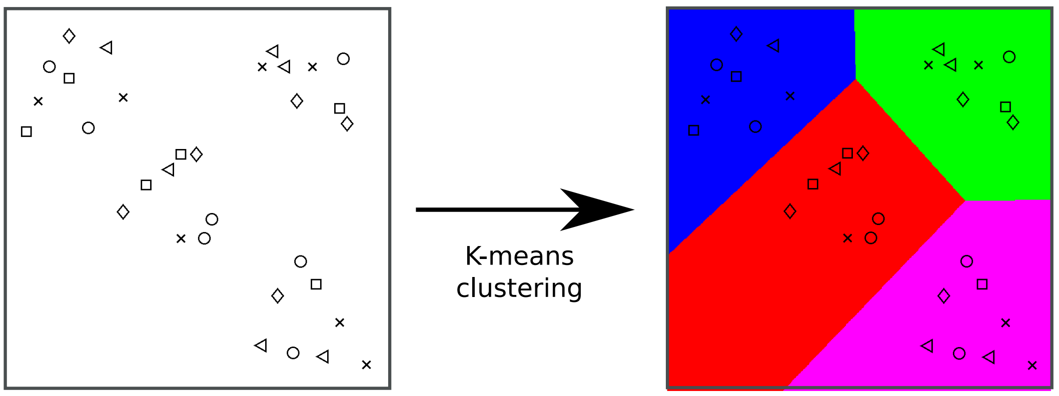


Figure 5.4: Illustration of how the layout is learnt in a 2D space with $K_{st} = 4$ (indicated by the colored cells). The features used to learn the layout are the aggregation of all the features in the dataset. Using the visual dictionary of $K_a = 5$, each feature point is grouped into a cluster represented by a shape (i.e. circle, diamond, triangle, cross, and square).

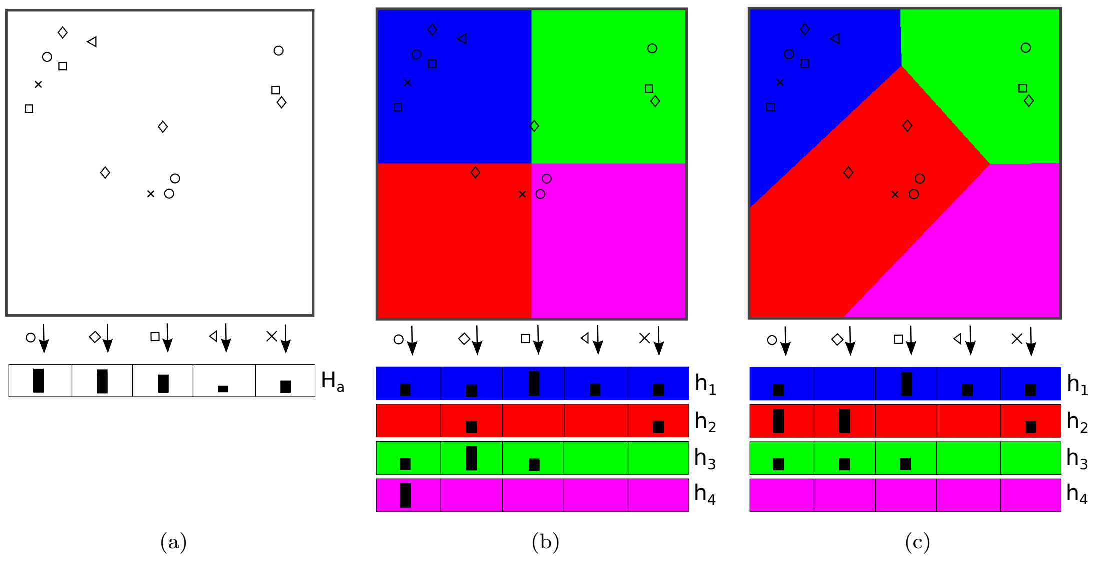


Figure 5.5: Illustration of feature encodings in a 2D space: (a) without any spatial layout, (b) with a rigid grid, and (c) with the proposed non-rigid grid. The histogram representation from the layout encoding is the concatenation of the histograms from all cells, i.e. $H_{st} = \left[h_1^\top, \ldots, h_{st}^\top\right]^\top$. The final histogram representation H is the concatenation of the histogram without the layout encoding H_a and the histogram with the layout encoding H_{st}, expressed as $H = \left[H_a^\top, H_{st}^\top\right]^\top$.

where K_{st}^{l} is the number of words in the spatio-temporal dictionary at resolution l where the dictionary size is smaller than the dictionary size of resolution $l+1$, i.e., $K_{st}^{l} < K_{st}^{l+1}$.

The final representations of the videos are obtained using both the visual and spatio-temporal dictionaries. First, for each interest point and at each spatio-temporal resolution, we find in which visual word and which spatio-temporal word (i.e., cell) they belong to. Second, we compute the histogram of visual word occurrence in every cell. Third, we concatenate all the histograms from the spatio-temporal cells with the histogram obtained from the whole image. These three steps will yield a histogram of size $K_a\,(1 + K_{st})$ for each video clip. The encoding and the concatenation process using a single-resolution spatial dictionary can be seen in Figure 5.5.

5.2.1.4 Classification

After the feature encoding process, every video clip V_i will be represented with a final feature vector $\mathbf{x}_i$. A collection of feature vectors $\mathbf{X} = [\mathbf{x}_1, \cdots, \mathbf{x}_N]$ of N video clips is used to train and test the classification model. Here, we use one-versus-all multi-class Support Vector Machines (SVMs) with two different kernels: chi-square and histogram intersection. We do not perform linear classification because as shown in Section 4.2.3, the BoW approach works best with non-linear classifiers.

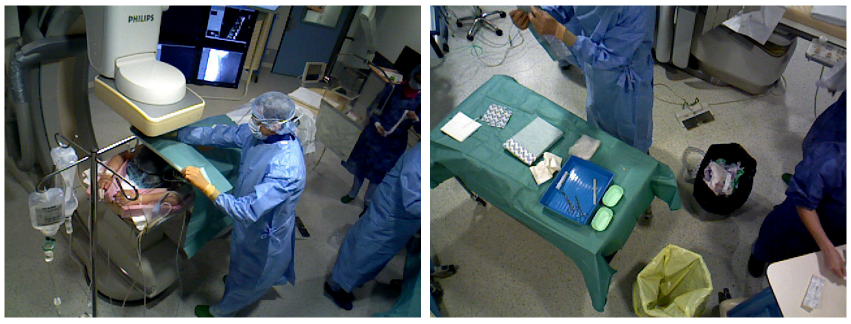
5.2.2 Experimental Setup

5.2.2.1 SurgActionMultiView3D Dataset

To validate our method, we perform surgical action recognition on recordings obtained in the C-arm room. These recordings are captured at 14 fps. Upon further observation, we observe that this temporal resolution is sufficient for the STIP and DSTIP detectors to perform reliably. This dataset is generated from 11 full-day recordings of the OR. It is annotated with 15 actions, containing general actions (e.g. bed entering, moving patient to the OR bed) and vertebroplasty-specific actions (e.g. hammering, cleaning patient's back). The final dataset, referred to as *SurgActionMultiView3D*, contains 1734 action-labeled video clips, totaling over 7.5 hours of recordings which is equivalent to 2×375k frames. Frame samples from some actions are shown in Figure 5.6. The details and statistics of the actions can be found in Table 5.1. The top part of the table contains actions which are captured from the first view and the bottom part from the second view. Note that even though the camera views are disjoint and since most actions are only captured by one camera, the other camera can also provide some discriminative information. For example, the action of putting sterile drape is observed by the first camera, however the second camera captures the movement of the clinician taking the sterile drape from the equipment table.

Action	#Inst.	Avg. duration (s)
Adjusting C-arm	507	14.5 ± 24.3
Adjusting OR bed	144	8.8 ± 4.9
Bed entering	59	15.8 ± 7.0
Bed leaving	61	19.9 ± 9.8
Cement injection	29	374.4 ± 105.1
Cleaning patient's back	245	20.0 ± 10.4
Hammering	216	15.8 ± 7.2
Monitor adjustment	136	12.6 ± 7.7
Moving patient from OR bed	31	15.0 ± 5.1
Moving patient to OR bed	30	20.2 ± 8.4
Putting sterile drapes	126	10.5 ± 4.4
Removing sterile drapes	42	17.4 ± 9.0
Mixing cement	28	75.9 ± 21.5
Opening package	51	12.7 ± 5.4
Throwing package	29	10.2 ± 4.0

Table 5.1: The list of actions to recognize in the SurgActionMultiView3D dataset, including the number of instances per action and the length of the video clips (mean $\pm$ std) in second.


(a) Putting sterile drape

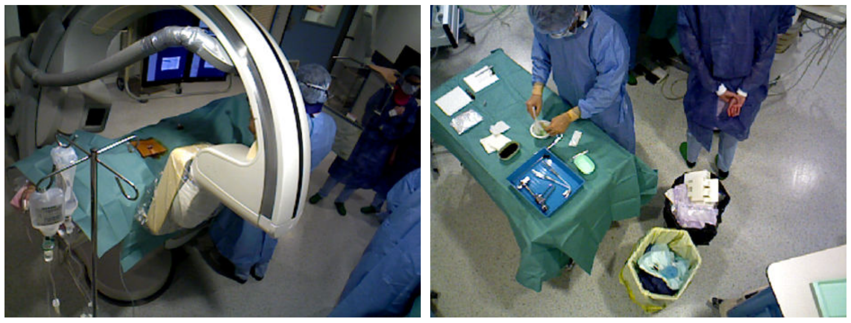

(b) Mixing cement

Figure 5.6: Frame samples of actions: (a) putting sterile drape and (b) mixing cement.

5.2.2.2 Feature Extraction and Encoding

Visual dictionary. Since we are working with a multi-modal multi-view camera system, each action is recorded in four video clips (two intensity and two depth video clips). From each of the intensity clips, we detect 1000 STIPs and extract 480-dimensional HOF feature; while from each of the depth clips, we detect 1000 DSTIPs and extract 4005-dimensional DCSF. We reduce the dimension of the visual features using Principal Component Analysis (PCA), shrinking the HOF and DCSF feature dimensions to 250 and 800, respectively. We build the visual dictionaries using the features with reduced dimensions. To obtain the optimal dictionary size, we use gradually increasing number of words $K_a \in \{100, 500, 1000, 1500\}$ to build the visual dictionary. To obtain the feature representation from the intensity clips, we collect the interest points extracted from both intensity clips and encode them with the corresponding visual dictionary. The same process is performed to obtain the representation from the depth clips. To get the final representation of the video clips, we concatenate the features coming from both views.

Here, we first investigate the performance of different features without incorporating the spatio-temporal feature encoding (i.e. by using only the visual dictionary). We compare the classification results of methods using the intensity information alone [Dollár 2005], the depth information alone [Xia 2013], and the combination of both. The feature combination is obtained by concatenating intensity and depth features.

Spatial and spatio-temporal dictionaries. To show the importance of the temporal information in the non-rigid layout encoding process, we build separately a 3D spatial dictionary and a 4D spatio-temporal dictionary. The 3D spatial dictionary is built using the 3D spatial information (x, y, z) of the interest points; while the 4D spatio-temporal dictionary is built using the 4D spatio-temporal information (x, y, z, t).

For both spatial and spatio-temporal dictionaries, we construct a two-level resolution dictionaries ($L = 2$). For the 3D spatial dictionary, we perform comparison with the multi-resolution rigid 3D spatial grid, where $K_{st}^1 = 2^3$ and $K_{st}^2 = 4^3$. Essentially, the rigid spatial grid splits equally the 3 axes in the 3D space into 2 and 4 regions. Ultimately, we compare the performance of the 4D spatio-temporal dictionary to the multi-resolution 4D spatio-temporal grid of sizes: $K_{st}^1 = 2^3 \times 3$ and $K_{st}^2 = 4^3 \times 3$, resulting in a 3 equally-spaced areas in the temporal axis. Note that these layout configurations are chosen in order to match the number of words in our spatio-temporal dictionaries (shown in Table 5.2b).

All dictionaries are trained using 5 random instances of every action type from the dataset. The summary of all dictionaries is presented in Table 5.2.

5.2.2.3 Classification

Due to the imbalanced number of instances in our dataset, we perform a bagging approach to train the classification model. We subsample our data into a bag and limit the number of instances from each class so that each bag contains balanced data. We repeat the process to get 5 bags. We then learn a classification model from each bag and average

Feature	Dim.	# words
HOF-STIP	250	100,500,1K,1.5K
DCSF-DSTIP	800	

(a) Visual

Dictionary	Size	
Rigid spatial	$l = 1$	$2 \times 2 \times 2$
	$l = 2$	$4 \times 4 \times 4$
Non-rigid spatial	$l = 1$	8
	$l = 2$	64
Rigid spatio-temporal	$l = 1$	$2 \times 2 \times 2 \times 3$
	$l = 2$	$4 \times 4 \times 4 \times 3$
Non-rigid spatio-temporal	$l = 1$	24
	$l = 2$	192

(b) Spatial and spatio-temporal

Table 5.2: Details of: (a) visual and (b) spatial and spatio-temporal dictionaries.

the models to get the final SVM classifier. In our experiments, we use Chi-Square (χ^2) and histogram intersection (HI) kernels as non-linear classifiers. We observe the accuracy of our method by using a five-fold cross validation.

5.2.3 Experimental Results

5.2.3.1 Visual Feature Type Comparison

In Fig. 5.7, we show interest points (STIPs and DSTIPs) detected from the intensity and depth sequences. We can see that they have complementary characteristics. For example, in *cleaning patient's back* action, the DSTIPs rarely appear on the arm of the clinician due to the low depth difference between the clinician's arm and the patient's back. On the other hand, the STIPs appear on these locations, due to a significant difference on intensity between the arm's edge and the patient's back. This can also be observed on the action of *throwing away package*. These results are expected because, by definition, the STIP detector looks for "corners" on the spatio-temporal space of the intensity, thus captures the interest points on the clinician's arm and the package on the table. On the other hand, the DSTIP detector observes the depth data, thus captures mostly interest points on the edges between foreground objects and the background. However, it is important to note that these DSTIPs are not detected because of the noise in depth images for these DSTIPs are detected where the activities are actually happening, not at the edges of random objects.

The classification accuracies for all dictionary-kernel configurations are reported in Table 5.3. It is intuitive that there is an optimal number of words for every feature. If the dictionary size is too small, the histogram loses its discriminative power; if the dictionary size is too large, the histograms from the same action will not match. We can see that the best number of words for intensity features is 1500, while the best number for depth and combined features is 1000. We tested the intensity features with a dictionary of 2000 words, but it did not yield better results.

To show the improvements that the combination of features brings, we present the

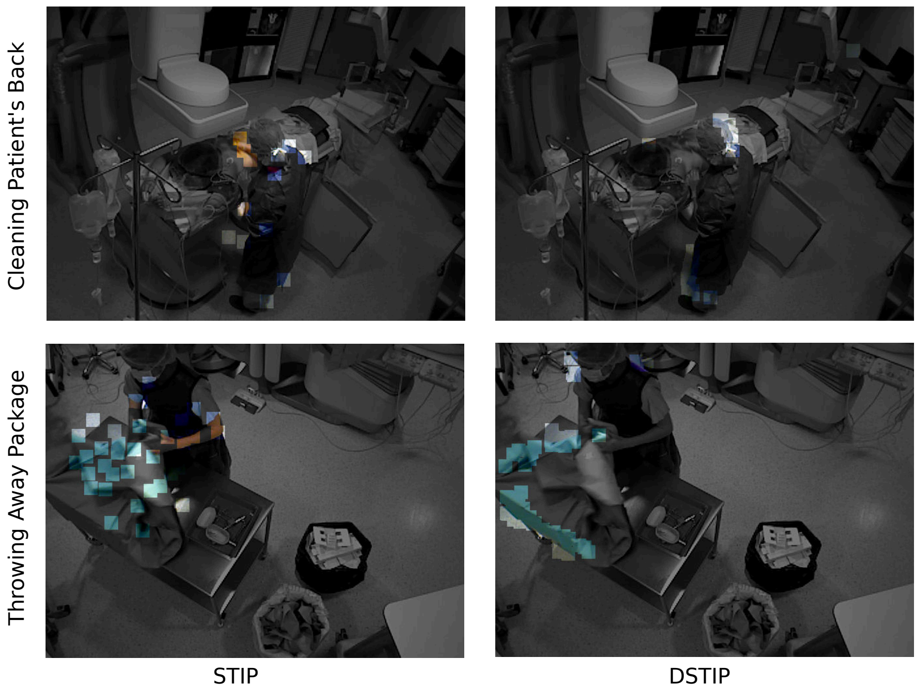


Figure 5.7: Shown in highlighted color rectangles are the projections of the detected STIPs and DSTIPs on the intensity images for the action of cleaning patient's back (view 1) and throwing away package (view 2).

Feature	# words	Kernel	
		HI	χ^2
Intensity [Dollár 2005]	100	59.9	60.4
	500	68.2	66.8
	1000	67.9	67.4
	1500	**69.6**	68.6
Depth [Xia 2013]	100	50.5	52.8
	500	60.7	61.6
	1000	63.7	**64.5**
	1500	64.2	64.4
Combination	100	70.2	70.0
	500	75.9	76.8
	1000	76.6	77.1
	1500	76.6	**77.2**

Table 5.3: Classification accuracies using solely visual dictionaries for multiple dictionary-kernel setups. The best dictionary-kernel setups for every feature are shown in bold.

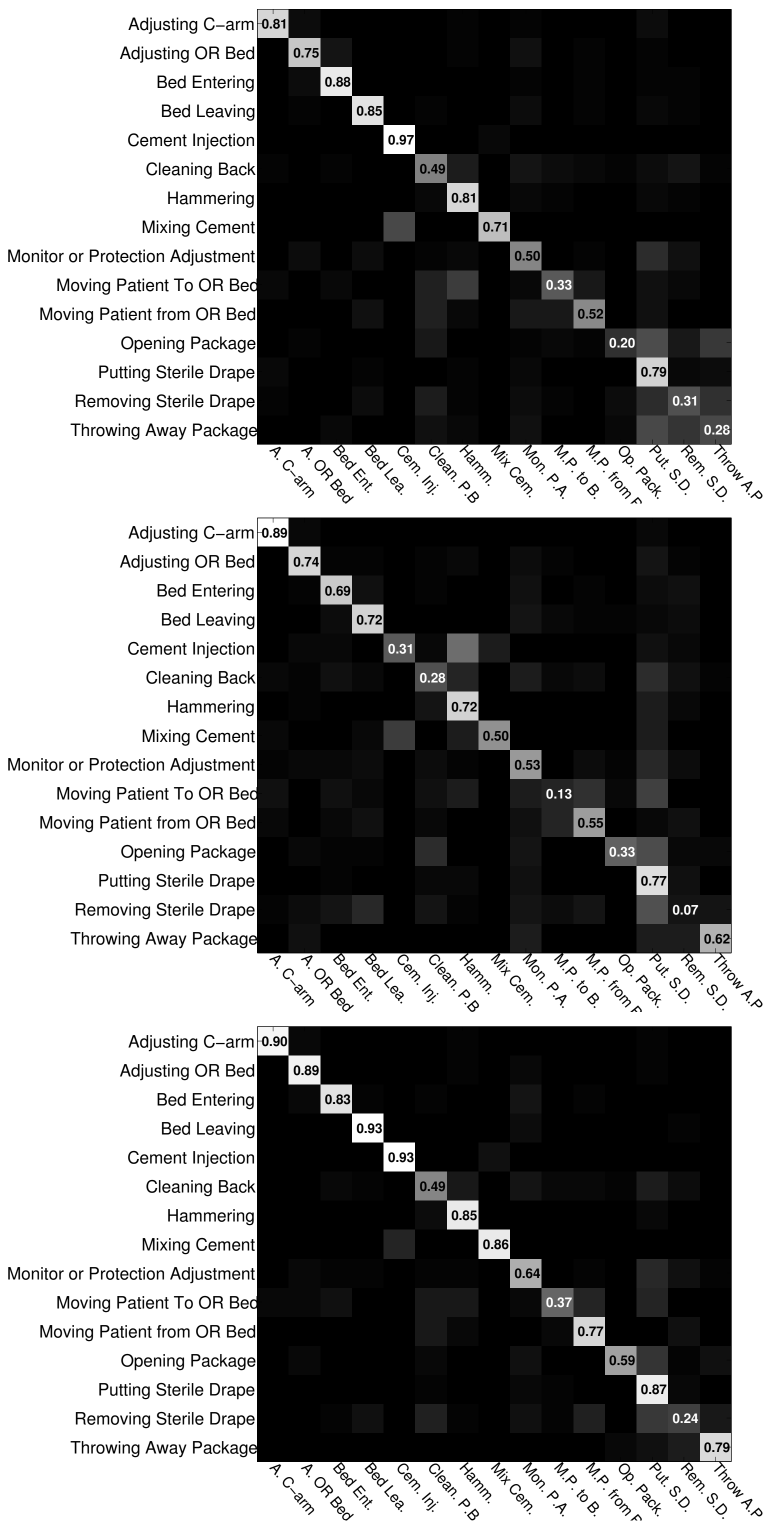


Figure 5.8: Confusion matrices for a classification configuration using visual dictionaries of 1500 words with χ^2 kernel on: (a) intensity, (b) depth, and (c) the combination of both.

Feature	Visual only	Resolution	3D Spatial		4D Spatio-Temporal	
			Rigid	Non-Rigid	Rigid	Non-Rigid
Intensity, 1500-HI	69.6	$L = 1$	76.3	77.2	75.0	80.4
		$L = 2$	77.2	80.3	75.1	**80.5**
Depth, 1000-χ^2	64.5	$L = 1$	71.5	76.9	73.1	78.4
		$L = 2$	75.4	79.5	75.5	**79.6**
Combination, 1500-χ^2	77.2	$L = 1$	81.4	83.3	81.1	**85.5**
		$L = 2$	83.0	85.1	81.5	**85.5**

Table 5.4: Comparison of classification accuracies without the spatio-temporal feature encoding, with the (rigid and non-rigid) 3D spatial encodings and with the (rigid and non-rigid) 4D spatio-temporal encodings, in both single and multi-resolution scenarios. The best classification accuracy for each feature is shown in bold.

confusion matrices for the 1500-χ^2 (i.e. 1500-word dictionary on χ^2 kernel) configuration setup on all the features in Figure 5.8. We can see from the confusion matrices that the intensity and depth information complement each other. For some actions, such as *throwing away package* and *opening package*, the depth information gives more discriminative features. However, for other actions, for example *cement injection*, the intensity information is more discriminative. With this observation, we combine both features by concatenating them to build a better classification model. The feature combination leads to the significant increase of recall for most actions, except for *bed entering*, *cement injection*, and *removing sterile drapes*. This decrease is, however, to be expected. By concatenating the intensity and depth features, we increase the dimensionality of the features without necessarily introducing new information. In other words, the depth and the intensity features could be correlated. Thus, in some cases, the combination could impede the effectiveness of the classifier in choosing the optimal parameters to carry out the classification. However, the decrease is insignificant compared to the increase and is solved once the non-rigid spatio-temporal feature encoding is introduced.

5.2.3.2 Spatio-Temporal Feature Encoding

To show how much the spatio-temporal feature encoding can improve the classification accuracies, we evaluate its performance on the best dictionary-kernel configuration for each feature type, i.e. 1500-HI for intensity, 1000-χ^2 for depth, and 1500-χ^2 for the combination of intensity and depth. The classification accuracies are shown in Table 5.4. We can see that by only incorporating the 3D spatial layout encoding, the classification accuracies increase significantly. The highest improvement is achieved on depth data, where the rigid 3D spatial layout gives over 7% improvement and the non-rigid gives higher improvement of more than 12% in accuracy. However, when we introduce the temporal information into the encoding process, the rigid layout drops the classification accuracies for intensity and the combination of features. This can be caused by our argument that with the rigid layout, many features might fall into the same cell and

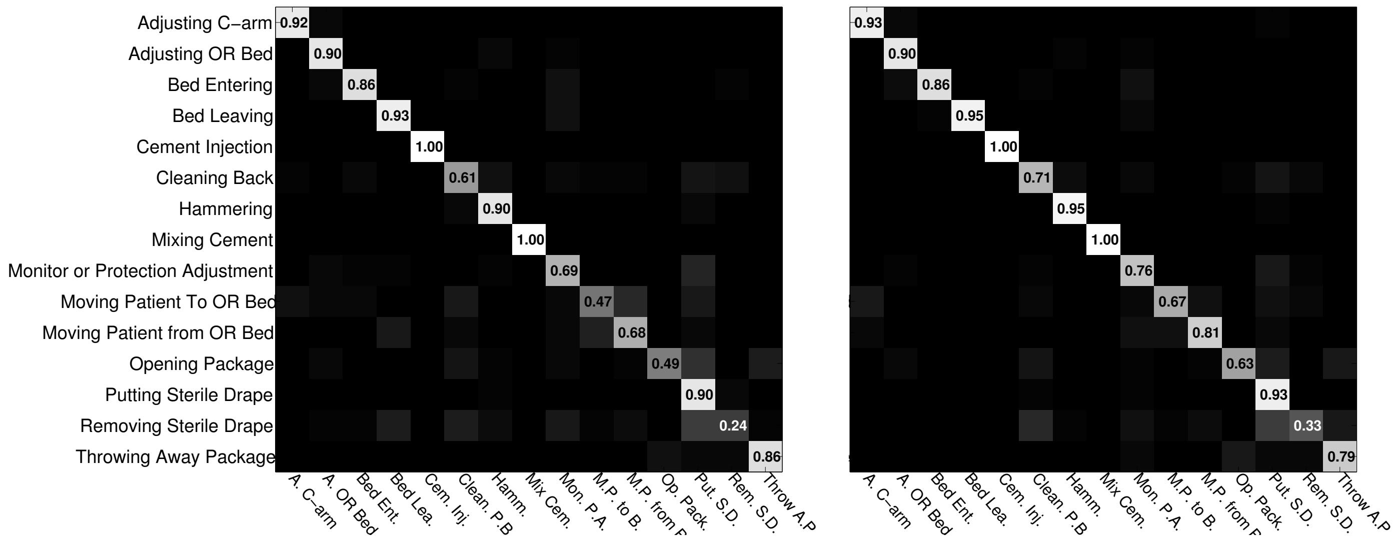


Figure 5.9: Confusion matrices for classification using the combination of depth and intensity features on 1500-χ^2 configuration with: (a) the rigid and (b) the non-rigid 4D spatio-temporal layout encodings in single resolution.

impede the advantage of introducing the layout encoding scheme. Especially in a big 4D space, it is hard to properly define the boundaries of the grid layout by hand. In contrast, the non-rigid counterpart further improves the classifier performance, resulting in the highest accuracy of 85.5%. The confusion matrices in Figure 5.9 depict the improvements that the non-rigid 4D spatio-temporal layout encoding yields for each class compared to the rigid counterpart.

We also perform the task using multi-resolution spatial and spatio-temporal dictionaries. In Table 5.4, it is shown that the performance is improved by the multi-resolution encoding approach for both rigid and non-rigid cases. However, we can see that even in the multi-resolution case, the rigid grid does not outperform our proposed encoding method in single resolution. For example, using the depth features, the multi-resolution 4D spatio-temporal rigid grid yields 75.5% while the single-resolution non-rigid encoding yields 78.4%. The fact that our single-resolution proposed encoding method outperforms the multi-resolution rigid encoding approach is a major advantage since the multi-resolution significantly increases the feature dimensionality. For example, using the combination of intensity and depth features encoded with 1500-word dictionaries results in $K_a = 2 \times 1500 = 3K$ (it is multiplied by two since the feature comes from both intensity and depth videos) and the multi-resolution spatio-temporal dictionary is of size $K_{st} = 192 + 24 = 216$, thus the final feature size is $3K \times (216 + 1) = 651K$. In summary, our proposed encoding approach works much better in retaining the spatial and spatio-temporal information than the SPM rigid approach, without the need to significantly augment the feature dimension.

5.2.4 Conclusions

Here, we have presented a pipeline to perform surgical action classification on OR-scene RGBD videos. We have also presented a novel feature encoding method which learns the layout of the spatial and spatio-temporal spaces instead of dividing them with rigid grids. To evaluate the pipeline, we have generated the SurgActionMultiView3D dataset containing more than 1700 video clips of 15 different actions. The experimental results show that the intensity and depth videos have complementary characteristics and the best classification performance is obtained when the visual features from both intensity and depth videos are combined. The results also demonstrate that the proposed feature encoding approach generates better features than the ubiquitous SPM approach [Lazebnik 2006].

Since the surgical action classification consists of classifying pre-segmented video clips into action classes, the recognition pipeline does not need to enforce the temporal constraints of the surgical workflow. Having observed the success of the method in performing the surgical action classification task, we are interested in investigating whether the approach would generalize to frame-wise classification where a temporal model needs to be enforced. To this end, we present in the following section, the surgical phase recognition task performed on the OR-scene RGBD videos.

5.3 Surgical Phase Recognition

Here, we address surgical phase recognition performed on the OR-scene RGBD videos. Specifically, we focus on vertebroplasty procedures. Since this procedure contains critical phases, the automatic surgical phase recognition could be used to construct a system to send automatic notifications to senior surgeons during the execution of these critical phases. It would also be useful for postoperative applications, such as video indexing and surgical training. For instance, a young surgeon, who needs to learn how a certain phase is executed, is just one click away from finding the phase without the need to scroll through the complete video.

This task is similar to the one presented in Section 4.3. For this reason, we are using the same recognition pipeline. In addition, since the encoding approach proposed in Section 5.2 has been shown to work very well to generate discriminative features on the OR-scene RGBD videos, the phase recognition pipeline presented here also incorporates the approach to encode the visual features.

5.3.1 Methodology

The pipeline to perform surgical phase recognition on OR-scene videos consists of the following steps: (1) visual feature extraction, (2) feature encoding, and (3) classification.

ID	Phase	Avg. duration (min.)
Q1	Patient preparation	13.9±4.7
Q2	Sterilizing	11.5±2.8
Q3	Needle insertion	16.6±8.0
Q4	Cement mixing	4.3±1.1
Q5	Cement injection	7.9±2.1
Q6	Wound dressing	3.5±1.0
Q7	Image acquisition	1.9±0.7
Q8	Patient leaving	2.8±0.7
Complete Surgery		61.4±17.2

Table 5.5: List of phases in the C-arm dataset, including the mean ± std of the duration of each phase in seconds.

5.3.1.1 Feature Extraction and Encoding

Since we are working with spatio-temporal features, the visual feature are extracted from image sequences, instead of single images. Thus, to perform *frame-wise* surgical phase recognition on a vertebroplasty video using such features, we divide the video into clips in such a way that each clip contains 5-second image sequence with one second overlap between each other.

From each sequence, we first extract STIPs and DSTIPs from intensity and depth sequences, respectively, which are then described by HOF and DCSF as presented in Section 5.2.1.2. Then, we aggregate the extracted features from the sequences using the feature encoding approach presented in Section 5.2.1.3.

5.3.1.2 Phase Recognition Pipeline

Here, the recognition pipeline consists of two classification algorithms: a Support Vector Machine (SVM) and a Hierarchical Hidden Markov Model (HHMM). The multi-class SVM is designed to take the feature representations to compute the confidence values representing the probability of the sequences belonging to the phases. To enforce the temporal constraints, we use use a two-level Hierarchical HMM. The top-level contains states that model the surgical phases and their transitions, while the bottom-level nodes model the intra-phase dependencies. All these parameters (except the number of top-level states which is set to the number of classes) are data-driven and automatically learnt from the data. The observations are given by the confidence values obtained from the SVM.

5.3.2 Experimental Setup

5.3.2.1 Dataset

Here, the dataset used to perform the surgical phase recognition contains 14 recorded vertebroplasty procedures, equivalent to over 13 hours of procedures. Recall that the

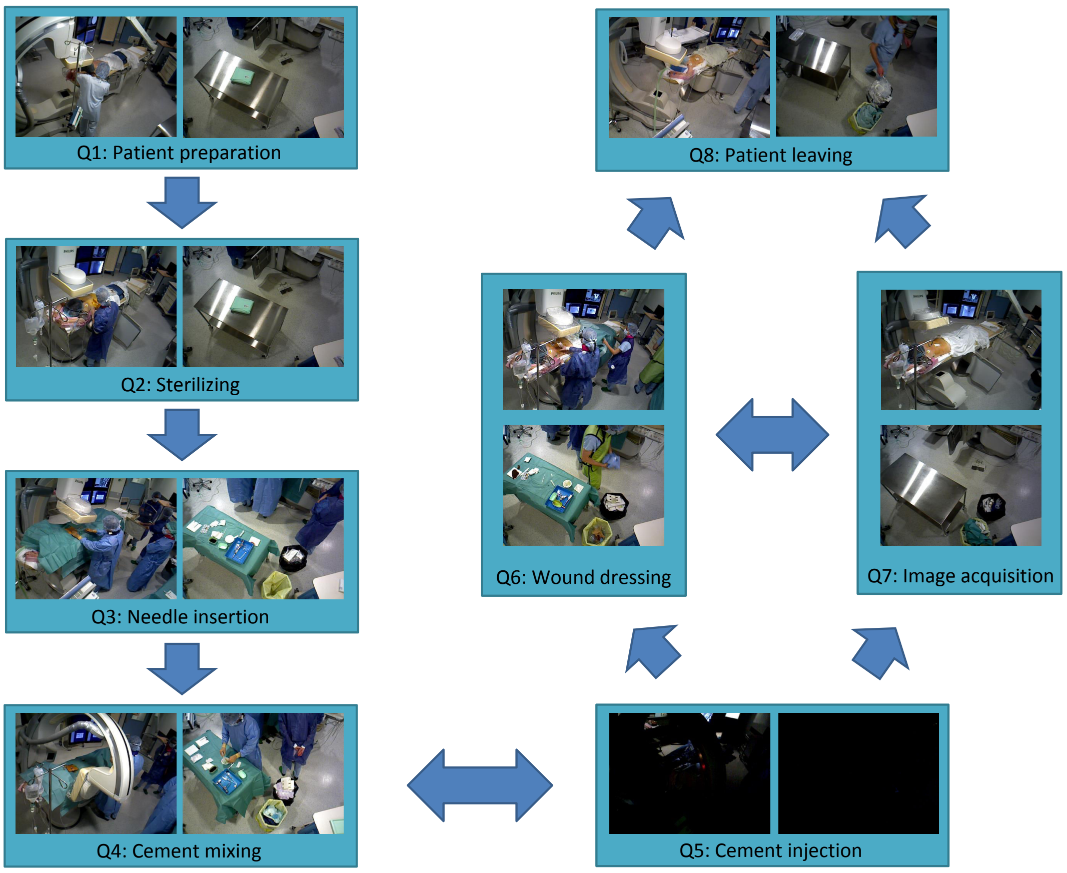


Figure 5.10: Surgical phase transitions of vertebroplasty procedures in the C-arm dataset.

dataset contains multi-modal multi-view recordings, thus all the procedures are recorded in four synchronized videos (two intensity and two depth videos). Therefore, in practice, the total duration of the recorded videos are four times the length of the procedures, i.e., over 50 hours of recordings.

After a discussion with a senior clinician, we define eight surgical vertebroplasty phases. The list and the statistics of the phases can be found in Table 5.5. In Fig. 5.10, we show color frame samples from each phase and illustrate the transitions between phases generated from our dataset. It can be seen that during the *cement injection* phase, the color video frames are very dark. This phase is the most crucial one in vertebroplasty because the cement might leak to other anatomical structures of the patients, which might lead to fatal outcomes. Therefore, the surgeons need to be very precise during this phase. By switching the lights off in the OR, the surgeons can see a better contrast on the monitor which displays the fluoroscopic images visualizing the progress of cement injection into the patients' vertebrae.

5.3.2.2 Feature Extraction and Encoding

From each 5-second intensity sequence, we detect 250 STIPs and extract 480-dimensional HOF feature; while from each 5-second depth sequence, we detect 250 DSTIPs and extract 4005-dimensional DCSF. We reduce the dimension of the visual features using Principal Component Analysis (PCA), shrinking the HOF and DCSF feature dimensions to 250 and 800, respectively. We build the visual dictionaries using the features with reduced dimensions. For both intensity and depth features, we build visual dictionaries of size $K_a = 1500$. To obtain the final representation, we concatenate the visual features from the two views.

As for the non-rigid layout, we do not use the spatio-temporal dictionary since the features are extracted from 5-second sequences which are generated from the video. Thus, the temporal placement of the features is not of significance during the feature encoding. We also do not use a multi-resolution spatial dictionary since it has been shown in Section 5.2.3.2 that our proposed feature encoding already performs very well in single-resolution. In addition, working in multi-resolution significantly increases the feature dimension. Therefore, we use the spatial dictionary of size $K_{st} = 8$.

5.3.2.3 Evaluation

The evaluation metrics used for this task are precision, recall, and accuracy. These are identical to the metrics used to measure the system's performance for surgical phase recognition on laparoscopic videos, as presented in Section 4.3.3.5. These metrics are shown for recognition before and after applying the HHMM.

5.3.3 Experimental Results

In Table 5.6, we show the recognition results before applying the HHMM. It can be seen that the results of intensity features are better than the ones of the depth features. In addition, the combination of the features yield better results than the intensity and depth features individually. Furthermore, when the encoding approach is applied on the features, the recognition results are improved. This finding conforms to the trend that is also found in the results of surgical action classification shown in Table 5.3 and 5.4.

Recall that the results shown in Table 5.6 are obtained without incorporating any temporal constraint, thus they are not the final results from the pipeline. We show the results after applying the HHMM in Table 5.7. Note that it is natural that the offline results are always better than the online counterparts since in offline mode, the algorithm recognize the phases in a video using the complete video, while it is not the case in online mode.

In Table 5.7, we can see that the HHMM significantly improves the recognition results for all features. One might observe that the proposed encoding approach does not yield significant improvements to the results. This might be due to the fact that for surgical phase recognition, the enforcement of temporal information (via the HHMM) is more important than retaining the spatial information from the 5-second clips using the spatial

Feature	Precision(%)	Recall(%)	Accuracy(%)
STIP	53.6±5.6	54.0±6.1	57.4±8.7
DSTIP	51.2±9.3	55.0±10.2	56.0±11.8
Combination	58.1±7.6	62.1±8.9	64.9±9.8
STIP + Enc.	57.8±8.6	56.4±7.8	63.7±8.7
DSTIP + Enc.	54.0±8.7	57.5±10.6	60.3±11.5
Combination + Enc.	**63.8±7.8**	**67.2±8.3**	**69.6±9.0**

Table 5.6: Phase recognition results before applying the HHMM (mean±std) on the C-arm dataset. The best result for each evaluation metric is shown in bold.

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
STIP	83.9±18.1	83.3±19.1	85.0±22.1	74.5±16.1	75.1±15.8	79.6±18.8
DSTIP	83.4±14.6	83.6±15.6	85.8±18.3	77.4±16.3	80.5±15.6	80.4±19.5
Combination	85.0±18.0	86.3±18.2	87.2±21.9	78.5±18.0	81.0±17.8	81.3±21.6
STIP + Enc.	80.0±15.8	81.4±16.4	84.5±22.1	74.5±19.9	75.9±17.4	80.9±20.7
DSTIP + Enc.	85.3±16.9	85.7±17.5	85.9±22.3	77.6±17.0	80.3±16.9	81.9±21.5
Combination + Enc.	**86.7±18.7**	**86.0±17.7**	**87.9±22.2**	**80.5±17.8**	**82.2±16.7**	**82.5±21.1**

Table 5.7: Phase recognition results after applying the HHMM (mean ± std) on the C-arm dataset. The best result for each evaluation metric is written in bold.

dictionary. In spite of this, one can see that the encoding approach applied on top of the combination of intensity and depth features yields the best phase recognition results.

5.3.4 Conclusions

Here, we have addressed the surgical phase recognition task on a multi-modal multi-view dataset using the recognition pipeline proposed in Section 4.3. Into the pipeline, we incorporated the feature encoding approach proposed in 5.2 to retain the spatial information from the visual features. To evaluate the method, we perform the surgical phase recognition task on a dataset containing 14 vertebroplasty surgeries. The experimental results show that both intensity and depth features contain discriminative information to perform the task and that the combination of intensity and depth features outperforms the individual visual features. Using the proposed encoding approach on top of the combination of intensity and depth features, we obtain a high offline accuracy of 87.9%.

5.4 Summary

In this chapter, we have presented a novel feature encoding method designed to retain spatio-temporal information from the visual features. We have incorporated this feature encoding method into the recognition pipelines to address two tasks: surgical action classification and surgical phase recognition on the OR-scene RGBD videos. The former has as objective to identify the actions performed in pre-segmented video clips, while the latter is a frame-wise classification task where the goal is to recognize the surgical phases of vertebroplasty procedures. Evaluated on real recordings of vertebroplasty procedures, we have shown that the combination of the handcrafted spatio-temporal visual features and the proposed feature encoding method yields satisfactory results for both surgical action classification and surgical phase recognition tasks on the multi-modal multi-view videos. To the best of our knowledge, this work is the first to address such tasks on multi-view RGBD recordings.

The tasks presented in this chapter are performed using *handcrafted* visual features. These handcrafted features are specifically designed to extract information from the color and depth images *separately*. Due to the visual challenges that are inherent in the OR-scene videos, these handcrafted features might not capture discriminative information from the data, even though these features yield very good results for both surgical action classification and surgical phase recognition. In Chapter 6, we propose to use deep learning algorithms to automatically learn the visual features from the data to perform surgical phase recognition on the OR-scene RGBD videos. We design a deep architecture which shares a feature representation across the image modalities. Therefore, the features from color and depth images are extracted simultaneously using a single network. In addition, we present another dataset containing multi-view RGBD recordings from the CT room. This dataset will be used to evaluate the generalizability of the proposed network.

6 Feature Learning for Surgical Phase Recognition

It is still magic even if you know how it is done.
– Terry Pratchett

Chapter Summary

In Chapter 4 and 5, we have presented recognition pipelines for pre-segmented and frame-wise surgical activity recognition tasks on laparoscopic and RGBD videos, respectively. To perform these tasks, we used several *handcrafted* visual features, such as color features, intensity gradients, and spatial and spatio-temporal interest points, extracted from the videos since they have been shown to yield satisfactory results for surgical activity recognition [Blum 2010, Lalys 2012, Zappella 2013, Dergachyova 2016]. Our results also show that the handcrafted visual features contain discriminative properties to perform surgical activity recognition. However, handcrafted features are manually engineered to extract certain characteristics from the data, which might lead to information loss during the feature extraction process. On the one hand, one can minimize this loss by exploiting expert knowledge regarding the data to carefully design other visual features. On the other hand, we are working on videos which contain inherent visual challenges, such as presence of smoke, motion blur and specular reflection on the laparoscopic videos; and object occlusions and invalid depth values on the RGBD videos. Thus, it is difficult to manually engineer discriminative features which extract all the important characteristics from these videos.

Instead of manually designing the visual features, we propose to automatically learn the visual features using deep convolutional neural network (CNN) architectures. We design multiple CNN architectures to perform surgical phase recognition on laparosopic and OR-scene RGBD videos. Since deep CNN models typically contain a high number of unknowns, they require a large training dataset. To alleviate this problem, we take advantage of transfer learning approaches. Here, we will train the proposed networks by finetuning a pre-trained network. Specifically, we finetune the networks using the AlexNet architecture [Krizhevsky 2012] that has been trained using the ImageNet dataset [Russakovsky 2015].

This chapter is structured as follows. In Section 6.1, we describe the AlexNet architecture which we use for the finetuning process. In Section 6.2, we introduce the *EndoNet* architecture which is designed to address recognition tasks on laparoscopic videos. In Section 6.3, we present a multi-stream network to perform surgical phase recognition on RGBD videos. Ultimately, in Section 6.4, we present an end-to-end architecture for surgical phase recognition on laparoscopic videos by connecting the EndoNet network to a long-short term memory (LSTM) network.

6.1 AlexNet Architecture

The AlexNet network is designed to perform image classification on ImageNet. The task involves categorizing images into 1000 classes. In this work, we use a slightly modified AlexNet architecture, which is shown in Fig. 6.1. In the original AlexNet architecture, the network splits the convolution layers into two streams in order to accommodate the delineation of responsibilities between two GPUs. In summary, the architecture consists of an input layer (in green), five convolutional layers (in red, `conv1-conv5`), and three fully-connected layers (in orange, `fc6-fc8`).

The first convolution layer (`conv1`) takes the $224 \times 224 \times 3$ image as input and consists of 96 kernels of size $11 \times 11 \times 3$ with a stride of 4 pixels, where stride is the distance

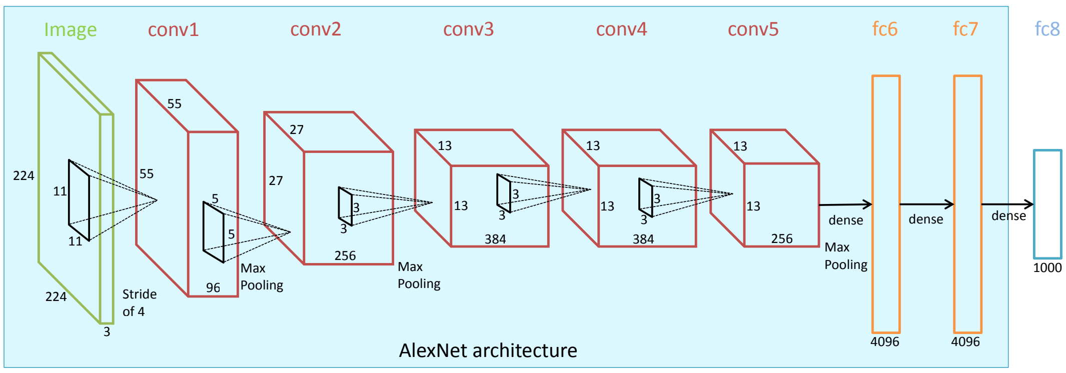


Figure 6.1: Illustration of the modified AlexNet architecture [Krizhevsky 2012].

between the centers of neighboring neurons in a kernel map. After being max-pooled (with filter size 3×3) and response-normalized, the output of the first layer is passed to the second convolutional layer (`conv2`), which consists of 256 kernels of size 5×5. The output of the second layer is also max-pooled and response-normalized. The third, fourth, and fifth convolution layers are connected to one another without any intervening pooling or normalization layers with the following number of parameters: 384 kernels of size 3×3 (`conv3`), 384 kernels of size 3×3 (`conv4`), and 256 kernels of size 3×3 (`conv5`), respectively. The output of the fifth layer is then max-pooled and connected to a fully-connected layer (`fc6`), containing 4096 neurons. The next fully-connected layer (`fc7`) also contains 4096 neurons. The output of these fully-connected layers are applied to a dropout layer, with 50% dropout rate. The output of `fc7` is passed to the last fully-connected layer (`fc8`), consisting of 1000 neurons, each of which represents the confidence of the image belonging to a class.

6.2 Deep Architectures for Multiple Tasks on Laparoscopic Videos

In Section 4.3, we have shown the results of surgical phase recognition on cholecystectomy videos using a pipeline based on handcrafted visual features. Here, we propose to automatically learn the discriminative visual features using deep architectures and use them to perform the phase recognition task.

Based on our results shown in Section 4.3.4 as well as the findings reported in [Padoy 2012, Stauder 2014, Dergachyova 2016], the binary tool features contain valuable information for the surgical phase recognition task on laparoscopic videos. However, these features are obtained through a manual annotation process, which makes it impractical to use at test time. Thus, in order to alleviate the need for manual annotation, we propose to jointly perform surgical phase recognition and *tool presence detection* by solely using the visual information from the videos. Here, we design a multi-task CNN architecture, that we call *EndoNet*, to carry out the tasks jointly.

The objective of the tool presence detection task is to provide a binary information denoting the presence of all tools of interest, which are shown in Fig. 4.9. In addition to helping in the learning of the discriminative visual features, the tool presence detection task itself is also interesting to perform because it could be exploited for many applications, for instance to automatically index a surgical video database by labeling the tool presence in the videos. Combined with other signals, it could also be used to identify a potential upcoming complication by detecting tools that should not appear in a certain phase. It is important to note that this task does not require tool localization, thus it differs from the usual tool detection task [Sznitman 2014]. In addition, the tool presence is solely determined by the visual information from the laparoscopic videos. Thus, it does not result in the same tool information as the one used in [Padoy 2012, Stauder 2014, Dergachyova 2016], which cannot always be obtained from the laparoscopic videos alone. For example, the presence of trocars used in [Padoy 2012] is not always apparent in the laparoscopic videos. Automatic presence detection for such tools would require another source of information, e.g., an external video.

6.2.1 Related Work on Tool Presence Detection

As previously demonstrated, the tool information is a discriminative feature which can be used to perform surgical phase recognition on laparoscopic videos. However, the literature addressing the problem of automatic tool *presence* detection in the CAI community is still limited. The approaches typically focus on other tasks, such as tool detection [Sznitman 2014, Bouget 2015, Hajj 2016], tool pose estimation [Allan 2015], and tool tracking [Reiter 2012, Rieke 2015]. In addition, most of the methods are only tested on short sequences, while we carry out the task on recordings of complete procedures.

In [Kranzfelder 2013, Neumuth 2012], a system using radio frequency identification (RFID)-tagged surgical tools has been proposed for tool detection and tracking. Such an active tracking system can be used to solve the tool presence detection problem, but this system is complex to integrate into the OR. In addition, since we focus on using visual features from the videos, we are particularly interested in vision-based approaches. For instance, in [Speidel 2009], a method to automatically recognize the types of tools that appear in laparoscopic images was presented. However, the method consists of many steps, such as tool segmentation and contour processing. In addition, it also requires the 3D models of the tools to perform the tool categorization. A more recent work [Lalys 2012] proposed to use an approach based on the Viola-Jones object detection framework to automatically detect the tools in cataract surgeries, such as the knife and Intra Ocular Lens instruments. However, the tool detection problem on laparoscopic videos poses other challenges that do not appear in cataract surgeries where the camera is static and the tools are not articulated. In [Primus 2016], the visual features from laparoscopic videos, which mainly consist of color features, are used to detect the location and the type of the surgical tools. These tool features are then used to recognize the phases. However, we observe from our data that the color features are not very reliable for the tool detection task. In addition, there is no clear evaluation in [Primus 2016] regarding the performance of the tool recognition as well as the phase recognition pipelines.

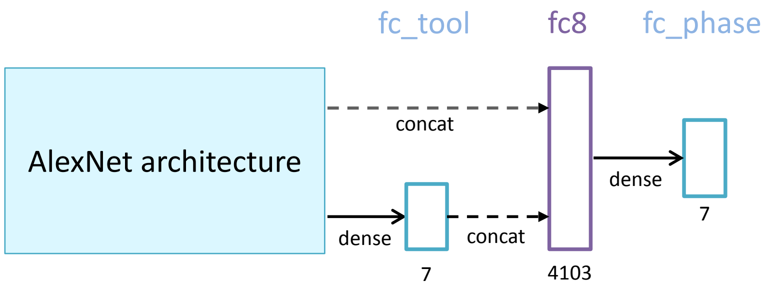


Figure 6.2: EndoNet architecture designed for phase recognition and tool presence detection tasks on laparoscopic videos. The AlexNet architecture is shown in Fig. 6.1.

6.2.2 CNN Architecture

Here, we design a multi-task CNN architecture based on the following hypothesis:

- more discriminative features for the phase recognition task can be learnt from the dataset if the network is fine-tuned in a multi-task manner, i.e., if the network is optimized to carry out not only phase recognition, but also tool presence detection;
- since the tool signals have been successfully used to carry out phase recognition, the inclusion of automatically generated tool detection signals in the final feature can improve the recognition.

The proposed EndoNet architecture is shown in Fig. 6.2.

In EndoNet, the output of layer `fc7` from AlexNet is connected to a fully-connected layer `fc_tool`, which performs the tool presence detection. Since there are seven tools defined in the dataset used to train the network, the layer `fc_tool` contains 7 nodes, where each node represents the confidence for a tool to be present in the image. This confidence is later concatenated with the output of layer `fc7` in layer `fc8` to construct the final feature for phase recognition. Ultimately, the output of layer `fc8` is connected to layer `fc_phase` containing 7 nodes, where each node represents the confidence that an image belongs to the corresponding phase.

The network is trained using stochastic gradient descent with two loss functions defined for the tasks. The tool presence detection task is formulated as N_t binary classification tasks, where $N_t = 7$ is the number of tools. For each binary classification task, the cross-entropy function (see Eq. 3.24) is used to compute the loss. Thus the final loss function $\mathcal{L}_{\mathcal{T}}$ for the tool presence detection task is simply the summation of the cross-entropy losses of all tools.

Here, phase recognition is regarded as a multi-class classification task. The softmax multinomial logistic function, which is an extension of the cross-entropy function, is utilized to compute the loss. The function is formulated as:

$$\mathcal{L}_{\mathcal{P}} = \frac{-1}{N_i} \sum_{i=1}^{N_i} \sum_{p=1}^{N_p} l_p^i \log \left(\varphi \left(w_p^i \right) \right), \tag{6.1}$$

where $p \in \{1, \ldots, N_p\}$ is the phase index and $N_p = 7$ is the number of phases, $l_p^i \in \{0, 1\}$ and w_p^i are respectively the ground truth of the phases and the output of layer `fc_phase` corresponding to phase p and image i, and $\varphi(\cdot) \in [0, 1]$ is the softmax function.

The final loss function is the summation of both losses: $\mathcal{L} = a \cdot \mathcal{L}_{\mathcal{T}} + b \cdot \mathcal{L}_{\mathcal{P}}$, where a and b are weighting coefficients. In this work, we set $a = b = 1$ as in preliminary experiments, the best results are obtained using these parameters. One should note that assigning either $a = 0$ or $b = 0$ is equivalent to designing a CNN that is optimized to carry out only the phase recognition task or the tool presence detection task, respectively.

6.2.3 Phase Recognition Pipeline

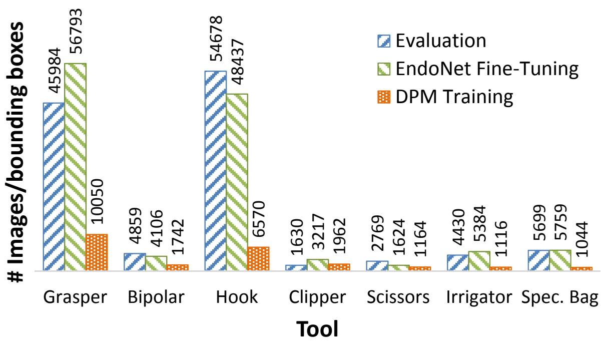
The phase recognition pipeline is identical to the one described in Section 4.3.2.2. We use a one-vs-all multi-class SVM and a Hierarchical Hidden Markov Model (HHMM). Here, the difference is in the image features that are used to perform the recognition. Using the EndoNet architecture, instead of the handcrafted visual features, the output of layer `fc8` is taken to represent each of the video frames.

One can observe that EndoNet already provides confidence values through the output of layer `fc_phase`, thus it is not essential to pass EndoNet features to the SVM to obtain the confidence values $\mathbf{v}_p$. Furthermore, in preliminary experiments, we observed that there was only a slight difference of performance between $\mathbf{v}_p$ and `fc_phase` in recognizing the phases both before and after applying the HHMM. However, this additional step is necessary in order to provide a fair comparison with other features, which are passed to the SVM to obtain the confidence. In addition, using the output of layer `fc_phase` as the phase estimation confidence is only applicable to datasets that share the same phase definition as the one in the fine-tuning dataset. Thus, this step is also required for the evaluation of the network generalizability to other datasets that might have a different phase definition, such as the EndoVis dataset.

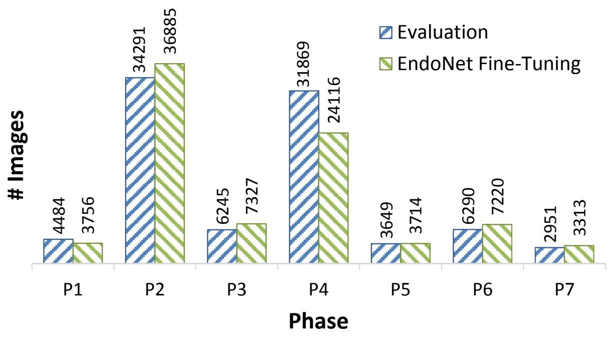
6.2.4 Experimental Setup

6.2.4.1 Datasets

We use the datasets described in Section 4.3.3.1 and 4.3.3.2, i.e., the Cholec80 and EndoVis datasets. The Cholec80 dataset is split into two subsets of equal size (i.e., 40 videos each). The first subset (i.e., the finetuning subset) contains ~86K annotated images (at 1 fps). This subset is used to finetune the AlexNet network. From this finetuning subset, 10 videos have also been fully annotated with the bounding boxes of tools. These bounding boxes are used to train Deformable Part Models (DPM) [Felzenszwalb 2010], which will be used as a baseline for tool presence detection. Because the grasper and hook appear more often than other tools, their bounding boxes reach a sufficient number from the annotation of three videos. The second subset (i.e., the evaluation subset) is used to test the methods for both tool presence detection and phase recognition. The dataset is split equally in order to provide enough data for both the fine-tuning and evaluation processes. The statistics of the complete dataset can be found in Fig. 6.3. The EndoVis



(a)



(b)

Figure 6.3: Distribution of annotations in the Cholec80 dataset for (a) tool presence detection and (b) phase recognition tasks.

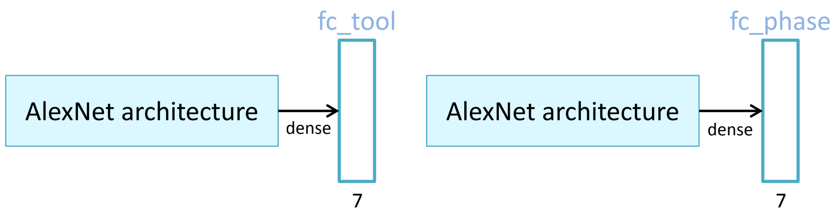
dataset is used solely for evaluation purposes, thus the network is finetuned only using the finetuning subset of Cholec80.

6.2.4.2 Parameters

EndoNet is trained by fine-tuning the publicly available AlexNet network which has been pre-trained on the ImageNet dataset. The layers that are not defined in AlexNet (i.e., `fc_tool` and `fc_phase`) are initialized randomly. The network is fine-tuned for 50K iterations with $N_i = 50$ images in a batch. The learning rate is initialized at 10^{-3} for all layers, except for `fc_tool` and `fc_phase`, whose learning rate is set higher at 10^{-2} because of their random initialization. The learning rates for all layers decrease by a factor of 10 for every 20K iterations. The fine-tuning process is carried out using the Caffe framework [Jia 2014].

The networks are trained using an NVIDIA GeForce Titan X graphics card. The training process takes ~80 seconds for 100 iterations, i.e., roughly 11 hours per network. The feature extraction process takes approximately 0.2 second per image. The computational time for SVM training depends on the size of the features, ranging from 0.1 to 90 seconds, while the HHMM training takes approximately 15 seconds using our MATLAB implementation.

To carry out phase recognition, all features are passed to a one-vs-all *linear* SVM, except the handcrafted features, which are passed through a histogram intersection kernel beforehand. We tried to use non-linear kernels for other features in our preliminary experiments, but this did not yield any improvements. The HHMM is set up identically as described in Section 4.3.3.4. We set the number of top-level states to seven, while the number of bottom-level states is data-driven (as in [Padoy 2009]). To model the output of the SVM, we use a mixture of five Gaussians for every feature, except for the binary tool signal, where one Gaussian is used. The type of covariance is diagonal.



(a) ToolNet (b) PhaseNet

Figure 6.4: Single-task network architectures: (a) ToolNet and (b) PhaseNet. The AlexNet architecture is shown in Fig. 6.1.

6.2.4.3 Baselines

For tool presence detection, we compare the results given by EndoNet (i.e., the output of layer `fc_tool`) with two other methods. The first method is DPM [Felzenszwalb 2010], since it is an ubiquitous method for object detection that is available online. In the experiments, we use the default parameters, model each tool using three components and represent the images using HOG features. The second method is a network trained in a single-task manner that solely performs the tool presence detection task (ToolNet). ToolNet is fine-tuned using the same parameters as the ones mentioned in Section 6.2.4.2. We compare the ToolNet results with the EndoNet results in order to show that performing the fine-tuning process in a multi-task manner yields a better network than in a single-task manner. The architecture of this network can be seen in Fig. 6.4-a.

For phase recognition, we compare the phase recognition results from the features described in Section 4.3.3.3 with the learnt features from the network. The features are as follows:

- binary tool information generated from the manual annotation;
- handcrafted visual features: bag-of-word of SIFT, HOG, RGB and HSV histograms;
- the handcrafted visual features + CCA, similar to the approach suggested in [Blum 2010]. The approach uses the binary tool information only at training time to obtain the projection matrix;
- the output of layer `fc7` of AlexNet trained on the ImageNet dataset (i.e., the initialization of the fine-tuning process); this is an interesting feature to compare to, because it has been shown that this network can be used for various computer vision tasks [Donahue 2013];
- the output of layer `fc7` from a network that is fine-tuned to carry out phase recognition in a single-task manner, shown in Fig. 6.4-b (PhaseNet); PhaseNet is also fine-tuned using the parameters mentioned in Subsection 6.2.4.2;
- our proposed features, i.e., the output of layer `fc8` from EndoNet (see Fig. 6.2).

6.2.4.4 Evaluation

Tool presence detection and phase recognition are two different classification tasks. The former is a binary classification task, while the latter is a multi-class one. Therefore, different evaluation metrics are used. The performance of the tool presence detection is measured by the average precision (AP) metric. It is obtained by computing the area under the precision-recall curve. For phase recognition, we use the following evaluation metrics: precision, recall, and accuracy as described in Section 4.3.3.5.

For tool presence detection, the learnt models (i.e., DPM, ToolNet, and EndoNet) can be directly used to perform the task without any additional training step. In contrast, the phase recognition task requires an additional step to train the SVM and HHMM after the networks are trained. Because of this reason, the tool presence detection task is performed directly on the evaluation subset of Cholec80; while for phase recognition, we run a 4-fold cross-validation on the evaluation subset of Cholec80 and a full cross-validation on the EndoVis dataset. For each validation fold, we perform five experimental runs and average the evaluation metrics over all experimental runs.

6.2.5 Experimental Results

6.2.5.1 Network Activation Visualization

To better understand what the network has learnt, we show in Fig. 6.5 the activations of the five convolution layers in EndoNet obtained from three image samples. At the early stage of the network, we can see that the filters could be used to segment the anatomical structure. By looking at the index of filters (bottom row of every image) from layer `conv1`, we can see that there is a group of filters that yield the highest activations for the anatomical structure. For example, the fat tissue in the first and second images is well “detected” by the filter depicted in red and the gallbladder in the third image by the filter depicted in orange.

Intuitively, as the network goes deeper, the feature representations are mixed together to a higher abstraction and it gets harder to interpret the extracted visual features. However, we can observe that as the network goes deeper, it is capable to distinguish the tools in the image. This is shown by the high activations on the regions where the tools are. For example, the activations in the first image are high around the grasper and the activations in the second image are high around the tip of grasper. Indeed, this shows the capability of deep architectures to automatically learn the discriminative visual features. It is interesting to see that, even though we did not specifically inform the network about the visual appearance of the surgical tools and we did not design the network to localize the tools, the network learnt that the surgical tool is a discriminative feature for the tasks and also learnt to localize the surgical tools in the images.

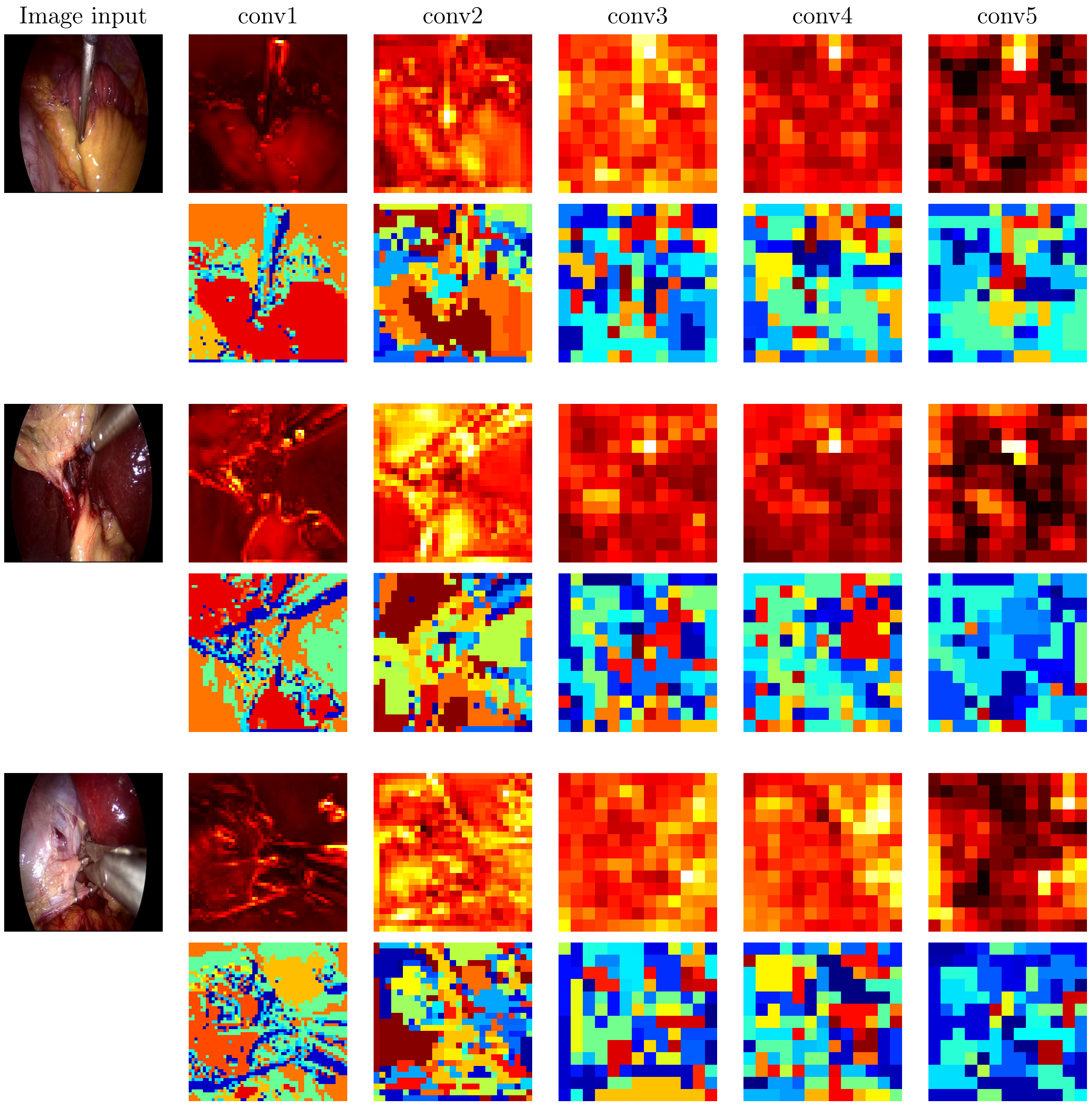


Figure 6.5: Visualization of activations of layer `conv1` to `conv5` of EndoNet obtained from three image samples in the Cholec80 dataset. At the top row, we show the best filter activations after each convolution layer. Light colors represent high activations. At the bottom row, we show the index of the filters which give the highest activation. Each color represents different filter index.

Tool	DPM	ToolNet	EndoNet
Bipolar	60.6	85.9	**86.9**
Clipper	68.4	79.8	**80.1**
Grasper	82.3	84.7	**84.8**
Hook	93.4	95.5	**95.6**
Irrigator	40.5	73.0	**74.4**
Scissors	23.4	**60.9**	58.6
Specimen bag	40.0	86.3	**86.8**
MEAN	58.4	80.9	**81.0**

Table 6.1: Average precision (AP) for all tools, computed on the 40 videos forming the evaluation dataset of Cholec80. The best AP for each tool is written in bold.

6.2.5.2 Results on Cholec80 Dataset

Tool Presence Detection. The results of the tool presence detection task are shown in Table 6.1. It can be seen that the networks yield significantly better results than DPM. It might be due to the fact that the number of images used for fine-tuning the networks is higher than the number of bounding boxes used for DPM training, but this may only partly explain this large difference. To provide a fairer comparison, we compare the performance of DPM with ToolNet and EndoNet models that are trained only with the 10 videos used to train DPM (see Fig. 6.8). The mean APs of these models are still better than the one of DPM: 65.9 and 62.0 for ToolNet and EndoNet, respectively. Note that, the networks are only trained using binary annotations (present vs. not-present), while DPM uses bounding boxes containing specific localization information. Furthermore, the networks contain a much higher number of unknowns to optimize than DPM. In spite of these facts, with the same amount of training data, the networks perform the task better than DPM.

From Table 6.1, it can be seen that EndoNet gives the best results for the tool presence detection task. This shows that training the network in a multi-task manner does not compromise the EndoNet's performance in detecting the tool presence. For any method, one can observe that the scissors are not detected as well as other tools. This might be due to the fact that this tool has the smallest amount of training data (see Fig. 6.3-a), as it only appears shortly in the surgeries.

Over the seven tools and 40 complete surgeries in the evaluation subset of Cholec80, EndoNet obtains 81% mean AP for tool presence detection. The success of this network suggests that binary annotations are sufficient to train a model for this task. This is particularly interesting, since tagging the images with binary information of tool presence is much easier than providing bounding boxes. It also shows that the networks can successfully detect tool presence without any explicit localization pre-processing steps (such as segmentation and ROI selection).

Surgical Phase Recognition. In Table 6.2, the results of phase recognition on Cholec80 before applying HHMM are shown. These are the results after passing the image features to the SVM. One might notice that the recognition results using handcrafted features are not identical with the ones shown in Section 4.3.4.1, this is due to the difference in number of videos used to train and test the pipeline. Here, instead of 80 videos, 40 videos (i.e., the evaluation subset) are used to train the phase recognition pipeline (i.e., SVM-HHMM pipeline). However, one can see by comparing the results in Table 4.6 and 6.2 that the trend holds across experiments.

The results in Table 6.2 show that the CNNs are powerful tools to extract visual features: despite being trained on a completely unrelated dataset, the AlexNet features outperform the other baseline features. Once finetuned to our dataset (i.e., PhaseNet), the features yield improvements for all metrics compared to the AlexNet features. In addition to yielding the tool presence detection as a by-product, the multi-task framework applied in EndoNet further improves the features for the phase recognition task. In Table 6.3, the phase recognition results after applying HHMM are shown. By comparing the results from Table 6.2-a and 6.3-a, we can see the improvement that the HHMM brings, which is consistent across all features.

In Fig. 6.6, we show confusion matrices to visualize how well the EndoNet features distinguish the phases from one another. The confusion matrices are generated using the results from one (randomly chosen) experimental run, both before and after HHMM (offline mode). It can be seen that before applying the temporal model, some images are misclassified by the SVM. This is to be expected since the SVM only relies on per-frame visual features and most laparoscopic images look similar to each other. Once the temporal model is incorporated, the recognition results are significantly improved. It can be seen that the phases are typically confused with the neighboring phases. However, the misclassification rates are significantly lower, except for the last few phases, which is due to the non-sequential transitions among these phases (see Fig. 4.8).

In Fig. 6.7, we visualize the confusions temporally on top-5 and bottom-5 recognition results obtained from the same experimental run used to generate Fig. 6.6. In offline mode, it can be seen that the top-5 results are very good, resulting in over 98% accuracies. In addition, the bottom-5 results in offline mode are comparable to the ground truth. The drop of accuracy for the bottom-5 are caused by the jumps that can happen between P5 and P6, which are shown by the alternating blue and red in Fig. 6.7-c. These jumps occur also because of the non-sequential transitions among these phases (see Fig. 4.8). In online mode, one can observe more frequent jumps in the phase estimations. This is due to the nature of recognition in online mode, where future data is unavailable, so that the model is allowed to correct itself after making an estimation. Despite these jumps, the top-5 online results are still very close to the ground-truth, resulting in accuracies above 92%.

6.2.5.3 Results on EndoVis Dataset

Similar results for phase recognition are obtained from the EndoVis dataset, as shown in Table 6.2 and 6.3-b. It can be observed that the improvements obtained by PhaseNet

Feature	Cholec80			EndoVis		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Tool binary	42.8±33.9	41.1±32.3	48.2±2.7	44.3±32.5	48.5±39.3	49.0±9.7
Handcrafted	22.7±28.8	17.9±28.9	44.0±1.8	35.7±6.6	33.2±10.5	36.1±2.6
H.Crafted+CCA	21.9±14.1	18.7±23.3	39.0±0.6	31.1±4.6	31.6±22.6	32.6±5.3
H.Crafted+Tool	51.3±21.7	42.3±27.6	53.9±1.3	56.7±9.8	55.3±32.6	56.4±9.0
AlexNet	50.4±12.0	44.0±22.5	59.2±2.4	60.2±8.0	57.8±9.3	56.9±4.1
PhaseNet	67.0±9.3	63.4±11.8	73.0±1.6	63.5±5.7	63.2±9.3	62.6±4.9
EndoNet	**70.0±8.4**	**66.0±12.0**	**75.2±10.0**	**64.8±7.3**	**64.3±11.8**	**65.9±4.7**

Table 6.2: Phase recognition results before applying the HHMM (mean ± std) on Cholec80 and EndoVis. The best result for each evaluation metric is written in bold.

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	68.4±24.1	75.7±13.6	69.2±8.0	54.5±32.3	60.2±23.8	47.5±2.6
Handcrafted	40.3±20.4	40.0±17.8	36.7±7.8	31.7±20.2	38.4±19.2	32.6±6.4
H.Crafted+CCA	54.6±23.8	57.2±21.2	61.3±8.3	39.4±31.0	41.5±21.6	38.2±5.1
H.Crafted+Tool	79.5±13.5	82.1±10.2	84.9±6.4	67.9±24.4	73.8±10.9	71.4±8.0
AlexNet	70.9±12.0	73.3±16.7	76.2±6.3	60.3±21.2	65.9±16.0	67.2±5.3
PhaseNet	82.5±9.8	86.6±4.5	89.1±5.4	71.3±15.6	76.6±16.6	78.8±4.7
EndoNet	**84.8±9.1**	**88.3±5.5**	**92.0±1.4**	**73.7±16.1**	**79.6±7.9**	**81.7±4.2**

(a) Cholec80

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	81.4±16.1	79.5±12.3	73.0±21.5	80.3±18.1	77.5±18.8	69.8±21.7
Handcrafted	49.7±15.6	33.2±21.5	46.5±24.6	46.6±16.2	48.0±18.5	43.4±21.6
H.Crafted+CCA	66.1±22.3	64.7±22.1	61.1±17.3	52.3±22.2	49.4±21.5	44.0±22.3
H.Crafted+Tool	87.4±13.5	85.5±9.4	82.3±8.9	80.6±13.6	80.1±13.8	77.1±9.3
AlexNet	85.7±13.2	80.8±10.4	79.5±11.0	78.4±14.1	73.9±11.4	70.6±12.3
PhaseNet	86.8±14.2	83.1±10.6	79.7±12.2	79.1±15.0	75.7±15.3	71.0±9.2
EndoNet	**91.0±7.7**	**87.4±10.3**	**86.0±6.3**	**83.0±12.5**	**79.2±17.5**	**76.3±5.1**

(b) EndoVis

Table 6.3: Phase recognition results after applying the HHMM (mean ± std) on: (a) Cholec80 and (b) EndoVis. The best result for each evaluation metric is written in bold.

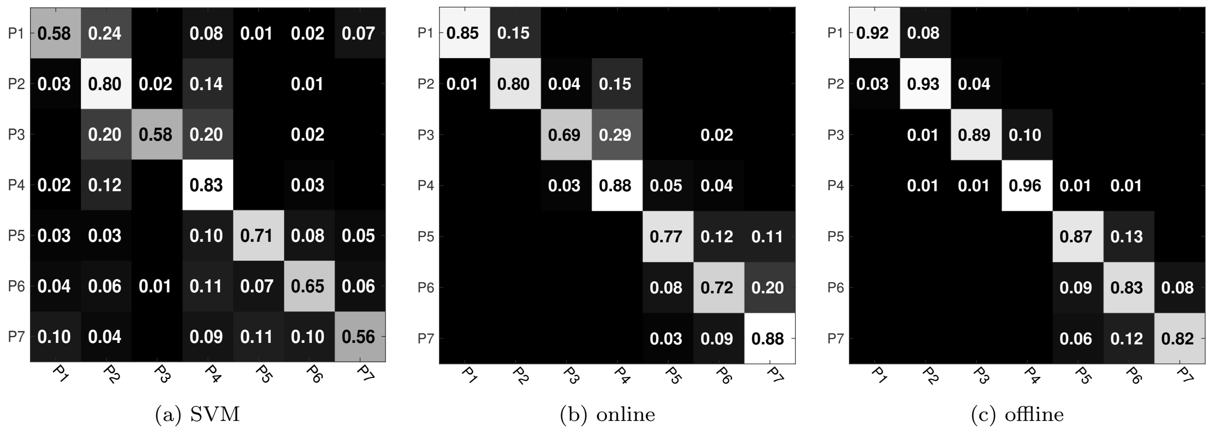


Figure 6.6: Confusion matrices for phase recognition on Cholec80: (a) before HHMM and after HHMM ((b) online and (c) offline modes).

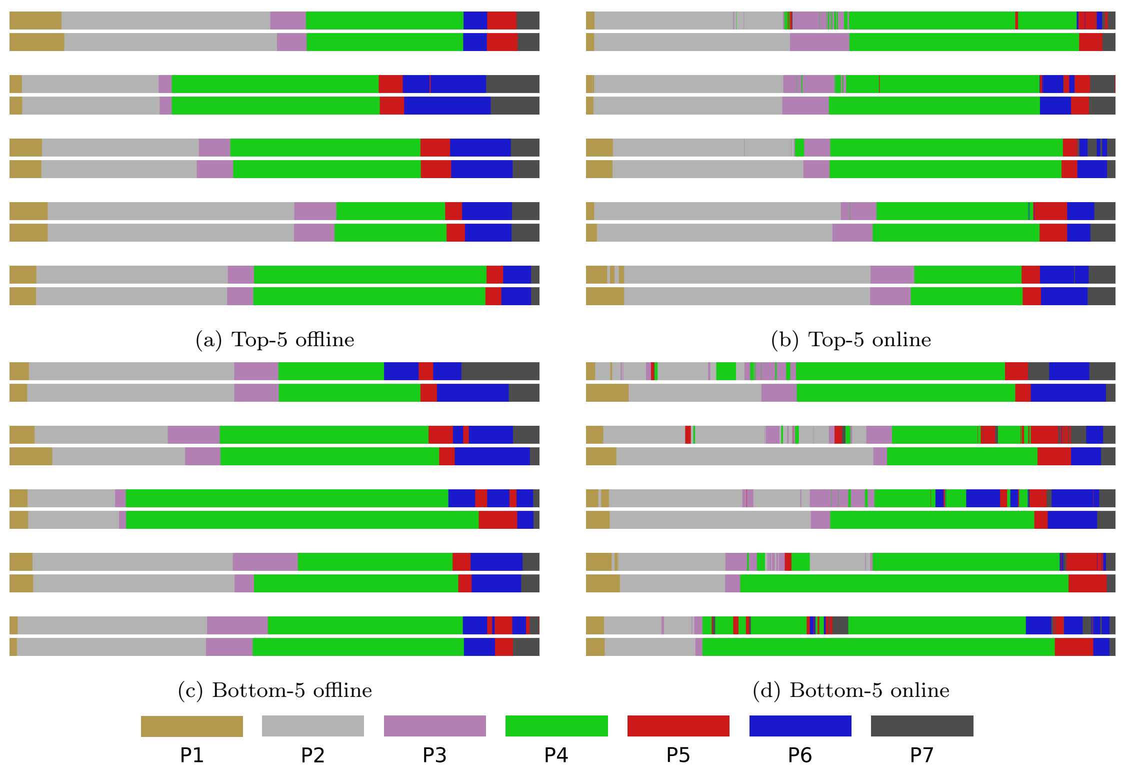
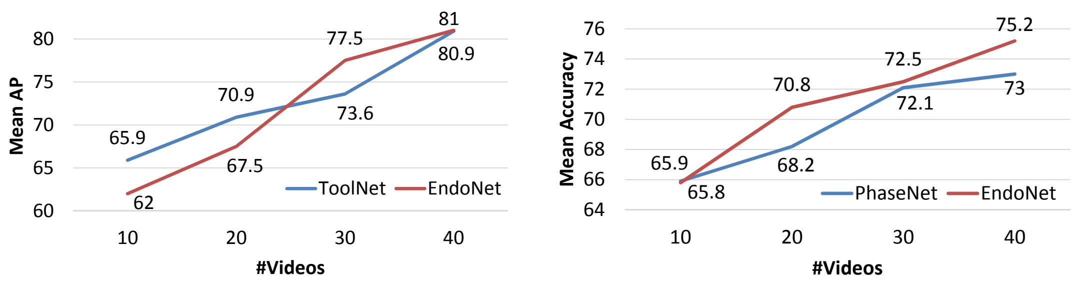


Figure 6.7: Phase recognition results vs. ground truth on the evaluation subset of Cholec80 in a color-coded ribbon illustration. The horizontal axis of the ribbon represents the time progression in a surgery. Since the surgeries are not of the same duration, the temporal axes are scaled for visualization purposes. The top ribbon is the estimated phase and the bottom ribbon is the ground truth.



(a) Tool presence detection

(b) Phase recognition

Figure 6.8: Evolution of network performance on Cholec80 with respect to the number of videos in the fine-tuning subset.

and EndoNet on EndoVis are not as high as the result improvements on Cholec80, which is expected since these networks are fine-tuned using the videos from Cholec80. In spite of this fact, the results on the EndoVis dataset also show that the EndoNet features improve the phase recognition results significantly. It indicates that the multi-task learning generates a better network than the single-task counterpart. The fact that the features from EndoNet yield the best results for all cases also shows that EndoNet is generalizable to other datasets.

One should note that we use the output of layer `fc8` from EndoNet as the image feature, which includes confidence values for tool presence. Because the tools used in EndoVis dataset are not the same tools as the ones in the Cholec80 dataset (which is used to train EndoNet), these confidence values can simply be regarded as 7 additional scalar features appended to the feature vector. The results show that these values help to construct more discriminative features.

6.2.5.4 Effects of Finetuning Subset Size

In order to show the importance of the amount of training data for the fine-tuning process, we fine-tune our networks using fine-tuning subsets with gradually increasing size: 10, 20, 30, and ultimately 40 videos. We perform both tool presence detection and phase recognition tasks on the evaluation subset of Cholec80 using the trained networks. The results are shown in Fig. 6.8. As expected, the performance of the networks increases proportionally to the amount of data in the fine-tuning subset. One might suggest that by putting more videos in the finetuning subset would increase the performance of the networks. However, in this work, we need to keep a high number of videos in the evaluation subset to obtain a statistically sufficient evaluation metrics for the cross-validation evaluation.

6.2.5.5 Clinical Applications

In the previous section, we have discussed tool presence detection results as well as phase recognition results in both offline and online modes. These results already demonstrate

Tolerance (s)	Phase						
	P1	P2	P3	P4	P5	P6	P7
<30	40	34	34	34	40	30	33
30-59	0	0	0	0	0	0	0
60-89	0	4	1	0	0	1	3
90-119	0	0	1	2	0	0	2
≥120	0	2	4	4	0	4	2
TOTAL	40	40	40	40	40	35	40

Table 6.4: Number of phases that are correctly identified in offline mode within the defined tolerance values in the 40 evaluation videos of Cholec80. The number of P6 occurrences is not 40 since not all surgeries go through the cleaning and coagulation phase.

the feasibility of using EndoNet for various medical tasks, such as context-aware support (online recognition) and reporting (offline recognition). Here, we further demonstrate the applicability of EndoNet for other practical CAI applications. First, to show the feasibility of using EndoNet as the basis for automatic surgical video indexing, we show the error of the phase estimation in seconds to indicate how precise the phase boundary estimations from EndoNet are. For this application, we present the results from the same experimental run that is used to generate Fig. 6.7. Second, we investigate further how accurately EndoNet detects the presence of two tools: clipper and bipolar. These tools are particularly interesting because: (1) the appearance of the clipper typically marks the beginning of the *clipping and cutting* phase, which is the most delicate phase in the procedure, and (2) the bipolar tool is generally used to stop haemorrhaging, which could lead to possible upcoming complications.

Automatic Surgical Video Database Indexing. For automatic video indexing, the task corresponds to carrying out phase recognition in offline mode. From the results shown in Fig. 6.7-a,c, one can already roughly interpret how accurate the phase recognition results are. To give a more intuitive evaluation, we present the number of phase boundaries that are detected within defined temporal tolerance values in Table 6.4. We can see that EndoNet generally performs very well for all the phases, resulting in 89% of the phase boundaries being detected within 30 seconds. It can also be seen that only 6% of the phase boundaries are detected with an error over 2 minutes. It is also important to note that this error is computed with respect to the strict phase boundaries defined in the phase annotations. In practice, these boundaries are not as harsh or visually obvious. Thus, this error is acceptable in most cases. In other words, it indicates that the results from EndoNet do not require a lot of corrections, which will make surgical video indexing a lot faster and easier.

Bipolar and Clipper Detection. One of the objectives of performing tool presence detection is to build a system that sends notifications when certain tools, in particular bipolar and clipper, are being used. A reliable system should neither miss nor detect

Tolerance (s)	Bipolar	Clipper
<5	114	49
6-29	9	10
30-59	1	0
≥60	0	1
Missed	0	1
False positives	3.8%	8.3%

Table 6.5: Appearance block detection results for bipolar and clipper, including the number of correctly classified blocks and missed blocks, and the false positive rate of the detection.

the tool presence too early/late. In order to evaluate this reliability, we present another metric to measure the performance of EndoNet in carrying out tool presence detection. First, we define a *tool block* as a set of consecutive frames in which a certain tool is present. Then, we merge the blocks (of the same tool) in the ground-truth data that have a gap less than 15 seconds to compensate the fact that the tools might not always be visible in an image even though they are currently being used. Here, we define a tool block as identified if EndoNet can detect the tool in at least one of the frames inside the block. In this experiment, we determine the tool presence by taking a confidence threshold that gives a high precision for each tool, so that the system can obtain the minimal amount of false positives and retain the sensitivity in correctly detecting the tool blocks. Since the false positive rate is measured using the tool block definition, we also close the gaps between the tool presence detections that are less than 15 seconds.

We show the results of this evaluation in Table 6.5. It can be seen that all the bipolar blocks are detected very well by EndoNet. Over 90% of the blocks are detected under 5 seconds. EndoNet also yields a very low false positive rate (i.e., 3.8% of the blocks in the prediction results are not defined as blocks in ground truth) for the bipolar. This excellent performance is obtained thanks to the distinctive visual appearance that the bipolar has (e.g., the blue shaft). For the clipper, it can be seen that the false positive rate is higher. This could be due to the fact that it has the second lowest amount of annotations in the dataset as a result of appearing shortly in the surgeries (i.e., only during the clipping and cutting phase). However, EndoNet still performs very well for clipper detection, showing that 80% and 97% of the blocks are detected under 5 and 30 seconds, respectively.

6.2.6 Conclusions

Here, we have proposed a multi-task CNN architecture which performs jointly the phase recognition and tool presence detection tasks. The experiments show that this approach overcomes the inherent visual challenges in the laparoscopic videos and subsequently yields visual features that outperform both previously used features and the features obtained from architectures designed for a single task. Interestingly, the EndoNet visual features also perform significantly better in the phase recognition task than binary tool signals indicating which tools can be seen in the image, even though these signals are obtained from ground truth annotations. These results therefore suggest that the images contain additional characteristics useful for recognition in addition to simple tool presence information and that these characteristics are successfully retrieved by EndoNet. Additionally, we have shown that EndoNet also performs well on another dataset recorded in a different hospital, namely EndoVis, and is therefore generalizable.

Having observed that the deep architectures can improve the results of phase recognition on laparoscopic videos, we present in the following section a deep architecture which learns the discriminative visual features from multi-view RGBD videos to perform surgical phase recognition in vertebroplasty procedures.

6.3 Multi-Stream Architecture for Multi-View RGBD Videos

In Section 5.3, we have addressed the problem of surgical phase recognition on RGBD recordings of vertebroplasty procedures using handcrafted visual features. Similarly to Section 6.2, in this section, we propose to automatically learn the discriminative features using deep architectures to perform the phase recognition task.

In the computer vision community, deep learning methods, such as convolutional neural networks (CNNs), have been shown to successfully perform various tasks, such as image classification [Krizhevsky 2012], object detection [Girshick 2014], and activity recognition [Tran 2015]. The methods have also been proven successful at carrying out several tasks on multi-modal data. For example, in [Simonyan 2014a], the combination of RGB and motion features extracted by the CNN demonstrated better performance for activity recognition compared to other networks with single modality. Another work [Wang 2015] proposed to use the combination of RGB and depth features extracted using deep networks. These features are shown to perform better than the individual features for an object recognition task. Inspired by these methods, here, we investigate the usage of deep learning techniques for feature learning on multi-modal multi-view data, by designing a multi-stream CNN architecture which takes color, depth and motion images as input.

6.3.1 CNN Architecture

As previously explained in Chapter 5, we are not only working with RGB images in this work, but also with depth images. Therefore, we here design a network which is optimized to perform surgical phase recognition using both RGB and depth images at the same time. Furthermore, in [Simonyan 2014a], it has been shown that the activity recognition results are improved when motion images are incorporated into the neural network as input. In addition, our previous experiments, shown in Chapter 5, have already demonstrated that the motion information from the scene (i.e., STIPs and DSTIPs) contains discriminative characteristics to perform phase recognition. Thus, in addition to the RGB and depth

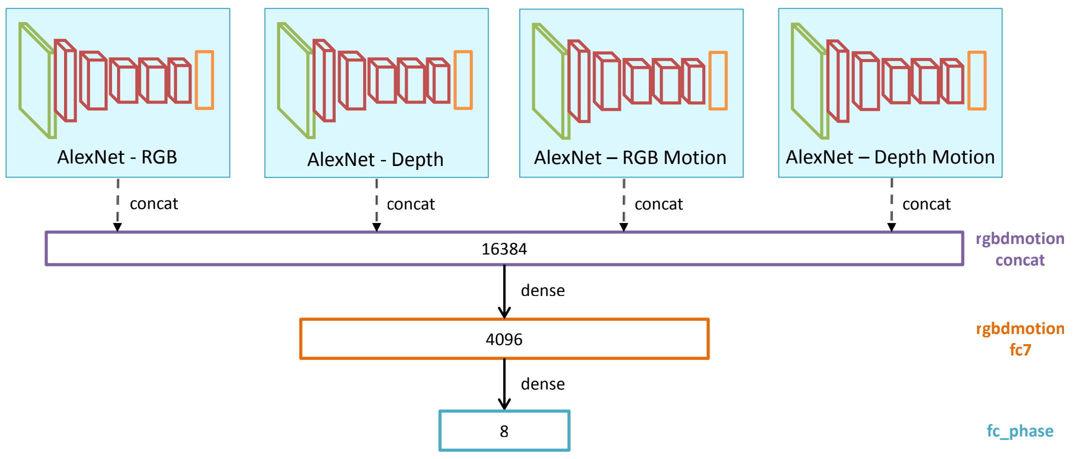


Figure 6.9: Multi-stream CNN architecture.

images, the designed network also takes as input the motion information from motion images. Here, the motion images are obtained from both RGB and depth videos by subtracting the image at time t with image at time $t-1$.

The proposed multi-stream deep architecture is shown in Fig. 6.9. The network takes four streams of input, i.e., RGB, depth, and their corresponding motion images. These networks are connected to a concatenation layer after the first fully-connected layer (i.e., `fc6`). We do not perform the concatenation after layer `fc7` because we want to build a shared feature representation between the image modalities before going to the last layer (i.e., `fc_phase`) without adding another fully-connected layer. Therefore, in our proposed network, each image still undergoes the same process as in AlexNet: five convolution and three fully-connected layers. To handle the multi-view images, this network is trained using images from the two views observing the bed and the equipment table. Once trained, the network is then used to extract the visual features from the images.

As shown in Fig. 6.9, it can be seen that the multi-stream network is extremely large, containing 20 convolution layers and six fully-connected layers. Due to its size, it is difficult to train the network in one finetuning process. In addition, the pre-trained AlexNet model is trained using RGB images, thus the network weights are not optimized for other image modalities. To alleviate this problem, we propose a two-step optimization. First, using a pre-trained AlexNet network, we finetune each network stream separately. We finetune a network solely using the RGB images and repeat the same process for depth, RGB motion, and depth motion images. Second, using the four networks obtained from the first step, we finetune the proposed network. This way, the optimization is not initialized randomly, but instead with four networks independently optimized for the surgical phase recognition task. Therefore, the main objective of the second optimization step is to optimize the weights for layers `rgbdmotionfc7` and `fc_phase`.

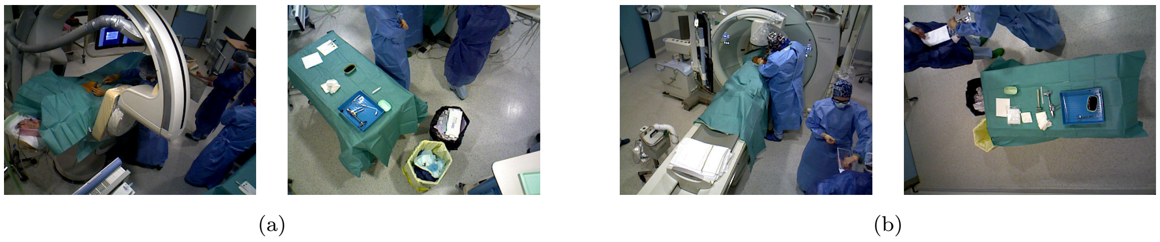


Figure 6.10: Frame samples from: (a) VerCArm24 and (b) VerCT13 datasets.

6.3.2 Phase Recognition Pipeline

We adopt the phase recognition pipeline presented in Section 5.3.1.2, which consists of a multi-class support vector machine (SVM) and a hierarchical hidden markov model (HHMM). Here, the feature representation is taken from the second last layer of the network, i.e., the output of layer `rgbdmotionfc7`. To represent the videos coming from two views, the final feature representation is obtained by concatenating the features extracted from both views.

6.3.3 Experimental Setup

6.3.3.1 Datasets

In order to properly train the neural network, we extend the dataset described in Section 5.3.2.1 by adding 10 videos recorded in the same hybrid OR (i.e., C-arm room), resulting in a dataset of 24 videos. We refer to this dataset as *VerCArm24*. The additional 10 videos are used to finetune the network (i.e., the finetuning subset) while the rest is used as the evaluation subset to train and test the phase recognition pipeline. To test the generalizability of the network, we have also constructed another dataset containing 13 recordings of the vertebroplasty procedures in another operating room (i.e., CT room). We refer to the second dataset as *VerCT13*. All videos in this dataset are used for evaluation. In summary, three subsets are used in this evaluation process: (1) VerCArm24 finetuning, (2) VerCArm24 evaluation, and (3) VerCT13 evaluation. On both evaluation subsets, the method is evaluated using a leave-one-out cross validation. For example, using the VerCArm24 evaluation subset, 13 videos are used to train the SVM-HHMM pipeline and 1 video is used for testing. Results are averaged over all possible video combinations. Due to redundancy and in order to reduce the computational cost, we downsampled the datasets to 1 fps. In Fig. 6.10, we show the views captured by the multi-view system in both ORs. The cameras are configured to observe the OR bed and the equipment table in both ORs.

Both datasets are annotated with 8 surgical phases defined by a senior clinician. The list and statistics of the phases are shown in Table 6.6. Both datasets contain identical phase transitions which are illustrated in Fig. 5.10.

ID	Phase	Length (min.)	
		VerCArm24	VerCT13
Q1	Patient preparation	16.3±10.0	25.9±15.2
Q2	Sterilizing	11.4±3.3	8.8±1.7
Q3	Needle insertion	18.6±13.6	19.8±15.4
Q4	Cement mixing	4.2±1.1	2.5±1.0
Q5	Cement injection	8.5±2.8	13.0±9.9
Q6	Wound dressing	3.6±1.3	3.3±1.3
Q7	Image acquisition	2.1±1.0	2.0±1.2
Q8	Patient leaving	3.0±0.9	4.0±3.2
Complete surgery		68.8±30.9	81.1±48.4

Table 6.6: (a) Phase list and statistics in VerCArm24 and VerCT13 datasets.

6.3.3.2 Parameters

We use the same training parameters for all finetuning processes. Each of them is performed for 30K iterations with a batch size of 50 images. For layers that are already in the pre-trained network, their learning rate is set to 10^{-3}; while other layers which are initialized randomly have higher learning rate of 10^{-2}. The finetuning process is performed using the Caffe framework [Jia 2014].

To carry out the phase recognition task, all features are passed to non-linear kernel SVMs, i.e., the histogram intersection kernel. For the HHMM, we set the number of top-level states to eight (equal to the number of phases), while the number of bottom level states is data-driven. To model the output of the SVM, we use a mixture of five Gaussians with diagonal covariance.

6.3.3.3 Baselines and Evaluation

Here, we compare the phase recognition results obtained from the handcrafted visual features used in Chapter 5 to the ones obtained from the learnt features from the network. The complete list of features is as follows:

- classical bag-of-word (BOW) approach on top of spatio-temporal interest points (STIPs) on color and depth videos, separately as well as their concatenation;
- the extended BOW approach presented in Section 5.2.1.3 on top of STIPs on color and depth videos, separately as well as their concatenation. We use the same parameters for the first two baselines as the ones described in Section 5.3.2.2;
- the output of layer `fc7` of AlexNet trained on the ImageNet dataset (i.e., the initialization of the fine-tuning process);
- the output of layer `fc7` from a finetuned AlexNet (FTAlexNet) using RGB and depth images, separately as well as their concatenation;

- our proposed features, i.e., the output of layer `rgbdmotionfc7`.

To perform the evaluation of the method, we use three evaluation metrics: precision, recall, and accuracy. Due to the stochastic properties of the SVM-HHMM pipeline, we perform the evaluation in 8 experimental runs. The displayed results are obtained by averaging the results over the runs.

6.3.4 Experimental Results

6.3.4.1 Network Activation Visualization

To understand better what the network has learnt, we show in Fig. 6.11 the activations of the five convolution layers of the multi-stream network obtained from three image samples. In contrast to the EndoNet layer activations (shown in Fig. 6.5), the activations from the multi-stream network are much more difficult to interpret. Qualitatively, there is no apparent characteristics that is specifically learnt by the network. It does not necessarily mean that the network does not learn any discriminative features from the dataset. Most probably, this is due to the fact that the OR-scene is a very dense and cluttered environment, therefore the discriminative characteristics to perform phase recognition might not be visible to human eyes, making it difficult to manually engineer the visual features. This hypothesis is supported by the success of the multi-stream network in performing phase recognition on the multi-view RGBD data, as shown in the following sections.

6.3.4.2 VerCArm24

In Table 6.7, we show the results of performing surgical phase recognition on the VerCArm24 dataset. Observing the results before applying the HHMM, one can notice that deep features extracted from AlexNet already outperform the handcrafted features. This shows that the visual features learnt by the network are much more discriminative than the handcrafted ones, despite the fact that the network is trained on a dataset containing completely different images. Once the network is finetuned, the performance of the features (denoted by FTAlexNet) is significantly improved. Interestingly, combining the RGB and depth deep features does not always lead to improvement. This might be due to the fact that the combination is performed through concatenation, which might lead to dimensionality problem for SVM classification. This is however not observed in our proposed network because instead of just concatenating the features, the multi-stream network also computes a shared feature representation among the modalities. As shown in Table 6.7, the best recognition results before applying the HHMM are obtained using our proposed network, yielding an accuracy of 91.3%.

Despite the high performance of the method before applying the HHMM, there is no temporal constraint incorporated into the recognition process. Once the temporal constraints are enforced by the HHMM, the recognition results are further improved. The results after applying HHMM are shown in Table 6.8. Generally, a similar trend is observed

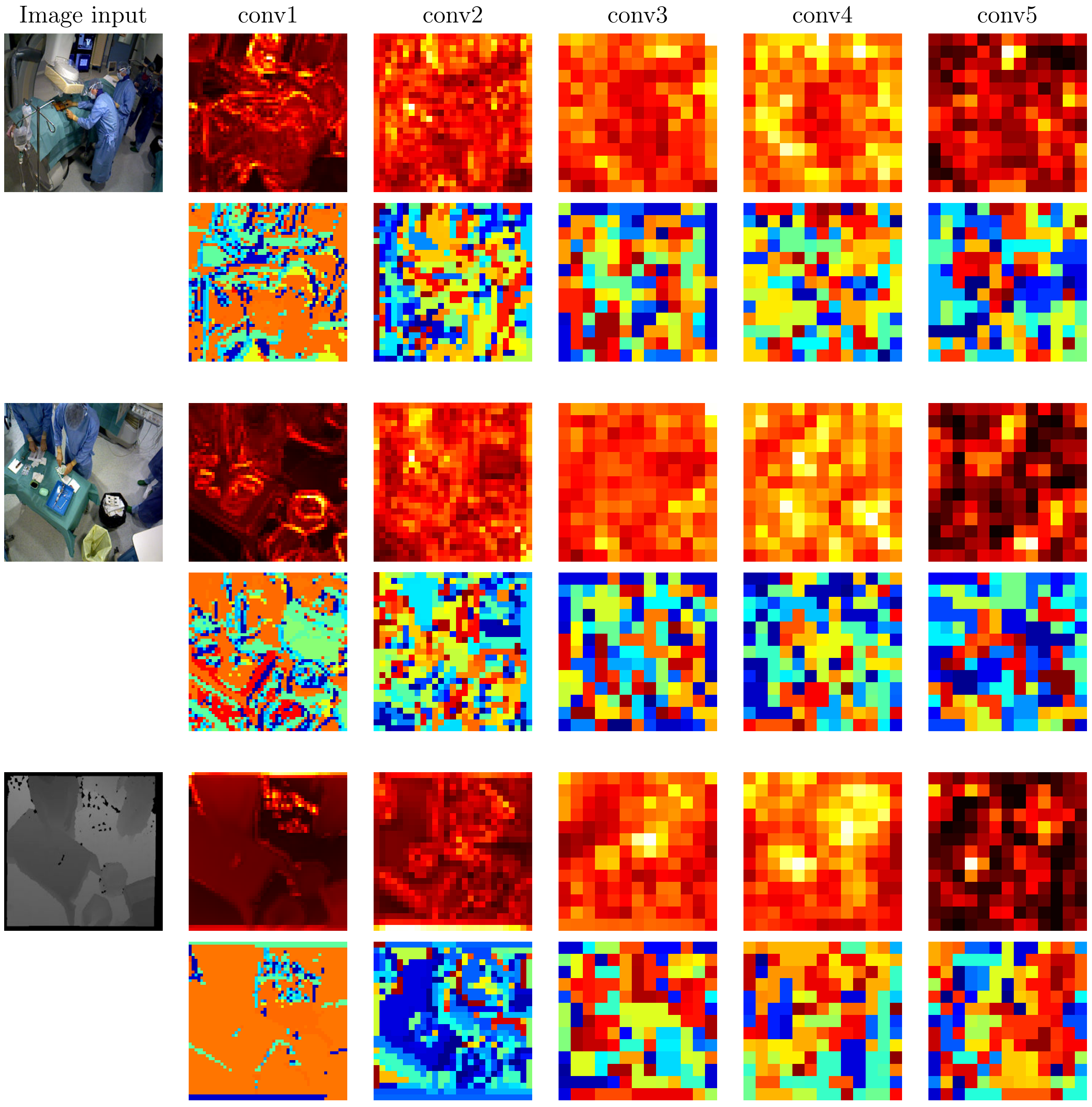


Figure 6.11: Visualization of activations of layer `conv1` to `conv5` of the multi-stream network obtained from three image samples in the VerCArm24 dataset. At the top row, we show the best filter activations after each convolution layer. Light colors represent high activations. At the bottom row, we show the index of the filters which give the highest activation. Each color represents different filter index.

Feature	Precision (%)	Recall (%)	Accuracy (%)
STIP	41.6±5.6	42.0±6.1	45.4±8.7
DSTIP	51.2±9.3	55.0±10.2	56.0±11.8
Comb.	58.1±7.6	62.1±8.9	64.9±9.8
STIP + Enc.	57.8±8.6	56.4±7.8	63.7±8.7
DSTIP + Enc.	54.0±8.7	57.5±10.6	58.3±11.5
Comb. + Enc.	63.8±7.8	67.2±8.3	69.6±9.0
AlexNet-RGB	74.4±0.8	74.5±1.1	77.3±0.9
AlexNet-Depth	60.2±0.9	56.8±1.3	59.6±1.2
AlexNet-RGBD	74.2±0.6	73.9±0.7	78.0±1.1
FTAlexNet-RGB	83.0±8.5	84.5±12.2	87.6±6.6
FTAlexNet-Depth	69.4±7.4	69.9±8.8	74.1±9.7
FTAlexNet-RGBD	82.0±0.4	82.8±0.5	86.2±0.3
Proposed Network	**86.6±5.8**	**88.9±4.2**	**91.3±4.3**

Table 6.7: Phase recognition results *before applying the HHMM* (mean±std) on the VerCArm24 dataset. The best result for each evaluation metric is shown in bold.

Feature	Offline			Online		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
STIP	76.9±18.1	76.3±19.1	79.0±22.1	64.5±16.1	65.1±15.8	66.6±18.8
DSTIP	83.4±14.6	83.6±15.6	85.8±18.3	77.4±16.3	80.5±15.6	80.4±19.5
Comb.	85.0±18.0	86.3±18.2	87.2±21.9	78.5±18.0	81.0±17.8	81.3±21.6
STIP + Enc.	78.0±19.8	77.4±18.4	81.0±22.1	72.5±19.9	72.9±17.4	75.9±20.7
DSTIP + Enc.	85.3±16.9	85.7±17.5	85.9±22.3	77.6±17.0	80.3±16.9	81.9±21.5
Comb. + Enc.	84.7±18.7	86.0±17.7	86.9±22.2	80.5±17.8	82.2±16.7	82.5±21.1
AlexNet-RGB	81.5±4.2	79.6±3.9	82.9±3.1	77.9±4.5	77.5±3.8	79.6±3.2
AlexNet-Depth	77.6±3.0	77.3±2.5	78.9±2.8	73.5±1.8	74.6±1.3	75.4±2.0
AlexNet-RGBD	85.8±1.7	84.4±1.5	86.9±1.8	82.4±1.0	82.3±1.6	83.7±1.3
FTAlexNet-RGB	77.1±30.7	77.9±28.1	81.4±25.5	73.5±30.3	75.3±27.3	78.5±24.8
FTAlexNet-Depth	85.8±10.1	83.9±10.9	86.4±10.1	82.9±8.8	82.3±9.4	83.9±8.9
FTAlexNet-RGBD	88.6±0.6	87.7±0.4	91.2±0.4	84.8±0.7	84.6±0.6	88.1±0.5
Proposed Network	**93.3±6.9**	**91.7±5.8**	**96.0±2.8**	**91.2±5.7**	**89.6±5.7**	**93.7±3.0**

Table 6.8: Phase recognition results *after applying the HHMM* (mean±std) on the VerCArm24 dataset. The best result for each evaluation metric is shown in bold.

Feature	Precision (%)	Recall (%)	Accuracy (%)
AlexNet-RGB	69.2±0.9	69.2±0.9	78.2±0.7
AlexNet-Depth	54.6±0.4	54.5±1.2	58.6±0.6
AlexNet-RGBD	71.4±1.1	71.4±1.1	78.8±1.0
FTAlexNet-RGB	72.7±1.0	74.8±1.4	80.7±1.0
FTAlexNet-Depth	55.6±0.5	56.4±1.0	62.6±1.0
FTAlexNet-RGBD	66.5±0.2	64.5±0.1	81.6±0.3
Proposed Network	**82.2±4.9**	**83.2±6.8**	**89.0±4.7**

Table 6.9: Phase recognition results *before applying the HHMM* (mean±std) on the VerCT13 dataset. The best result for each evaluation metric is shown in bold.

Feature	Offline			Online		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
AlexNet-RGB	76.9±3.0	72.5±2.9	82.9±1.4	71.6±2.6	68.9±2.4	78.9±1.4
AlexNet-Depth	65.3±2.6	64.5±1.6	75.2±2.1	64.0±2.7	63.6±1.7	73.3±1.6
AlexNet-RGBD	83.2±2.8	79.4±2.7	89.1±2.6	78.4±2.7	76.1±2.4	86.4±2.1
FTAlexNet-RGB	76.4±6.9	72.7±5.9	83.0±5.1	71.7±5.7	70.1±5.3	80.7±4.4
FTAlexNet-Depth	68.8±2.2	65.2±2.1	76.9±2.5	64.6±2.2	63.3±1.6	75.1±1.9
FTAlexNet-RGBD	73.6±5.1	69.7±4.5	88.6±4.5	70.1±4.5	67.6±4.2	86.8±3.9
Proposed Network	**89.3±6.6**	**84.8±7.4**	**95.2±2.4**	**84.8±7.4**	**82.0±6.6**	**93.9±2.2**

Table 6.10: Phase recognition results *after applying the HHMM* (mean±std) on the VerCT13 dataset. The best result for each evaluation metric is shown in bold.

across offline and online results: our proposed network yields the best performance for both offline and online recognitions.

6.3.4.3 VerCT13

In order to test the generalizability of the networks, we also perform evaluations using another dataset, i.e., the VerCT13 dataset, that has not been seen by the networks. Here, we solely focus on the performance of deep features since it has been shown in the previous section that the handcrafted features are outperformed by the finetuned deep features.

We show the phase recognition results on the VerCT13 dataset before and after applying the HHMM in Table 6.9 and 6.10, respectively. It can be seen that the results hold the same trend as the ones obtained from the VerCArm24 dataset. Before applying the HHMM, our proposed network yields the best performance, with an accuracy of 89.0%. It can also be seen that the recognition results after applying the HHMM obtained by our proposed network are the best, yielding accuracies of 95.2% and 93.9% for offline and online recognitions, respectively. The fact that the features extracted from our proposed network perform similarly on the VerCT13 and the VerCArm24 datasets demonstrates that the network does not overfit the finetuning dataset, i.e., it generalizes to other datasets.

6.3.5 Conclusions

Here, we have proposed a multi-stream CNN architecture which learns a shared feature representation among multi-modal data, i.e., color, depth, and their respective motion images. We train the network using a dataset, namely VerCArm24, containing multi-view RGBD videos recorded in the C-arm room. The experimental results show that the visual features extracted from such multi-modal data using the multi-stream networks outperform other visual features in performing phase recognition on multi-view OR-scene RGBD videos. We have also evaluated the generalizability of the network using another dataset, namely VerCT13, containing 13 vertebroplasty videos. The network has demonstrated satisfactory performance on both VerCArm24 and VerCT13 datasets, yielding accuracies of 96% and 95.2% for offline recognition, respectively.

6.4 End-to-End Architecture for Phase Recognition

In previous sections, we have shown that the visual features extracted from the deep neural networks are significantly more discriminative than the handcrafted visual features to perform surgical phase recognition on both laparoscopic and multi-view RGBD videos. In the phase recognition pipeline, the temporal constraints of the surgical workflow are enforced by hierarchical HMM (HHMM). However, the HHMM has several drawbacks. For instance, since the HHMM is based on the HMM, it carries the first-order Markov assumption where the current state only depends on the previous one. In addition, the amount of information passed along the sequence is limited to the number of states defined in the model, which is typically set to the number of classes. These drawbacks could lead to the loss of temporal information, especially long-term information, along the sequences.

These limitations are however not present in LSTM networks [Hochreiter 1997]. In theory, the LSTM network can retain a long-term information along the sequence. In addition, the amount of information passed along the network is defined by the number of memory cells, which can be set to any desired number. Furthermore, by appending the LSTM network to the deep networks, one can establish a pipeline which is solely based on neural network approaches. This end-to-end pipeline is different from the phase recognition pipelines explained in previous sections, which contain two independent steps: feature extraction and temporal constraint enforcement via HHM-based classification. Using the end-to-end pipeline, the phase recognition results can be directly obtained from the network since the temporal constraints are already enforced by the LSTM network.

In this section, we propose to use an LSTM network to enforce the temporal constraints of the surgical workflow and replace the HHMM in the recognition pipeline. In order to allow the LSTM network to capture the temporal constraints of the surgical workflow on such long videos, a large amount of training data is required. Because of this reason, we apply the LSTM network to perform surgical phase recognition on the largest dataset

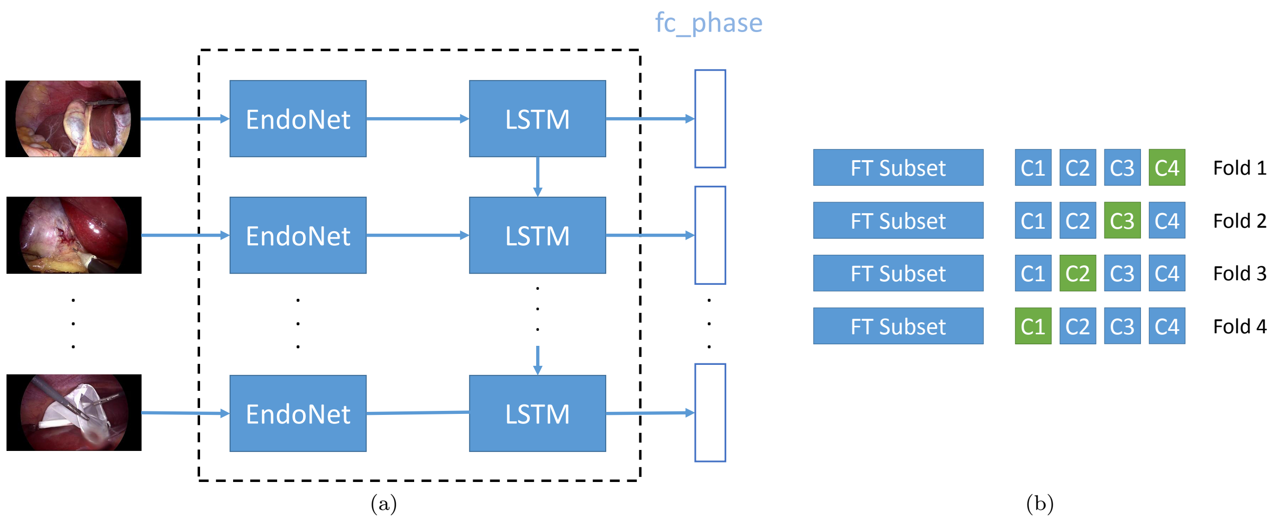


Figure 6.12: (a) LSTM configuration for surgical phase recognition on laparoscopic videos and (b) illustration of the training-testing configuration where in each fold, the subsets depicted in blue are used for training and in green for testing.

that we have, i.e., the Cholec80 dataset. Here, we construct the end-to-end pipeline by connecting EndoNet with an LSTM network.

6.4.1 Methodology

In Fig. 6.12-a, we show an illustration of the end-to-end architecture. To perform the surgical phase recognition task, we design the LSTM network to take the output of the second last layer of EndoNet as the image features, i.e., `fc8` (see Fig. 6.2). We pass these features to an LSTM network with 128 states. The output of the LSTM network is then passed to a fully-connected layer `fc_phase` with 7 nodes (equal to the number of phases in the Cholec80 dataset). The output of this fully-connected layer represents the confidences of the image belonging to the phases and are used for final predictions. In the case where bidirectional LSTM is used, both forward and backward LSTM networks have 128 states. The outputs of the forward and backward networks are concatenated before being passed to the layer `fc_phase`.

One of the interests of constructing an end-to-end network is the possibility to optimize the pipeline in one process; in contrast to previous pipelines which require separate training processes for the networks and the HHMMs. To train the end-to-end network in one process, it would create a huge memory requirement since EndoNet is a big network and the surgeries are of long durations (up to 2 hours per surgery). Due to this memory requirement, it is still difficult to train the pipeline in an end-to-end manner on complete recordings of surgeries. To overcome this problem, we train the end-to-end network differently. First, we use the trained EndoNet network from Section 6.2 to extract the image features from the dataset. Then, we train the LSTM network on complete surgery videos using these extracted image features as input. We use the same softmax loss function, defined in Eq. 6.1, for the optimization process. Once the LSTM network is trained, the EndoNet and the LSTM networks are then connected to construct the end-to-end pipeline which is used at test time.

6.4.2 Experimental Setup

6.4.2.1 Dataset

Here, we use the Cholec80 dataset to evaluate the approach. The training-testing configuration is modified in order to augment the amount of data used to train the LSTM network. This configuration is illustrated in Fig. 6.12-b, where the finetuning (FT subset) and evaluation subsets are depicted. The evaluation is still performed with a 4-fold cross validation, therefore the evaluation subset is divided into four folds (C1-4), each of which contains 10 videos. For each evaluation fold, in order to increase the amount of training data, we combine the videos from the FT subset and some videos from the evaluation subset to train the LSTM network. For instance, in the first fold where we test the fold C4, the LSTM network is trained on the combination of FT subset and the rest of evaluation subset (C1-3).

6.4.2.2 LSTM Training

The LSTM network is trained over complete sequences, thus the batch size is equal to 5985. This number corresponds to the maximum video duration found in the dataset, i.e., 5985 seconds since we are working at 1 fps. For sequences that are shorter than 5985 seconds, we pad them with zeros. As the LSTM network is not finetuned on a pre-trained network, we set the learning rate to 10^{-2}. The LSTM training process is carried out using the Caffe framework [Jia 2014] and it is performed for 30K iterations.

6.4.2.3 Baselines

Here, we compare the phase recognition results obtained using the HHMM to the results obtained using the LSTM network. For both pipelines, the image features are the output of the second last layer of EndoNet, i.e., `fc8`. For the sake of fairness, both the HHMM and the LSTM network are trained using the dataset configuration described in Section 6.4.2.1. The online recognition results obtained using the HHMM are compared to the results of the (*unidirectional*) LSTM network, while the offline results of HHMM are compared to the bidirectional LSTM network.

6.4.3 Experimental Results

6.4.3.1 Phase Recognition Results

In Table 6.11, we show the phase recognition results. Note that the HHMM results shown in this table are not the same as the ones shown in Table 6.3-a since they are computed using a different dataset configuration. However, we can see that there is not much difference between these HHMM results. This shows that the HHMM does not necessarily

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
EndoNet+HHMM	83.1±9.9	85.1±5.8	91.0±3.7	74.4±16.2	77.6±6.7	82.0±4.6
EndoNet+LSTM*	**88.1±6.5**	**89.0±5.7**	**92.2±7.7**	**84.4±7.9**	**84.7±7.9**	**88.6±9.6**

Table 6.11: Phase recognition result comparison on the Cholec80 dataset. Note that these are the results after the temporal constraints are enforced into the pipeline. *For EndoNet+LSTM, the online results are obtained using the vanilla LSTM network, while the offline results from the bidirectional LSTM.

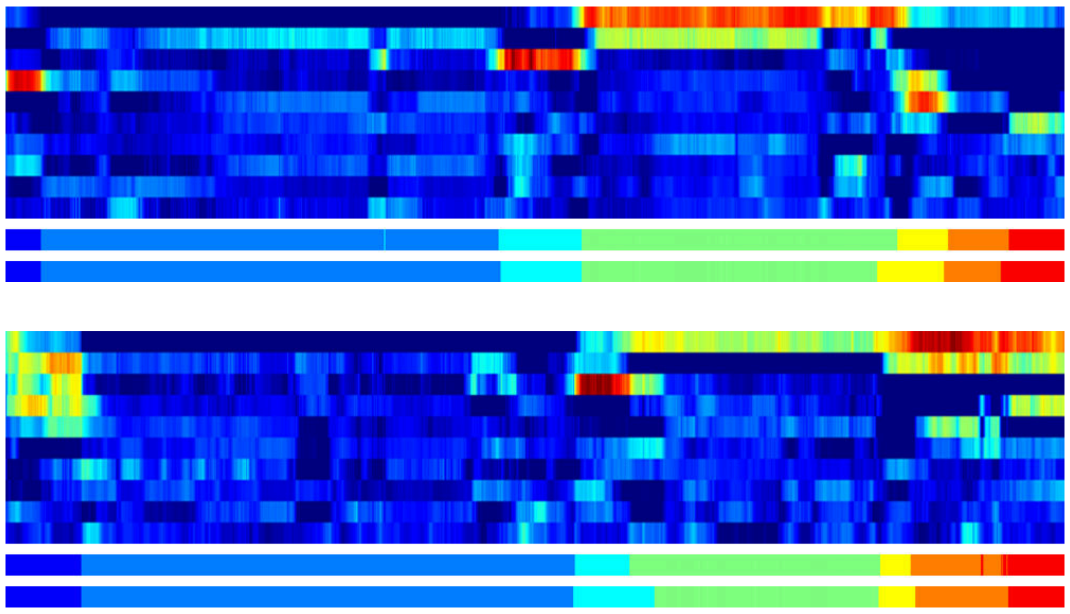
Feature	Metric	P1	P2	P3	P4	P5	P6	P7
HHMM - offline	Prec.	90.0±10.5	96.9±1.8	68.0±12.5	91.7±6.1	73.9±7.1	79.4±7.7	71.7±12.1
	Rec.	84.4±8.9	88.9±9.3	81.2±16.7	92.1±5.7	83.9±3.5	79.9±8.6	84.2±8.0
LSTM - offline	Prec.	90.2±6.5	95.0±1.5	88.9±3.0	93.9±6.2	87.0±3.8	75.5±9.9	85.9±5.0
	Rec.	88.2±7.0	95.9±2.7	87.9±3.7	95.8±2.8	90.0±3.3	79.7±5.9	85.4±4.3
HHMM - online	Prec.	94.1±5.2	95.8±1.3	63.9±12.5	81.7±7.2	57.7±12.0	64.2±7.8	64.8±9.1
	Rec.	75.6±5.1	81.1±9.3	64.8±17.3	85.8±6.2	78.1±2.0	78.1±8.5	81.0±6.7
LSTM - online	Prec.	93.7±5.6	93.1±7.1	80.4±4.8	91.1±7.1	79.9±7.3	76.6±15.4	75.9±8.8
	Rec.	86.8±6.9	95.7±2.1	80.6±5.6	91.1±9.3	79.6±8.5	71.8±3.9	87.1±4.5

Table 6.12: Precision and recall of phase recognition for each phase on Cholec80 using the EndoNet features: HHMM vs. LSTM.

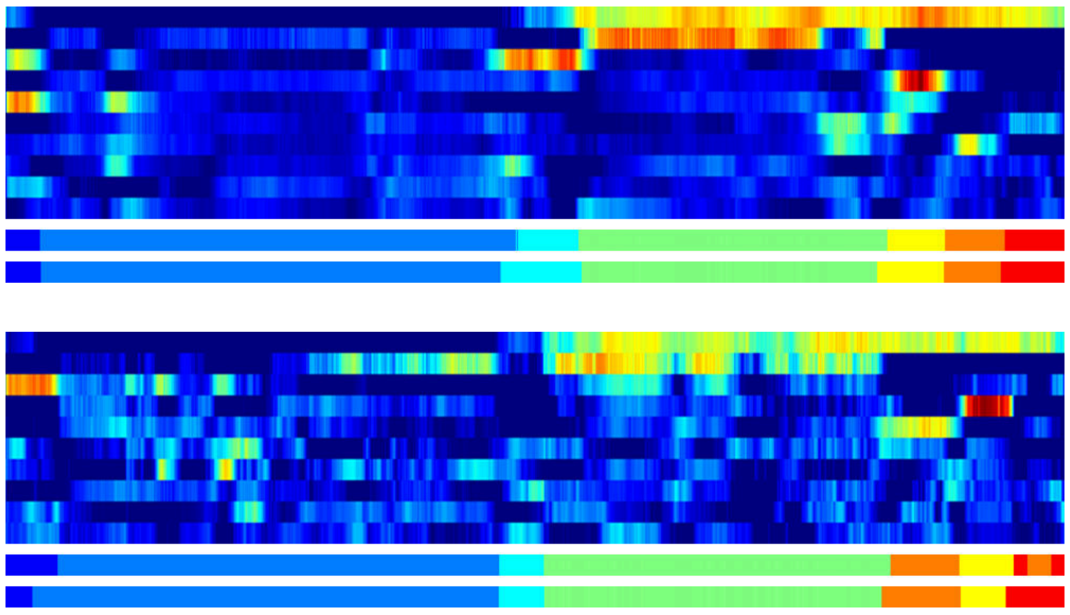
benefit from the increase of training data size. On the other hand, it is not the case for the end-to-end architecture. It can be observed that the LSTM network outperforms the HHMM. In particular, the online LSTM results are significantly better compared to the results of HHMM.

To demonstrate in detail the improvements introduced by the LSTM network, we show in Table 6.12 the precision and recall for each phase obtained by the HHMM and LSTM in both offline and online modes. One can observe that the LSTM network performs generally better than the HHMM over all phases. One of the particular interests in performing surgical phase recognition on cholecystectomy procedures is to construct a system to send a notification to senior surgeons when the *clipping and cutting* phase (i.e., P3) is occurring in the OR since it is a crucial phase of the procedure. It can be seen in Table 6.12 that P3 is recognized much better by the LSTM than the HHMM, showing the potential of using the end-to-end network to develop the notification system.

In order to better understand the end-to-end network, we visualize in Fig. 6.13 the results of the LSTM network in both offline and online modes. This visualization is obtained by taking the 10 principal components of the LSTM output (of size 128 in online and 256 in offline) from several sequences. We also show the estimated phases as well as the phase ground truth annotations of the sequences. Qualitatively, we can clearly see that there is a strong correlation between the LSTM output and the phases. For example, the fourth component of the first example (online) has high values during the phase P1,



(a) LSTM results (online)



(b) Bi-LSTM results (offline)

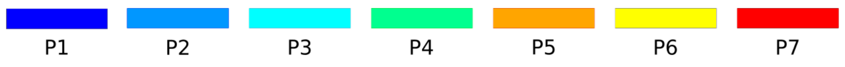


Figure 6.13: Visualization of LSTM and Bi-LSTM results. The horizontal axis represents the time progression in a surgery. Since the surgeries are not of the same duration, the temporal axes are scaled for visualization purposes. Each visualization consists of (in top-bottom order): (1) 10 principal components of the LSTM output, (2) the estimated phases and (3) the ground truth annotation.

Tolerance (s)	Phase						
	P1	P2	P3	P4	P5	P6	P7
<30	40	36	35	31	39	25	30
30-59	0	2	2	2	1	3	1
60-89	0	1	1	2	0	7	7
90-119	0	1	2	5	0	0	2
≥120	0	0	0	0	0	0	0
TOTAL	40	40	40	40	40	35	40

Table 6.13: Number of phases that are correctly identified using the Bi-LSTM network within the defined tolerance values in the 40 evaluation videos of Cholec80. The number of P6 occurrences is not 40 since not all surgeries go through the cleaning and coagulation phase.

but not in other phases; while the third component is more dominant when the phase P3 is being executed. This shows that the LSTM can be used to model the temporal constraints of the surgical workflow. Furthermore, it is interesting to see that the LSTM network which runs in online mode can yield smooth predictions which rarely contain jumps in between phases, in contrast to the HHMM results shown in Fig. 6.7-b. This demonstrates how reliable the LSTM can be in performing the online phase recognition task.

The offline recognition results obtained using the Bi-LSTM network (offline) also contain a trend similar to the ones of unidirectional LSTM (online). However, it is interesting to see in the last example shown in Fig. 6.13-b that the estimated phases jump between P7 and P6 at the end of the sequence, even though there is no transition from P7 to P6 in the dataset (see Fig. 4.8). It is important to recall that the HMM-based pipeline explicitly models the phase transitions using a state transition matrix computed using the training data, while it is not the case for the LSTM network. In other words, even in offline mode, the Bi-LSTM network is more likely to allow phase transitions which are not observed in training data. Even though this is not desirable on the Cholec80 dataset where the phase transitions are already determined, the fact that the LSTM network allows such unobserved phase transitions does not necessarily lead to poor results in other cases. It might actually be useful to relax the phase transition constraints so that the approach could be applied to other videos which might have a workflow that is different from the one defined in the training data.

6.4.3.2 Clinical Application: Surgical Video Database Indexing

In Section 6.2.5.5, we have shown how the HHMM pipeline would perform in automatic video database indexing. Here, we compute the same metrics for the phase recognition results obtained by the end-to-end architecture (i.e., the Bi-LSTM network since this corresponds to offline recognition) and present the number of phase boundaries that are detected within defined temporal tolerance values in Table 6.13. It can be observed that by comparing the results to the ones presented in Table 6.4, some phases do not necessarily enjoy improvements from the Bi-LSTM network. For instance, 30 instances of P6 are detected under 30 seconds using the HHMM, while 25 instances are detected using the LSTM network. However, we can still see significant improvements in the phase detection results since all the phases are correctly detected within a 2-minute threshold, which is not the case when the HHMM pipeline is used. It is important to recall that this error is computed with respect to the strict phase boundaries defined in the phase annotations. In practice, these boundaries are not as harsh or visually obvious. Therefore, the fact that all the phases are correctly detected within a 2-minute boundary is satisfactory, resulting in a faster and easier surgical video indexing process.

6.4.4 Conclusions

We have presented an end-to-end architecture to address surgical phase recognition on laparosopic recordings of cholecystectomy procedures. Even though the LSTM network is not trained in an end-to-end manner, it has been shown to outperform the HHMM in carrying out the task. This indicates that the LSTM network can be used to enforce the temporal constraints of the surgical workflow. We have also shown how several real-world applications, such as automatic video indexing and real-time notification system, could benefit from the improvements introduced by the LSTM network.

Here, we have presented an end-to-end network by connecting EndoNet with an LSTM network. Despite the fact that the end-to-end connection between EndoNet and the LSTM network is established at test time, this is not the case at training time. Due to memory constraints, it is still difficult to train both EndoNet and LSTM on complete sequences in an end-to-end manner. For future work, it would be interesting to investigate deep network training strategies which allow the end-to-end optimization over long sequences.

6.5 Summary

In this chapter, we have shown how deep neural network approaches can significantly improve the results of surgical activity recognition tasks. We have presented two deep architectures which perform surgical phase recognition on laparoscopic and RGBD videos. We have shown that by using the features learnt by the network, the results are significantly better than the ones obtained using handcrafted visual features. In addition, we have also presented an end-to-end architecture which consists of EndoNet and an LSTM network. Having evaluated the end-to-end architecture on the Cholec80 dataset, we demonstrated that further improvements can be achieved when the LSTM network is used to enforce the temporal constraints of the surgical workflow instead of the HHMM.

The high performance of the deep learning approaches suggest that they have a high potential to be used in real world applications. We have demonstrated a high accuracy for surgical phase recognition on RGBD data using the multi-stream network. We have also shown how the end-to-end network could be used to build a notification system as well as to facilitate the video indexing process on laparoscopic videos. Considering such satisfactory results, future work should address the implementation of such approaches to develop a context aware system (CAS) in the OR.

7 Conclusions and Future Work

Ends are not bad things, they just mean that something else is about to begin.
– C. JoyBell C.

Chapter Summary

In this chapter, we conclude the thesis by presenting the summary of this work. We also discuss the possible directions of future work in order to address the current limitations and improve the performance of the methods.

7.1 Conclusions

The objective of this thesis is to develop methods to recognize the surgical activities occurring in the operating rooms (ORs). Here, we focus on using visual information from the cameras, specifically laparoscopic and multi-view OR-scene RGBD videos because of their complementary nature: the former captures in detail the anatomical structure inside the abdominal cavity of the patient, while the latter captures the general activities occurring in the OR. Despite the vast literature on activity recognition in the computer vision community, *surgical* activity recognition is still at its early stage. In the course of this thesis, we have generated large datasets containing laparoscopic and multi-view OR-scene RGBD recordings of real surgeries. These datasets have not only permitted us to establish strong baselines using state-of-the-art methods, but also allowed us to construct robust approaches to carry out the tasks. They have also allowed us to perform extensive experiments and evaluate the proposed approaches on *real* data.

In Chapter 4, we have presented two recognition pipelines to address two surgical activity recognition tasks on laparoscopic videos. For the first task, we introduced surgery classification, which is a pre-segmented classification task involving the identification of the surgery type from a video. As for the second task, it is surgical phase recognition, which is a frame-wise classification task where the goal is to recognize surgical phases in cholecystectomy procedures. The recognition pipelines consist of the following steps: (1) extraction of visual features, (2) feature encoding using bag-of-word (BOW) approach, and (3) classification using SVM-based and HMM-based models. We have shown that the handcrafted visual features carry discriminative characteristics for both tasks and yield promising recognition results.

In Chapter 5, we presented approaches to address two surgical activity recognition tasks on multi-view OR-scene RGBD videos. The first task, surgical action classification, is a pre-segmented classification task which consists of identifying the surgical actions performed in video clips. As for the second task, it is frame-wise surgical phase recognition on vertebroplasty procedures. Here, we proposed a novel encoding approach that is based on the BOW approach. This approach retains the spatio-temporal information from the visual features during the feature encoding process. The experimental results show that the intensity and depth features contain complementary characteristics. The best results are obtained when the features are combined to perform the tasks. We have also demonstrated that our proposed encoding approach outperforms the ubiquitous spatial pyramid matching (SPM) [Lazebnik 2006] and improves the performance of the handcrafted visual features in carrying out both tasks on the multi-view RGBD videos.

In Chapter 6, we proposed to use deep learning approaches to automatically learn the discriminative visual features from the data. This eliminates the need for the manual engineering of robust features, which is very challenging due to the inherent visual challenges in both laparoscopic and multi-view RGBD videos. We have presented several deep architectures to perform surgical phase recognition on both video types. For the laparoscopic videos, we proposed EndoNet which is a network designed to perform jointly surgical phase recognition and tool presence detection. For the multi-view RGBD videos, we presented a multi-stream network architecture which learns a shared feature representation among the multiple modalities (i.e., color, depth, and the respective motion images). We have demonstrated that the visual features extracted from the proposed networks outperform other features in carrying out surgical phase recognition on laparoscopic and multi-view RGBD videos, respectively. Furthermore, we have also presented an end-to-end deep architecture with LSTM network to address surgical phase recognition on cholecystectomy procedures. We have shown that the LSTM network outperforms the HMM-based pipeline and it can be used to enforce the temporal constraints of the surgical workflow into the recognition pipeline.

In this thesis, we have shown that our proposed pipeline based on the deep learning algorithms yields promising results. Both EndoNet and multi-stream networks significantly outperform other methods based on handcrafted visual features. In addition, we have also shown that EndoNet could be used to construct a notification system for tool usage detection during cholecystectomy procedures. Moreover, we have also demonstrated how the combination of EndoNet and LSTM could be used to recognize the clipping and cutting phase of cholecystectomy procedure and to facilitate the annotation process of a laparoscopic video database. These encouraging results show the potentials of implementing an activity recognition system based on our proposed approach in order to address real clinical applications.

7.2 Discussion and Future Work

Combination of sources of visual information. In this thesis, we perform surgical activity recognition tasks using the visual information coming from both laparoscopic and OR-scene RGBD videos. Ideally, both video types should be used simultaneously to obtain the complete information regarding the activities occurring in the OR. However, since the laparoscopic and OR-scene videos in our datasets were recorded in different rooms, we perform the task separately on each video type. Future work should address this limitation, so that the recognition pipeline would receive more comprehensive information regarding the activities occurring in the OR.

Recognition of finer activities. Here, we have presented several methods to recognize surgical phases during cholecystectomy and vertebroplasty procedures. These phases typically consist of finer actions, such as interactions between tool and tissue in laparoscopic videos, and interactions between surgeon and surgical staff (or patient) in OR-scene videos. It would be interesting to detect these fine activities to get a better understanding regarding the activities occurring in the OR. In particular, it would be interesting to perform such task on laparoscopic videos in order to bridge the gap between high-level modeling formalization [Katić 2014] and the signals from the OR. Future work should address and overcome the challenges by proposing robust methods to extract high-level features, such as tool and tissue segmentation and human pose estimation.

Multi-task learning. In Section 6.2, we have shown that the deep architecture (i.e., EndoNet) that is designed to perform surgical phase recognition jointly with tool presence detection outperforms another architecture optimized solely for the phase recognition task. We hypothesize that the performance of the network could be further improved by jointly addressing other related tasks, such as tool detection and anatomy identification. For the multi-stream RGBD network, we could also improve the performance by designing the network to perform jointly phase recognition and clinician pose estimation.

Weakly-supervised learning for tool detection. The Cholec80 dataset contains the annotation of tool presence, which has been used to train the multi-task EndoNet model. We hypothesize that these tool *presence* annotations could also be used to address tool detection by using weakly-supervised learning approaches where object localization is solved using the annotations of object presence in the images [Hwang 2016]. It is particularly interesting since in most tool detection approaches, the methods typically require a model of the tools, which are obtained either from a 3D model of the tool or a model generated from the data (via bounding box annotations). By using the weakly-supervised approach, the tool model could be obtained by only using the annotations of the tool presence, which are easier to obtain than bounding box annotations. Combined with the results of anatomy detection, the localization of the tool could be used as to analyze the interaction between the tools and the anatomical structures.

Multi-view deep architecture. In Section 6.3, we have presented a multi-stream network which extracts a feature representation shared among multiple modalities, i.e., color, depth and their respective motion images. The deep network however does not consider the multi-view aspect of the dataset. Thus, there is no semantic connection established between views in the feature representation. Future work should address this limitation by exploiting this characteristic of the data in order to generate more discriminative features.

Training data. Currently, state-of-the-art methods are typically based on deep learning algorithms, which require a large training set. Our experimental results in Section 6.2 have also shown that the performance of the network highly depends on the size of training data. Intuitively, with more training data, better results could be obtained for the surgical activity recognition tasks.

Active learning. In the course of this thesis, we have generated large datasets which took a long time to annotate. This process is necessary for fully supervised learning approaches such as the ones proposed in this thesis. In order to reduce the workload of the annotation process in the future and to exploit also the large volume of unannotated videos, it would be interesting to explore active learning techniques. Such techniques do not require the complete dataset to be annotated and demand fewer expert annotations during the training process to generate the model.

End-to-end network training. We have presented an end-to-end network by connecting EndoNet with an LSTM network. Despite the fact that the end-to-end connection between EndoNet and the LSTM network is established at test time, this is not the case at training time. Due to memory constraints, it is still difficult to train both EndoNet and LSTM on complete sequences in an end-to-end manner. Training them in such a manner is important due to the fact that when trained separately, EndoNet is only trained to classify images. Therefore, the visual features extracted by EndoNet do not take into account the temporal information from the surgical workflow. For future work, it would be interesting to investigate deep network training strategies which allow the end-to-end optimization on long sequences so that the temporal information could be incorporated into the visual features.

List of Publications

[Twinanda 2014a] Andru Putra Twinanda, Michel De Mathelin, and Nicolas Padoy. Fisher kernel based task boundary retrieval in laparoscopic database with single video query. In: MICCAI, 2014. Supplementary video (cited on page 11)

[Twinanda 2014b] Andru Putra Twinanda, Jacques Marescaux, Michel De Mathelin, and Nicolas Padoy. *Towards Better Laparoscopic Video Database Organization by Automatic Surgery Classification.* In IPCAI, 2014. Supplementary video (cited on pages 12 and 165)

[Twinanda 2015a] Andru Putra Twinanda, Emre O. Alkan, Afshin Gangi, Michel de Mathelin, and Nicolas Padoy. *Data-driven spatio-temporal RGBD feature encoding for action recognition in operating rooms.* Int. J. Computer Assisted Radiology and Surgery, vol. 10, no. 6, pages 737–747, 2015. (cited on pages 12 and 167)

[Twinanda 2015b] Andru Putra Twinanda, Jacques Marescaux, Michel de Mathelin and Nicolas Padoy. *Classification approach for automatic laparoscopic video database organization.* Int. J. Computer Assisted Radiology and Surgery, vol. 10, no. 9, pages 1449–1460, 2015. (cited on pages 12 and 165)

[Twinanda 2016a] Andru Putra Twinanda, Pramita Winata, Afshin Gangi, Michel De Mathelin, and Nicolas Padoy. Multi-Stream Deep Architecture for Surgical Phase Recognition on Multi-View RGBD Videos. In: M2CAI Workshop at MICCAI, 2016. (cited on page 12)

[Twinanda 2016b] Andru Putra Twinanda, Didier Mutter, Jacques Marescaux, Michel de Mathelin, and Nicolas Padoy. Single- and Multi-Task Architectures for Surgical Workflow Challenge at M2CAI 2016. CoRR, vol. abs/1610.08844, 2016.

[Twinanda 2016c] Andru Putra Twinanda, Didier Mutter, Jacques Marescaux, Michel de Mathelin, and Nicolas Padoy. Single- and Multi-Task Architectures for Tool Presence Detection Challenge at M2CAI 2016. CoRR, vol. abs/1610.08851, 2016.

[Twinanda 2017] Andru Putra Twinanda, Sherif Shehata, Didier Mutter, Jacques Marescaux, Michel de Mathelin, and Nicolas Padoy. *EndoNet: A Deep Architecture for Recognition Tasks on Laparoscopic Videos.* IEEE Transactions on Medical Imaging, vol. 36, no. 1, pages 86–97, Jan 2017. Supplementary video (cited on pages 12, 166, and 168)

A List of Datasets

In the course of this work, we have generated seven datasets: four of them contain laparoscopic videos and the rest contains the multi-view OR-scene RGBD videos. Two of the laparoscopic datasets are already published. Here, we present the details of each dataset.

A.1 Laparo208 Dataset

This dataset contains 208 laparoscopic videos of 8 different types of surgeries, recorded at the University Hospital of Strasbourg/IRCAD. The surgical procedures procedures were performed by 10 different surgeons. All videos are recorded at 25 fps. In total, they sum up to more than 300 hours or approximately $25M$ frames. This dataset has been used to perform the evaluation of surgery classification on laparoscopic videos (Section 4.2). We show the statistics and the frame samples from the dataset in Table A.1 and Fig. A.1, respectively.

	Surgery	#Vid	Avg. length (min.)
1	Sigmoidectomy	26	113 ± 41
2	Eventration	28	73 ± 52
3	Bypass	36	119 ± 29
4	Hernia	34	56 ± 28
5	Cholecystectomy	36	65 ± 36
6	Nissen Gerd	22	94 ± 33
7	Adrenalectomy	11	108 ± 346
8	Sleeve Gastrectomy	15	91 ± 31

Table A.1: List of the abdominal surgery classes in our dataset along with their corresponding number of videos and average video length (± std) in minutes.

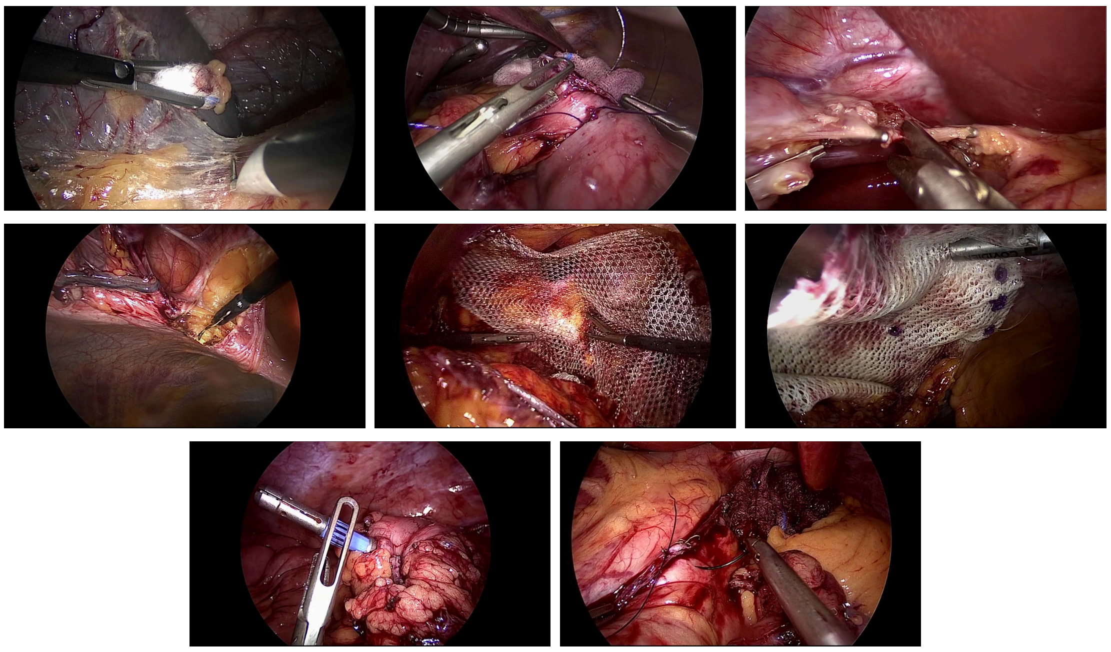


Figure A.1: Frame samples from the Laparo208 dataset.

A.2 Cholec80 Dataset

This dataset contains 80 videos of cholecystectomy procedures performed by 13 different surgeons at the University Hospital of Strasbourg/IRCAD. The dataset is annotated with the surgical phases and the tool presence. The surgical phases are defined by our senior surgeon. For the tool annotation, A surgical tool is defined as present if at least 50% of the tip is visible in the video frame. This dataset has been used for the evaluation in Section 4.3 and 6.2.

The statistics and the list of the surgical tools is shown in Fig. A.2 and A.3. The list of the phases along with the statistics can be found in Table A.2. In Fig. A.4, we show the frame samples from each phase and illustrate the transitions between phases generated from our dataset.

This dataset was publicly released in January 2017[1].

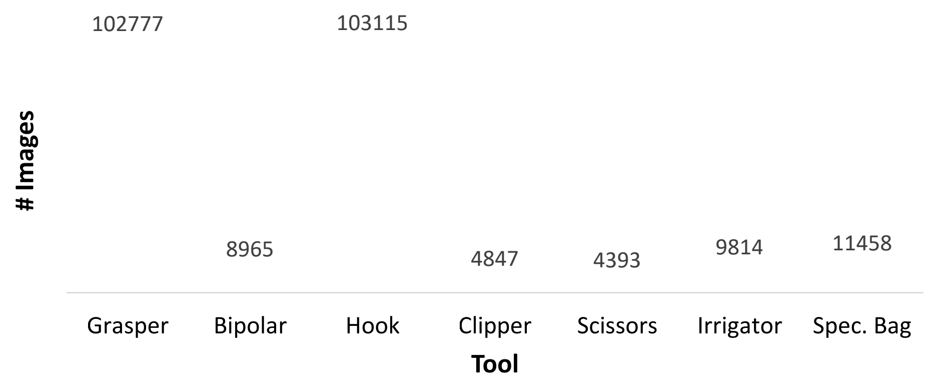


Figure A.2: Distribution of tool presence annotations in the Cholec80 dataset for tool presence.

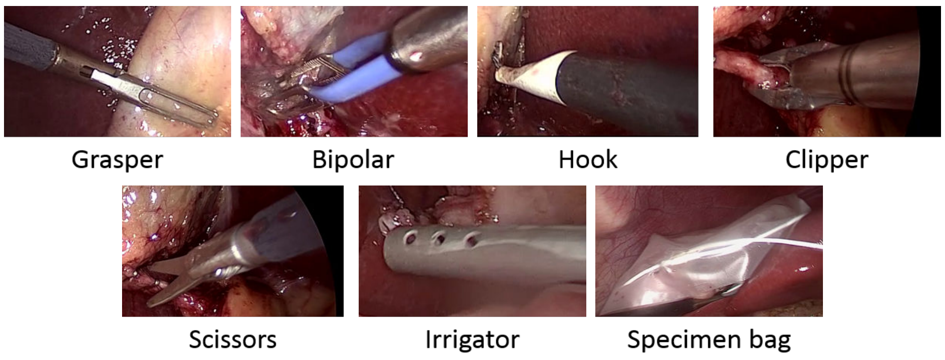


Figure A.3: List of the seven surgical tools used in the Cholec80 dataset.

[1]http://camma.u-strasbg.fr/datasets

ID	Phase	Avg. duration (min)
P1	Preparation	1.8±1.7
P2	Calot triangle dissection	15.6±11.1
P3	Clipping and cutting	2.9±2.1
P4	Gallbladder dissection	12.2±8.9
P5	Gallbladder packaging	1.6±0.8
P6	Cleaning and coagulation	3.0±2.6
P7	Gallbladder retraction	1.4±1.2
Complete surgery		38.4±17.1

Table A.2: List of phases in the Cholec80 dataset, including the mean ± std of the duration of each phase in seconds.

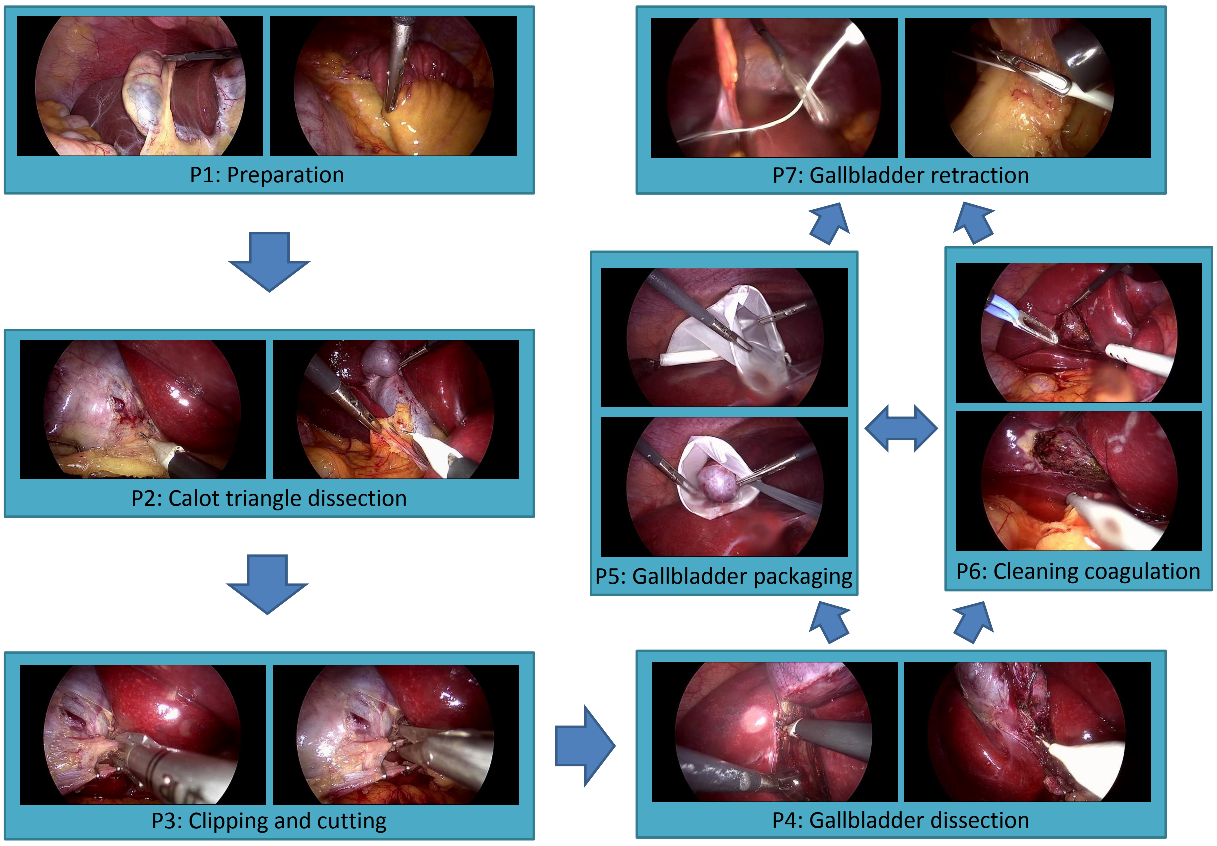

P1: Preparation
P2: Calot triangle dissection
P3: Clipping and cutting
P4: Gallbladder dissection
P5: Gallbladder packaging
P6: Cleaning coagulation
P7: Gallbladder retraction

Figure A.4: Surgical phase transitions in the Cholec80 dataset.

A.3 m2cai16-workflow Dataset

This dataset contains 41 cholecystectomy videos, recorded at the University Hospital of Strasbourg/IRCAD and the Hospital Klinikum Rechts der Isar in Munich [Stauder 2016]. Some of the videos in this dataset are taken from the Cholec80 dataset. This dataset is annotated with 8 surgical phases defined through a collaboration between the two institutions. This dataset has been used for the M2CAI workflow challenge[2] and was publicly released in October 2016.

The statistics and the frame samples from the dataset can be found in Table A.3 and Fig. A.6, respectively.

	Phase	Avg. duration (min)
1	TrocarPlacement	2.8±1.5
2	Preparation	1.5±1.8
3	CalotTriangleDissection	9.8±7.7
4	ClippingCutting	4.3±3.0
5	GallbladderDissection	9.3±7.1
6	GallbladderPackaging	1.6±2.0
7	CleaningCoagulation	5.0±3.8
8	GallbladderRetraction	4.2±4.8
Complete surgery		38.4±13.8

Table A.3: List of phases in the m2cai16-workflow dataset, including the mean ± std of the duration of each phase in seconds.

A.4 m2cai16-tool Dataset

This dataset includes 15 cholecystectomy videos, recorded at the University Hospital of Strasbourg/IRCAD. These videos are taken from the Cholec80 dataset. It has been used for the M2CAI tool presence detection challenge[2] and was publicly released in October 2016. The dataset contains the tool presence annotations of seven surgical tools (as shown in Fig. A.3). The statistics of this dataset is presented in Fig. A.5.

[2]http://camma.u-strasbg.fr/m2cai2016/

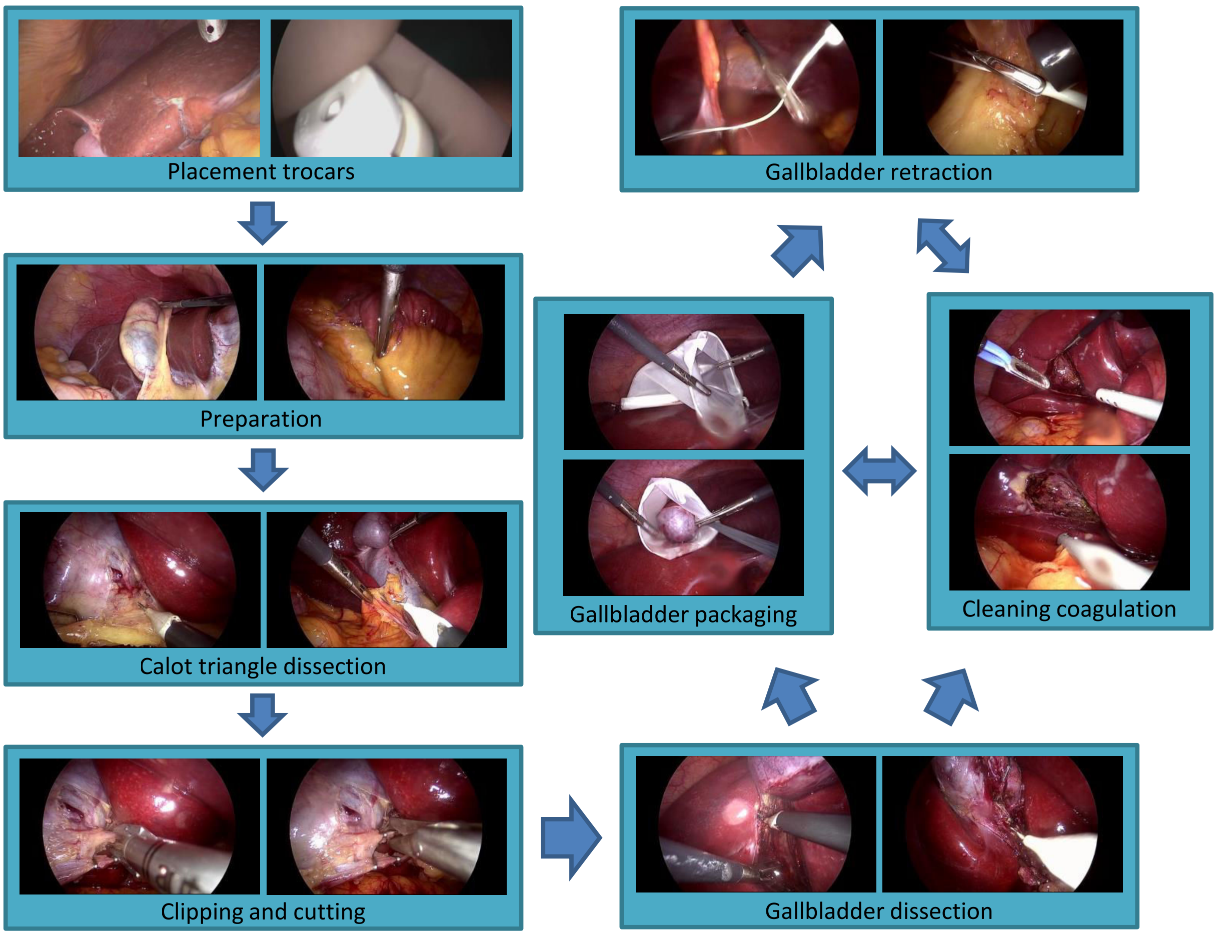


Figure A.6: Surgical phase transitions in the m2cai16-workflow dataset.

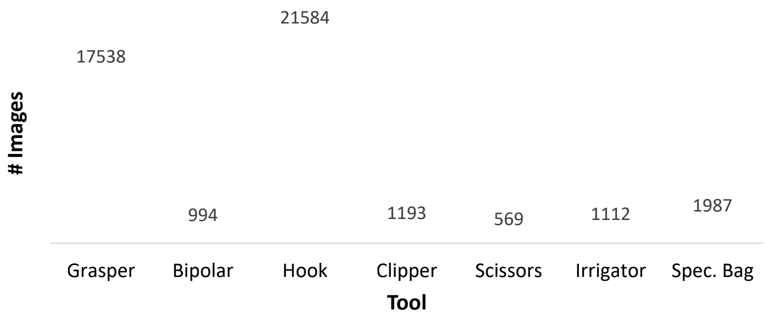


Figure A.5: Distribution of tool presence annotations in the m2cai16-tool dataset.

A.5 SurgActionMultiView3D Dataset

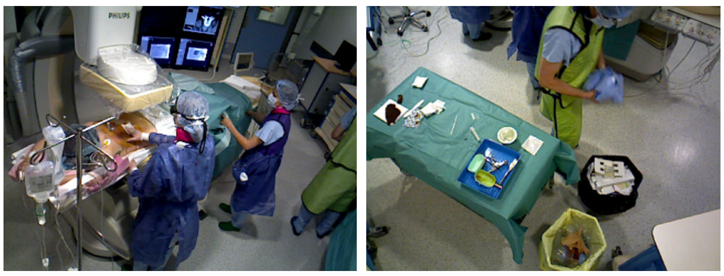
This dataset contains 1734 action-labeled multi-view RGBD video clips of 15 different actions. These actions consist of general actions (e.g. bed entering, moving patient to the OR bed) and vertebroplasty-specific actions (e.g. hammering, cleaning patient's back). The details and statistics of the actions can be found in Table A.4. In Fig. A.7, we show frame samples from several actions from the dataset.

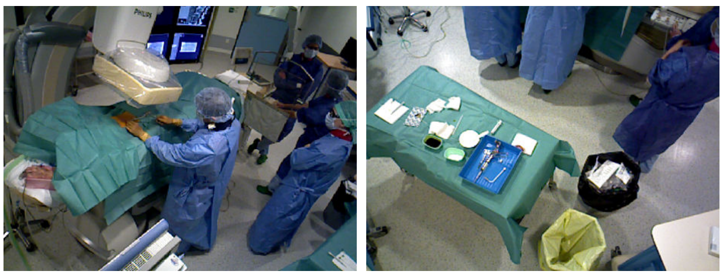
Action	#Inst.	Avg. duration (s)
Adjusting C-arm	507	14.5 ± 24.3
Adjusting OR bed	144	8.8 ± 4.9
Bed entering	59	15.8 ± 7.0
Bed leaving	61	19.9 ± 9.8
Cement injection	29	374.4 ± 105.1
Cleaning patient's back	245	20.0 ± 10.4
Hammering	216	15.8 ± 7.2
Monitor adjustment	136	12.6 ± 7.7
Moving patient from OR bed	31	15.0 ± 5.1
Moving patient to OR bed	30	20.2 ± 8.4
Putting sterile drapes	126	10.5 ± 4.4
Removing sterile drapes	42	17.4 ± 9.0
Mixing cement	28	75.9 ± 21.5
Opening package	51	12.7 ± 5.4
Throwing package	29	10.2 ± 4.0

Table A.4: The list of actions to recognize in the SurgActionMultiView3D dataset, including the number of instances per action and the length of the video clips (mean $\pm$ std) in second.

(a) Bed entering

(b) Mixing cement


(c) Cleaning patient's back


(d) Hammering

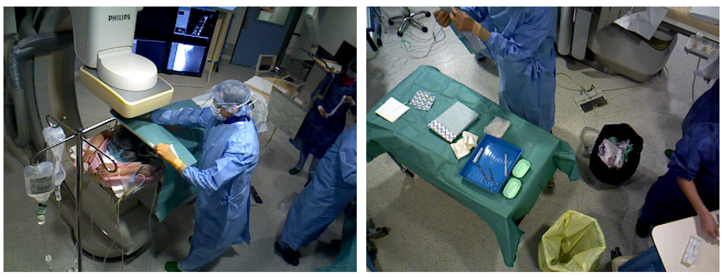

(e) Putting sterile drape

Figure A.7: Frame samples from several actions in the SurgActionMultiView3D dataset: (a) bed entering, (b) mixing cement, (c) cleaning patient's back, (d) hammering, and (e) putting sterile drape.

A.6 VerCArm24 Dataset

This dataset contains 24 multi-view RGBD videos of vertebroplasty procedures, recorded at the C-arm room of Interventional Radiology Department of University Hospital of Strasbourg. Several videos from the SurgActionMultiView3D dataset are included in this dataset. This dataset is annotated with 8 surgical phases defined by our senior clinician. A part of this dataset has been used to evaluate the phase recognition methods in Section 5.3. The complete dataset was utilized in Section 6.3. We show the phase statistics and the phase transitions in Table A.5 and Fig. A.8, respectively.

ID	Phase	Avg. duration (min.)
Q1	Patient preparation	16.3±10.0
Q2	Sterilizing	11.4±3.3
Q3	Needle insertion	18.6±13.6
Q4	Cement mixing	4.2±1.1
Q5	Cement injection	8.5±2.8
Q6	Wound dressing	3.6±1.3
Q7	Image acquisition	2.1±1.0
Q8	Patient leaving	3.0±0.9
Complete surgery		68.8±30.9

Table A.5: Phase list and statistics in the VerCArm24 dataset.

A.7 VerCT13 Dataset

This dataset contains 13 multi-view RGBD videos of vertebroplasty procedures, recorded at the CT room of Interventional Radiology Department of University Hospital of Strasbourg. This dataset is annotated with 8 surgical phases defined by our senior clinician. It has been used to evaluate the method presented in Section 6.3. We show the phase statistics and the defined phases in Table A.6 and Fig. A.9, respectively.

ID	Phase	Avg. duration (min.)
Q1	Patient preparation	25.9±15.2
Q2	Sterilizing	8.8±1.7
Q3	Needle insertion	19.8±15.4
Q4	Cement mixing	2.5±1.0
Q5	Cement injection	13.0±9.9
Q6	Wound dressing	3.3±1.3
Q7	Image acquisition	2.0±1.2
Q8	Patient leaving	4.0±3.2
Complete surgery		81.1±48.4

Table A.6: Phase list and statistics in the VerCT13 dataset.

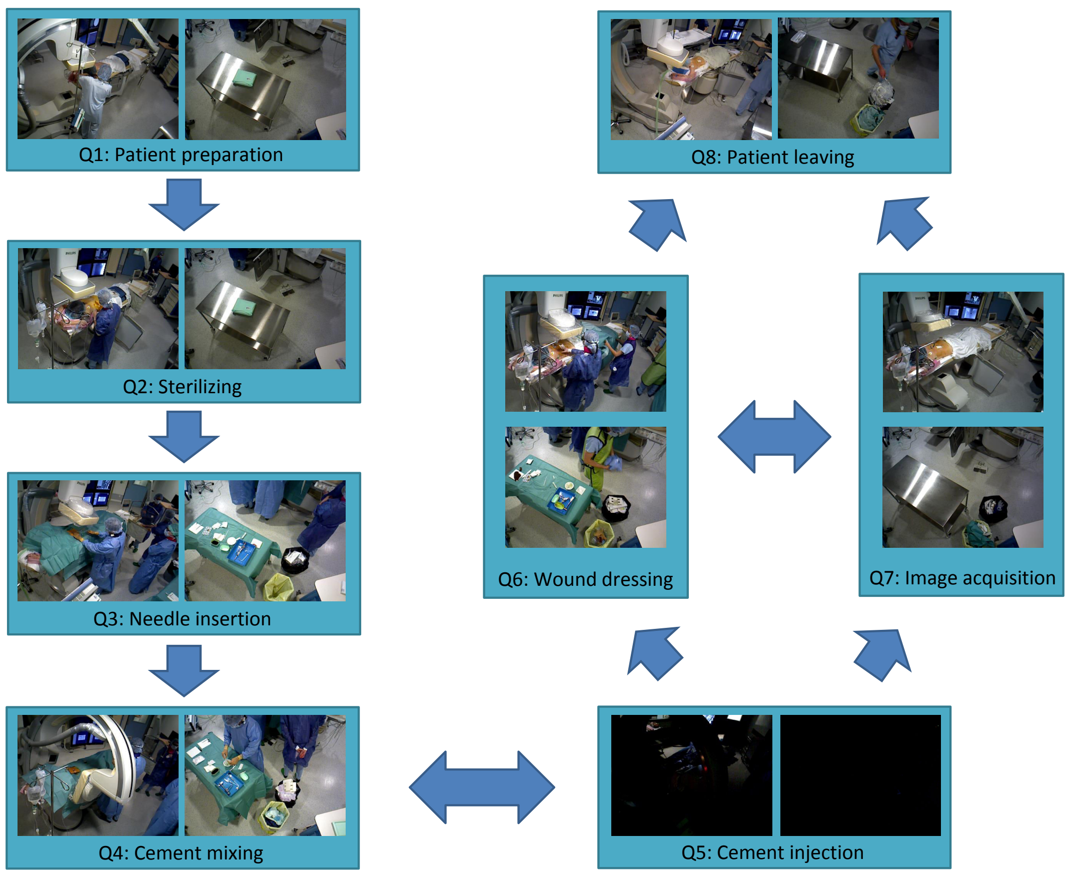


Figure A.8: Surgical phase transitions in the VerCArm24 dataset.

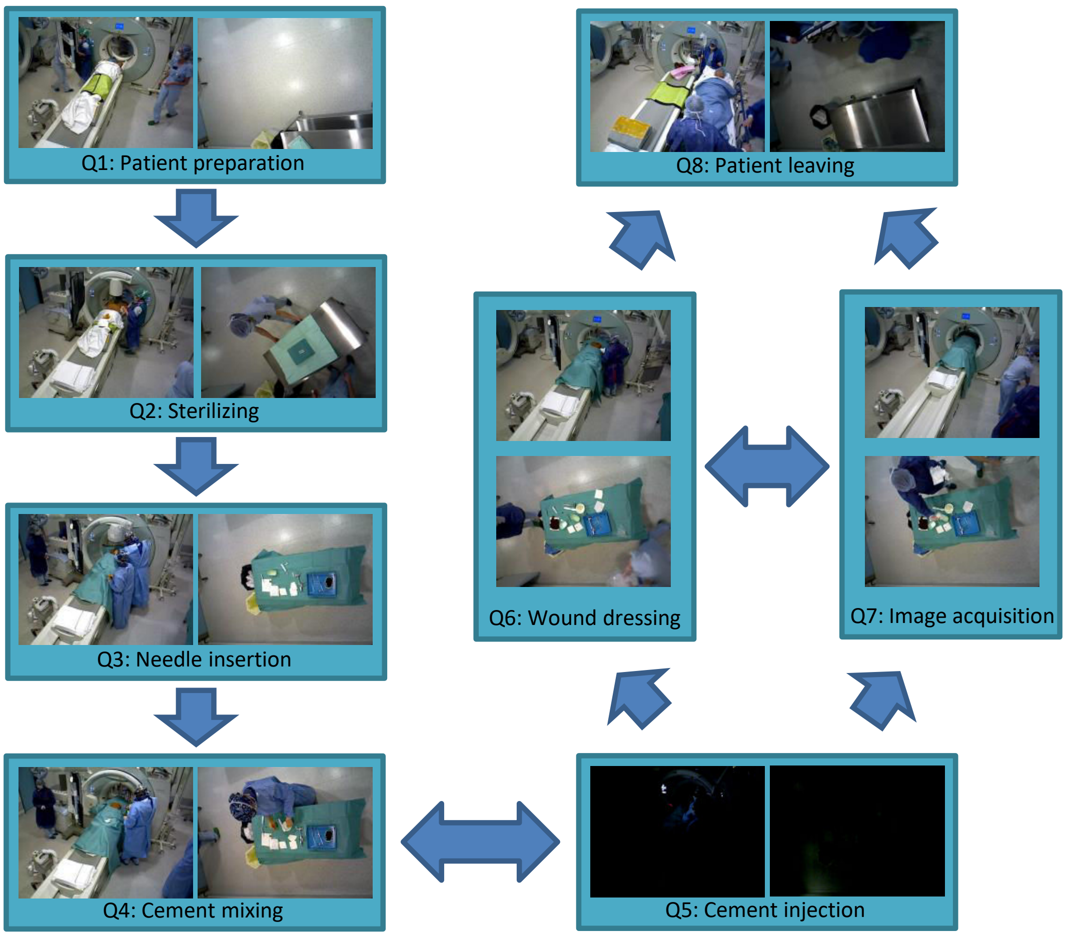


Figure A.9: Surgical phase transitions in the VerCT13 dataset.

B Publication Not Discussed in This Thesis

Here, we show a paper, published at MICCAI 2014, which presents an unsupervised approach to retrieve from the laparoscopic video database the occurrence of a task depicted in a query video.

Fisher Kernel Based Task Boundary Retrieval in Laparoscopic Database with Single Video Query

Andru Putra Twinanda, Michel De Mathelin, and Nicolas Padoy

ICube, University of Strasbourg, CNRS, France
{twinanda,demathelin,npadoy}@unistra.fr

Abstract. As minimally invasive surgery becomes increasingly popular, the volume of recorded laparoscopic videos will increase rapidly. Invaluable information for teaching, assistance during difficult cases, and quality evaluation can be accessed from these videos through a video search engine. Typically, video search engines give a list of the most relevant videos pertaining to a keyword. However, instead of a whole video, one is often only interested in a fraction of the video (e.g. intestine stitching in bypass surgeries). In addition, video search requires semantic tags, yet the large amount of data typically generated hinders the feasibility of manual annotation. To tackle these problems, we propose a coarse-to-fine video indexing approach that looks for the time boundaries of a task in a laparoscopic video based on a video snippet query. We combine our search approach with the Fisher kernel (FK) encoding and show that similarity measures on this encoding are better suited for this problem than traditional similarities, such as dynamic time warping (DTW). Despite visual challenges, such as the presence of smoke, motion blur, and lens impurity, our approach performs very well in finding 3 tasks in 49 bypass videos, 1 task in 23 hernia videos, and also 1 cross-surgery task between 49 bypass and 7 sleeve gastrectomy videos.

Keywords: surgical workflow analysis, laparoscopy, time boundaries, video indexing, sliding window, Fisher kernel.

1 Introduction

Most knowledge, in the form of texts, images, and even videos, is just one keyword and a click away thanks to search engines. Because images and videos have a rich content that still cannot be fully extracted automatically by computers, their retrieval is mostly possible because of the semantic tags provided by manual annotation. However, with vast amounts of data, browsing the videos and associating them with semantic tags manually becomes tedious. This is especially true for medical data, where the skilled annotators are moreover likely to be clinicians with little time on their hands.

In this paper, we therefore propose a method that looks for the time boundaries of a task in a video based on a video query. We use a video snippet of the task as query in order to eliminate the need for semantic tags. By providing the snippet, the method automatically provides the annotation of the task in relevant videos in the database. We call this problem video indexing and focus in this work on laparoscopic videos. Such videos contain invaluable information about the execution of surgeries in various clinical configurations. By implementing this method, fellow surgeons can observe different techniques in performing a specific surgical task. Additionally, such a method can provide boundary candidates in order to automate or simplify semantic tag annotation. Processing laparoscopic videos is, however, not a trivial problem due to atypical visual challenges, such as the presence of smoke, specular reflection, motion blur, and lens impurity.

In the domain of surgical video processing, one of the most explored topics is surgical phase recognition [1,2,3]. For instance, Lalys et al. [1] presented a framework to segment high-level surgical phases from 20 videos of cataract surgeries based on visual features; Padoy et al. [2] proposed a method that uses the signals from the surgical tool to model and recognize the surgical workflow of 16 cholecystectomies; and Blum et al. [3] combined both visual features and surgical tool signals to train a classification model for segmenting 7 cholecystectomy videos. However, all of these previous works depended on a model that was trained on videos whose phases had been previously fully annotated by a human annotator. Moreover, the model required training from complete workflows, which can be an issue when the workflow is not sequential or when the videos are incomplete. In contrast, our work aims to perform the video indexing in an unsupervised manner using a single video query.

Our work is closely related to video sequence matching, which is the process of finding similarities between two video sequences. Typically, such a method is performed to find videos that are relevant to the query video [4], but not to find the time boundaries of a particular task in the video. Other related methods, such as [5], focus on action recognition and work in a supervised environment using datasets that usually contain short actions. One of the closest works to ours is [6] where Chu et. al proposed the temporal commonality discoveries (TCD) method. It is an unsupervised approach to find the time boundaries of the commonalities between two sequences using a branch and bound (B&B) optimization with histogram encoding and l_1/χ^2 bounding distance. We observed that, despite the optimization, the B&B still carries out too many evaluations, thus taking a long time to find the task boundaries for one video. Moreover, the global optimality comes at the cost of a suitable similarity measure, namely TCD enforces the use of histogram encoding. In contrast, we are particularly interested in the Fisher kernel (FK) encoding. The FK has become popular since Perronnin et. al. [7] showed that it can be understood as an extension of the bag-of-words (BOW) approach (i.e. histogram encoding). It is a generic framework that combines the benefits of generative and discriminative approaches. As presented in a recent work [8], applying FK on frame-based features is also superior to BOW due to its ability to model the variation in time in the videos.

In this work, we propose a novel coarse-to-fine temporal search to find the time boundaries using FK-based similarity. The coarse-to-fine search speeds up time boundaries discovery compared to the traditional sliding window approaches.

While coarse-to-fine approaches are popular in image processing, they have not been investigated as much in such time-series data because the considered tasks are usually already short. Furthermore, unlike TCD, our method accepts any types of feature encoding and similarity measures. We will show in particular that FK-based similarity largely outperforms traditional similarities between time-series, such as dynamic time warping (DTW) [9]. We will also show that the combination of FK with coarse-to-fine temporal search gives higher performance compared to the globally optimal TCD. We carried out extensive experiments to retrieve the time boundaries of 3 tasks (i.e. intestine stapling, intestine stitching, and fat stitching) from 49 bypass videos, and one task (i.e. net placing) from 23 hernia videos. We also carried out a cross-surgery retrieval of the intestine stitching from the bypass videos to 7 sleeve gastrectomy videos.

In summary, the contributions of this paper are three-fold: (1) we tackle the problem of automatic video indexing which, to the best of our knowledge, has not previously been addressed in the medical community; it is also very different from the shot detection and action recognition problem from the traditional computer vision community; (2) we propose a coarse-to-fine temporal search combined with Fisher-kernel based similarity and show its suitability for laparoscopic video data; and (3) we present an extensive retrieval comparison with multiple techniques and similarity measures.

2 Methodology

2.1 Frame Rejection

This step is carried out to reject irrelevant frames (e.g. blank or static images, arbitrary views outside the patient's body) from the laparoscopic videos before the feature extraction process. Supervised methods, such as [10], have been presented to tackle this problem. However, to keep the whole process completely unsupervised, we propose a simple RGB histogram thresholding. Through observation, it is apparent that the red color channel is particularly more dominant compared to the other color channels in relevant frames (i.e. views inside the patient's body). A scalar value is computed to represent each color channel histogram in such a way that if the red scalar value is in a certain range and superior to the blue and green scalars, the frame will be accepted and then processed. It was shown in [11] that this approach significantly reduced the number of frames to be processed and improved the accuracy of surgery classification for laparoscopic videos.

2.2 Feature Representation

In the field of video processing, many visual-based features have been explored, such as color information, image gradients, optical flow and spatio-temporal interest points (STIP) [12]. However, based on preliminary experiments, the color histogram is not a discriminative feature since our frames look very similar to one another. Also, optical flow fails due to the rapid movement of the laparoscopic camera, and the STIP was observed to be very sensitive to specular reflection. Thus, we decided to extract histogram of gradients (HOG) [13] since it acts as a global descriptor for the video frames.

We encode the features using two approaches: bag-of-words (BOW) and fisher kernel (FK). We use the typical BOW approach using k-means clustering to build the vocabulary with a hard data-to-cluster assignment. For the fisher kernel, we represent the vocabulary as a Gaussian Mixture Model (GMM) with K Gaussians. As repeatedly suggested in many works such as [8], we also tried to reduce the dimensionality of HOG using principal component analysis (PCA) for the FK representation. However, we found during preliminary experiments that the dimensionality reduction did not bring any improvements to the overall precision and recall, so we keep the original dimensions of the features.

2.3 Video Sequence Similarity

Given two video sequences $\mathbf{A} = \begin{bmatrix} \mathbf{a}_1 \dots \mathbf{a}_{nA} \end{bmatrix}$ and $\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 \dots \mathbf{b}_{nB} \end{bmatrix}$ where nA is the number of frames in video $\mathbf{A}$ (respectively nB and $\mathbf{B}$) and $\mathbf{a}_i$ is the vector representation of the i-th frame in video $\mathbf{A}$ (respectively $\mathbf{b}_i$ and $\mathbf{B}$), the similarity between the two is defined as either $S(\mathbf{A}, \mathbf{B}) = DTW(\mathbf{A}, \mathbf{B})$ or $S(\mathbf{A}, \mathbf{B}) = D_d^e(\mathbf{A}, \mathbf{B})$. $DTW(\mathbf{A}, \mathbf{B})$ computes the similarity between $\mathbf{A}$ and $\mathbf{B}$ using DTW, while $D_d^e(\mathbf{A}, \mathbf{B})$ computes the similarity using encoding $e \in \{\text{BOW, FK}\}$ and distance d. In this paper, we consider vector distances (i.e. l_1 and l_2), histogram distance (i.e. χ^2), and also mutual information (MI).

2.4 Boundary Search

We represent our query and target videos respectively as matrices $\mathbf{Q} = \begin{bmatrix} \mathbf{q}_1 \dots \mathbf{q}_m \end{bmatrix}$ and $\mathbf{R} = \begin{bmatrix} \mathbf{r}_1 \dots \mathbf{r}_n \end{bmatrix}$ where $m < n$. The problem of video indexing is to find the best time interval $[b, e] \subseteq [1, n]$ in the target video, such that

$$(b^*, e^*) = \underset{b,e}{\arg\min}\ S(\mathbf{Q}[1, m], \mathbf{R}[b, e]), \tag{1}$$

where $\mathbf{R}[b, e] = \begin{bmatrix} \mathbf{r}_b \dots \mathbf{r}_e \end{bmatrix}$ denotes the subsequence of $\mathbf{R}$ that begins from frame b and ends in frame e, hence $\mathbf{Q}[1, m] = \mathbf{Q}$.

We initialize our algorithm by temporally partitioning the target video $\mathbf{R}$ into L overlapping segments with the size of m. The amount of overlapping depends on the predefined time step $s = \alpha m$, where $0 < \alpha < 1$ to ensure overlapping. For a target video $\mathbf{R}$, this partitioning then defines $\mathbf{R}_i = \mathbf{R}[b_i, e_i]$ where $i \in \{1, L\}$, $b_1 = 1$, $e_i = b_i + m - 1$, and $b_j = b_{j-1} + s, j \in \{2, L\}$.

We find the most similar segment to the query by computing the similarity:

$$i^* = \underset{i}{\arg\min}\ S(\mathbf{Q}, \mathbf{R}_i) \tag{2}$$

Taking $\mathbf{R}_{i^*}$ as our initial segment, we find the time interval by refining the boundary through boundary shrinking and expansion. This is carried out since the size of the queried task in video $\mathbf{R}$ is not necessarily equal to the query size m. By considering a step σ, we compute the distance of $\mathbf{Q}$ to four possible segments: $\mathbf{R}\left[b_{i^*}+\sigma, e_{i^*}\right]$, $\mathbf{R}\left[b_{i^*}-\sigma, e_{i^*}\right]$, $\mathbf{R}\left[b_{i^*}, e_{i^*}+\sigma\right]$, and $\mathbf{R}\left[b_{i^*}, e_{i^*}-\sigma\right]$. Next, we choose the best segment among the four possible ones. This process is repeated until the similarities between the query and all four possible segments are less than between the query and the initial segment.

2.5 Coarse-to-Fine Approach

In order to improve the computational time, we propose to use a coarse-to-fine approach. Both query and target videos are downsampled N times by a factor of 2. The search algorithm begins at the lowest resolution. Next, we limit our search on the higher resolutions based on the result (i.e. the time boundaries) from the search at lower resolution. This process is repeatedly done until the highest resolution is reached. This way, the total number of comparisons is reduced since we get the rough estimation of the time boundaries from the lower resolutions.

3 Experimental Results

We conducted experiments to retrieve 4 tasks: intestine stitching (IStit), intestine stapling (IStap), fat stitching (FStit) and net placing (NP). Our dataset consists of 79 surgeries performed by 8 surgeons. The details of the tasks and surgeries are shown in Table 1-b. There are only 45 videos of 49 videos for the task IStap due to incomplete recordings. For quantitative analysis, we manually annotated the time boundaries of the tasks in all videos. To evaluate the method, we performed random testing for bypass and hernia surgeries by searching 4 random queries within the remaining videos. We repeated this process 5 times. The underlying assumption was that the task was present in every target video. Methods, such as in [4], that retrieve relevant videos can be used to determine whether the task is present in the target videos. This is however not the focus of this work.

In Table 1-c, the full experimental setup is shown and the explanation of the naming conventions is given in Table 1-a. We trained BOW and GMM dictionaries with respectively 500 and 50 words. We set $\alpha = 0.1$ to get high overlapping segments and a large refinement step $\sigma = 5$. For TCD configuration, we used the χ^2 bounding function since it was shown in [6] to provide the best result. This was also confirmed in our preliminary experiments. We downsampled the videos to 0.5 frame per second (fps) for the best performance-time trade-off. For our method, we carried out extensive experiments to observe the effects of various parameters. Due to limited space, we only show the most significant results.

For evaluation, we use the method proposed in [6]. Given the ground truth of the time boundaries $\hat{T} = [\hat{b}, \hat{e}]$, the evaluation of the estimated time boundaries $T = [b, e]$ is done by computing precision $\phi_p = \frac{T \cap \hat{T}}{T}$ and recall $\phi_r = \frac{T \cap \hat{T}}{\hat{T}}$. We consider a boundary estimation to be correct in terms of precision if $\phi_p > 0.5$, meaning more than half of the frames in the estimated time boundaries is in the ground true boundaries. We also apply the similar thresholding for recall.

Table 1. Experimental setup: (a) the naming conventions for the experimental configurations; (b) the details of the tasks and surgeries, including number of surgeries and the mean ± std of the task and surgery length; and (c) the configuration of experiments complying to the conventions defined in (a), except for TCD.

Code	Description
C	Coarse-to-fine approach
B	BOW representation
F	FK representation
L	Low resolution
H	High resolution
R	Refinement

(a)

Task	Surgery	#Surg.	Avg. Len. (min.)	
			Task	Surg
IStit	Bypass	49	14 ± 5	111 ± 27
	Sleeve Gast.	7	25 ± 8	109 ± 16
FStit	Bypass	49	6 ± 2	111 ± 27
IStap	Bypass	45	7 ± 5	114 ± 26
NP	Hernia	23	4 ± 2	50 ± 25

(b)

ID	FPS	N
TCD	0.5	-
CBH-DTW	2.5	3
CBH-MI	2.5	3
CBH-χ^2	2.5	3
FL-l_1	0.5	1
FL-l_2	0.5	1
FH-l_1	2.5	1
FH-l_2	2.5	1
CFH-l_1	2.5	3
CFH-l_2	2.5	3
CFHR-l_1	2.5	3
CFHR-l_2	2.5	3

(c)

Comparison with TCD. From multiple search configurations shown in Table 1-c, we show in Table 2-a the best configurations (with the highest $F = 2 \cdot \frac{\text{rec}\cdot\text{prec}}{\text{rec}+\text{prec}}$) of our video indexing method compared to TCD. Compared to TCD, our method is significantly faster since it does less evaluations. Note that the best results are all obtained from the coarse-to-fine configurations. Thus, not only does the coarse-to-fine approach decrease the number of evaluations, but it also improves the performance of the method. This is possible because less noise is present in lower resolution, giving better initialization at the higher resolution.

For all events, our method gives the same or higher precision compared to TCD. The recall is only slightly decreased for IStit task and largely increased for all the other tasks. Out of 5 task retrievals, 4 are obtained using coarse-to-fine approaches with FK, 3 of which use the boundary refinement method.

It can be seen that the method tends to fail at retrieving IStap and NP tasks. Compared to stitching tasks, these tasks have higher variability in terms of the sub-task execution. However, the precision and recall are still much higher than TCD and chance. In summary, the complete approaches that we propose, namely CFHR-l_1 and CFHR-l_2, have higher average F-measure (respectively 35.29 and 35.36) over all retrievals compared to TCD (34.88).

Effect of Resolution, Feature Representation and Choice of Similarity Measures. To observe the effect of various parameters, we show the complete results from IStit task retrieval in Bypass videos using queries from Bypass videos (B→B) in Table 2-b. Since our method performs really fast, we have the possibility to increase the data resolution up to full resolution. In preliminary

Table 2. Experimental results. (a) Comparison of precision, recall, and mean ± std of execution time between TCD and our boundary search method. Q and T respectively stands for query and target. We use the first letter of the surgery to ID them, thus B, S, and H respectively represents bypass, sleeve gastrectomy, and hernia. For instance, B→S means looking for a task based on video query from bypass surgery in sleeve gastrectomy videos. (b) Precision and recall comparison of our method for IStit B→B.

Task	Q→T	TCD			Best of ours			
		Prec (%)	Rec (%)	Time (s)	ID	Prec (%)	Rec (%)	Time (s)
IStit	B→B	70.91	**78.26**	33 ± 21	CFHR-l_1	**78.57**	77.85	4.4 ± 3.3
	B→S	58.57	**22.14**	45 ± 24	CFHR-l_1	**59.28**	19.28	1.9 ± 0.5
FStit	B→B	37.44	38.19	21 ± 12	CBH-MI	**53.51**	**45.61**	0.3 ± 0.2
IStap	B→B	10.02	12.44	26 ± 17	CFH-l_2	**23.44**	**26**	2.4 ± 1.2
NP	H→H	19.47	18.52	9 ± 8.2	CFHR-l_2	**38.04**	**31.95**	2.2 ± 1.2

(a)

Task	Eval.	CBH			FL		FH		CFH		CFHR	
		DTW	MI	χ^2	l_1	l_2	l_1	l_2	l_1	l_2	l_1	l_2
IStit, B→B	Prec	40.91	70.12	56.42	77.65	58.36	76.12	58.16	77.95	58.57	78.57	58.77
	Recall	41.83	71.3	55.51	78.06	57.85	77.85	58.26	78.36	57.95	77.85	56.53

(b)

results, we observed that 2.5 fps gave the best performance-cost trade-off. Having data with higher resolution means that the we can retrieve the boundary more precisely. However, searching directly on the high resolution may cause wrong initialization due to the presence of noise. Thus, most of the time, the improvement from using higher resolution is obtained after the coarse-to-fine approach. This can be observed from the FL, FH and CFH configurations.

As expected, the FK representation performs better in most cases than the BOW approach, which confirms the conclusions of previous works [14,8]. In terms of similarity measures, our FK-based approach significantly outperforms the usual similarity on sequence, DTW, by over 35%. However, it can be noticed that the FK is sensitive to the distance function (i.e. l_1- and l_2-distance).

4 Conclusions

In this paper, we present a video indexing method for a laparoscopic video database using Fisher kernel based similarities. This method performs significantly better than DTW, the standard similarity measure between sequences. In addition, our method is more flexible than TCD as it can be adapted to any feature representation and similarity measure. We also compared our method with the globally optimal TCD and showed large improvements in most cases by using more suited similarities. Furthermore, we demonstrated that in addition to decreasing the number of evaluations, the coarse-to-fine approach does not impede the method's performance. In future work, we plan to try our approach on bigger datasets containing a greater number of tasks and explore more possibilities for cross-surgery retrievals. We also plan to further investigate the use of other distance functions for FK in order to find a function that will work uniformly well in all cases.

Acknowledgements. This work was supported by French state funds managed by the ANR within the Investissements d'Avenir program under references ANR-11-LABX-0004 (Labex CAMI), ANR-10-IDEX-0002-02 (IdEx Unistra) and ANR-10-IAHU-02 (IHU Strasbourg). The authors would like to thank the IRCAD audio-visual team for their help in generating the dataset.

References

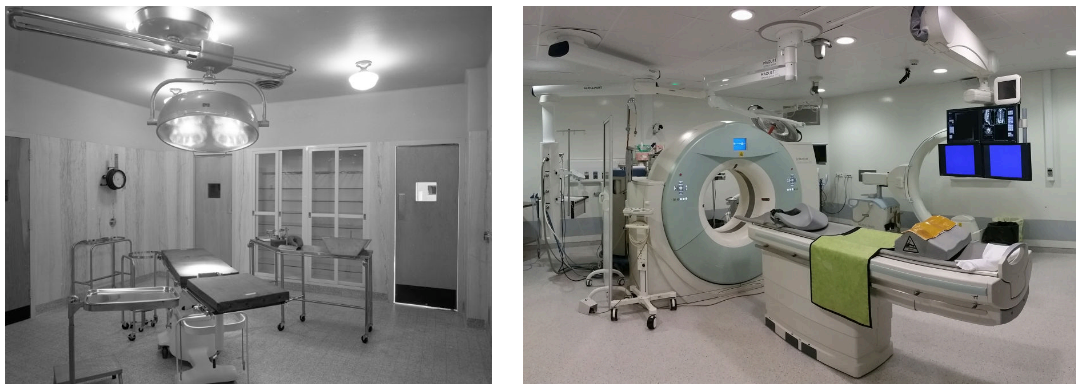
1. Lalys, F., Riffaud, L., Bouget, D., Jannin, P.: A framework for the recognition of high-level surgical tasks from video images for cataract surgeries. IEEE Trans. Biomed. Engineering 59(4), 966–976 (2012)
2. Padoy, N., Blum, T., Ahmadi, S.A., Feussner, H., Berger, M.O., Navab, N.: Statistical modeling and recognition of surgical workflow. Medical Image Analysis 16(3), 632–641 (2012)
3. Blum, T., Feußner, H., Navab, N.: Modeling and segmentation of surgical workflow from laparoscopic video. In: Jiang, T., Navab, N., Pluim, J.P.W., Viergever, M.A. (eds.) MICCAI 2010, Part III. LNCS, vol. 6363, pp. 400–407. Springer, Heidelberg (2010)
4. Chen, L.H., Chin, K.H., Liao, H.Y.: An integrated approach to video retrieval. In: 19th Australasian Database Conference. CRPIT, vol. 75, pp. 49–55. ACS (2008)
5. Jhuang, H., Gall, J., Zuffi, S., Schmid, C., Black, M.J.: Towards understanding action recognition. In: ICCV (2013)
6. Chu, W.-S., Zhou, F., De la Torre, F.: Unsupervised temporal commonality discovery. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 373–387. Springer, Heidelberg (2012)
7. Perronnin, F., Dance, C.R.: Fisher kernels on visual vocabularies for image categorization. In: CVPR (2007)
8. Mironica, I., Uijlings, J., Rostamzadeh, N., Ionescu, B., Sebe, N.: Time matters! capturing variation in time in video using fisher kernels. ACM Multimedia (2013)
9. Sakoe, H.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. on Acoustics, Speech, and Signal Processing 26, 43–49 (1978)
10. Atasoy, S., Mateus, D., Meining, A., Yang, G.Z., Navab, N.: Endoscopic video manifolds for targeted optical biopsy. IEEE Trans. Med. Imaging 31(3), 637–653 (2012)
11. Twinanda, A.P., Marescaux, J., De Mathelin, M., Padoy, N.: Towards better laparoscopic video database organization by automatic surgery classification. In: Stoyanov, D., Collins, D.L., Sakuma, I., Abolmaesumi, P., Jannin, P. (eds.) IPCAI 2014. LNCS, vol. 8498, pp. 186–195. Springer, Heidelberg (2014)
12. Laptev, I.: On space-time interest points. Int. J. Comput. Vision 64(2-3), 107–123 (2005)
13. Dalal, N., Triggs, B.: Histograms of oriented gradients for human detection. In: CVPR, pp. 886–893 (2005)
14. Chatfield, K., Lempitsky, V., Vedaldi, A., Zisserman, A.: The devil is in the details: an evaluation of recent feature encoding methods. In: BMVA, pp. 76.1–76.12 (2011)

C Résumé en français

Chapter Summary

C.1 Contexte

La chirurgie est le domaine de la médecine qui concerne la manipulation des tissus et des organes afin de diagnostiquer, prévenir et traiter des maladies. Depuis la révolution industrielle, les avancées technologiques ont transformé la chirurgie en une discipline scientifique capable de traiter de nombreuses maladies. Cette transformation permet aujourd'hui l'exécution de procédures qui sont de plus en plus complexes, mais est aussi à l'origine d'une augmentation exponentielle de la quantité d'information disponible dans une salle opératoire. Dans la figure C.1, nous montrons la comparaison entre deux salles opératoires de siècles différents. Par exemple, les différents écrans dans la salle permettent au chirurgien de suivre les signes vitaux du patient et en même temps de guider la navigation pendant la procédure. En effet, ces informations sont essentielles pour l'exécution de procédures complexes. Cependant, des informations qui ne sont



(a) (b)

Figure C.1: Scènes en deux salles opératoires de siècles differents: (a) Veterans Administration Hospital en 1953 and (b) Nouvel Hôpital Civil Strasbourg en 2015.

pas présentées de la bonne façon peuvent affecter de façon négative le déroulement d'une chirurgie. De plus, une partie de ces signaux n'est pas exploitée au maximum de son potentiel. Il existe alors un intérêt de la part de la communauté scientifique pour développer des systèmes réactifs au contexte de la salle opératoire, capables d'analyser les signaux provenant des équipements pour ainsi fournir de l'assistance aux chirurgiens. Cette assistance peut aller d'une simple visualisation des informations pertinentes de façon adéquate pendant la chirurgie, jusqu'à assister les chirurgiens dans leur tâches chirurgicales. Pour pouvoir fournir ce support, le système doit être capable de comprendre l'environnement de la salle ainsi que les interactions qui y ont lieu. Ceci est possible grâce à l'identification automatique des personnes et des objets présents dans la salle. Dans cette thèse, nous nous concentrons sur une autre tâche essentielle pour les systèmes réactifs au contexte : la reconnaissance automatique des activités dans la salle opératoire.

Être capables de reconnaître automatiquement les activités qui ont lieu dans la salle opératoire ouvre les portes à des nombreuses applications peropératoire ainsi que postopératoires. En effet, un système de reconnaissance des activités peut non seulement contribuer à montrer des informations pertinentes au chirurgien à l'instant le plus adéquat, mais il peut aussi être capable d'identifier et fournir d'autres types d'assistance au personnel. A travers une analyse plus profonde du workflow de la chirurgie, des anomalies dans l'exécution peuvent être détectées à travers l'identification des variations anormales dans les activités chirurgicales. De même, la reconnaissance des activités peut permettre d'estimer automatiquement le temps qui reste avant la fin de la chirurgie. Ceci peut être utile par exemple pour démarrer la préparation du patient suivant après un certain seuil et ainsi minimiser le temps d'attente. Des applications pour après la procédure sont également envisageables. La détection automatique d'événements clefs et des moments auxquels ils se produisent peut faciliter la rédaction de rapports, par exemple.

Grâce à l'ensemble de ses applications possibles, la reconnaissance automatique des activités dans la salle opératoire est aujourd'hui le sujet principal de nombreux travaux

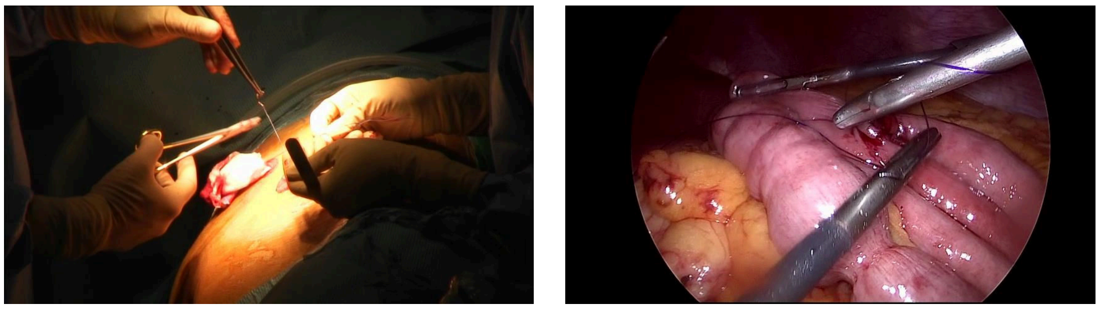

(a) (b)

Figure C.2: Exemples d'images acquises lors d'une chirurgie laparoscopique: (a) l'image enrégistrée par un caméra fixé au plafond and (b) l'image laparoscopique montrant l'anatomie interne dans la cavité abdominale du patient.

de recherche. Certains proposent d'utiliser des signaux disponibles dans la salle comme les signes vitaux du patient [Xiao 2005] ou l'utilisation des instruments [Padoy 2012]. Cependant, ce type de signaux fournit uniquement des informations spécifiques et limitées à la procédure. Leur obtention est aussi souvent difficile puisqu'ils sont collectés par des processus complexes comme par des annotations manuelles ou par l'utilisation d'outils additionnels et des marqueurs. Néanmoins, ces limitations peuvent être évitées en utilisant des enregistrements vidéos de la salle opératoire. En effet, grâce aux progrès réalisés dans les technologies des caméras, les salles opératoires modernes sont aujourd'hui équipées de différents systèmes d'imagerie pour assister le chirurgien. Ils sont utilisés en partie pour fournir une meilleure visualisation de l'anatomie interne du patient et aussi pour enregistrer les activités dans la salle pour des motifs de documentation et/ou sécurité. Les informations visuelles sont non seulement plus simples à obtenir, mais elles fournissent aussi une source d'information plus riche concernant les évènements qui ont lieu dans la salle. Pour ces raisons-là, nous nous intéressons dans cette thèse au problème de la reconnaissance automatique des activités dans la salle opératoire en utilisant des informations visuelles provenant de caméras, i.e. des enregistrements de chirurgies. Plus précisément, nous allons reconnaître des activités chirurgicales dans des vidéos laparoscopiques et dans des enregistrements de la salle obtenus grâce à des caméras RGBD.

La laparoscopie est une chirurgie mini-invasive, où une optique (endoscope) est introduite dans la cavité abdominale pour visualiser l'anatomie interne du patient. Nous montrons dans la figure C.2 des images acquises lors d'une chirurgie laparoscopique. Les images provenant de la caméra endoscopique (figure C.2-b) peuvent alors être enregistrées sans difficulté, résultant dans des vidéos laparoscopiques. Puisque ces vidéos fournissent des informations précieuses concernant les interactions entre les tissus et les outils chirurgicaux, elles font partie des vidéos les plus utilisées pour la reconnaissance d'activité chirurgicales dans ce type de procédures.

A part ces caméras qui ont une vue privilégiée des interactions entre les instruments et les tissus, d'autres caméras sont présentes dans la salle opératoire qui capturent plutôt une

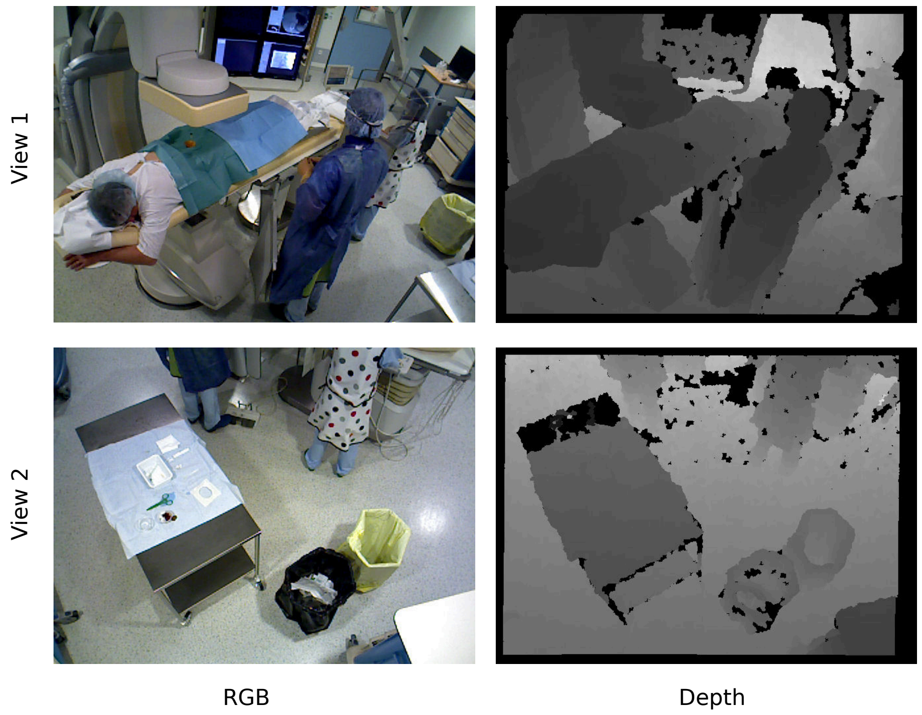


Figure C.3: Exemples d'images RGBD synchronisées du système multi-caméras.

vue globale des activités, comme les caméras de surveillance par exemple. Ces caméras sont de plus en plus utilisées dans les hôpitaux pour des raisons légales, d'enseignement, de sécurité et/ou de documentation. Elles sont aussi exploitées par des systèmes de réalité augmentée pour la salle opératoire. Comme c'est le cas pour les vidéos laparoscopiques, ces vidéos sont aussi disponibles immédiatement. Leur utilisation pour la reconnaissance d'activités est intéressante puisqu'elles fournissent des informations complémentaires aux interactions entre les tissus et les instruments capturées par les vidéos laparoscopiques. En effet, des actions génériques qui ont lieu dans la salle opératoire sont plutôt capturées, comme l'entrée et la sortie du patient ou la préparation des instruments chirurgicaux par exemple. Typiquement, des caméras pan-tilt sont utilisées pour des raisons de surveillance. Ce type de caméra enregistre des vidéos en couleur (RGB). Dans ce travail, nous utilisons des caméras qui génèrent simultanément des images couleur et des cartes de profondeur, c'est-à-dire des caméras RGBD. L'information de profondeur représente la distance de chaque pixel de l'image à la caméra. Grâce à cette information, une reconstruction 3D de la scène peut être générée, ce qui peut être exploité pour la reconnaissance automatique d'activités chirurgicales. Afin de couvrir une surface assez importante de la salle ainsi que d'éviter les occlusions, nous enregistrons la salle opératoire grâce à un système multi-caméra fixé au plafond. Dans la figure C.3, nous montrons des exemples d'image RGBD synchronisées obtenues par le système multi-caméra.

Puisque nous utilisons des vidéos comme source d'information pour la reconnaissance d'activités, des approches par Vision par Ordinateur seront présentés dans cette thèse. En effet, la reconnaissance d'activités est l'un des domaines les plus étudiés par la communauté de Vision par Ordinateur. Cependant, très peu de méthodes utilisant des informations visuelles dans un environnement médical ont été proposées. Malgré les résultats prometteurs des méthodes considérées comme l'état de l'art par la communauté, ces méthodes ne réussissent pas aussi bien dans un environnement médical. La reconnaissance d'activités implique typiquement l'identification des actions exécutées par des acteurs observées dans des vidéos. Les contraintes spécifiques de la salle opératoire comme les nombreuses occlusions causées par les objets présents dans la salle, peuvent prendre à défaut les méthodes actuelles utilisant des caractéristiques visuelles. De plus, il n'est pas facile de trouver une configuration optimale du système multi-caméras, permettant de capturer le plus d'information possible sans interférer avec la procédure et avec les équipements déjà présents dans la salle.

Comme mentionné auparavant, nous nous intéressons au problème de la reconnaissance d'activités chirurgicales dans des vidéos laparoscopiques qui, au lieu de capturer les actions des humains dans la scène, capturent les interactions entre les instruments et les structures anatomiques à l'intérieur de la cavité abdominale. Les images enregistrées sont typiquement très similaires entre elles, ce qui rend plus difficile la tâche de reconnaissance d'activités à cause d'une faible variabilité intra-classe. De plus, les difficultés visuelles des vidéos laparoscopiques, comme les réflexions spéculaires, les images floues à cause du mouvement et la présence de fumée, compliquent d'autant plus l'extraction de descripteurs visuels. Ils existent peu de travaux qui ont évalué les performances des descripteurs visuels typiquement utilisées en Vision sur ce type de vidéos pour la reconnaissance automatique d'activités.

Les objectifs de cette thèse sont : analyser les performances de méthodes de Vision par Ordinateur pour la reconnaissance automatique d'activités chirurgicales, proposer des nouvelles méthodes pour répondre aux difficultés mentionnées auparavant, et obtenir des résultats compétitifs dans des tâches de reconnaissance d'activités dans un environnement médical.

C.2 Reconnaissance d'activités dans des vidéos laparoscopiques

Nous nous intéressons à deux types de tâches de reconnaissance d'activités dans des vidéos laparoscopiques. Tout d'abord, nous souhaitons évaluer les performances des descripteurs visuels pour la reconnaissance d'activités sur ce type de vidéos qui présente des nombreuses difficultés. Nous nous interessons alors au problème au niveau le plus grossier : la classification de chirurgies [Twinanda 2014b, Twinanda 2015b]. Cette tâche consiste à identifier la nature de la chirurgie qui est exécutée dans la vidéo. Nous l'abordons par l'extraction de descripteurs visuels (i.e. histogrammes RGB et HS, SIFT, et histogrammes de gradients orientés (HOG)) dans les images et par leur agrégation grâce à une approche de type « sac de mots » (BOW). En utilisant une approche d'apprentissage à multiples noyaux, nous montrons que, malgré la faible variabilité intra-classe, notre méthode arrive à atteindre une haute précision dans la classification des chirurgies dans

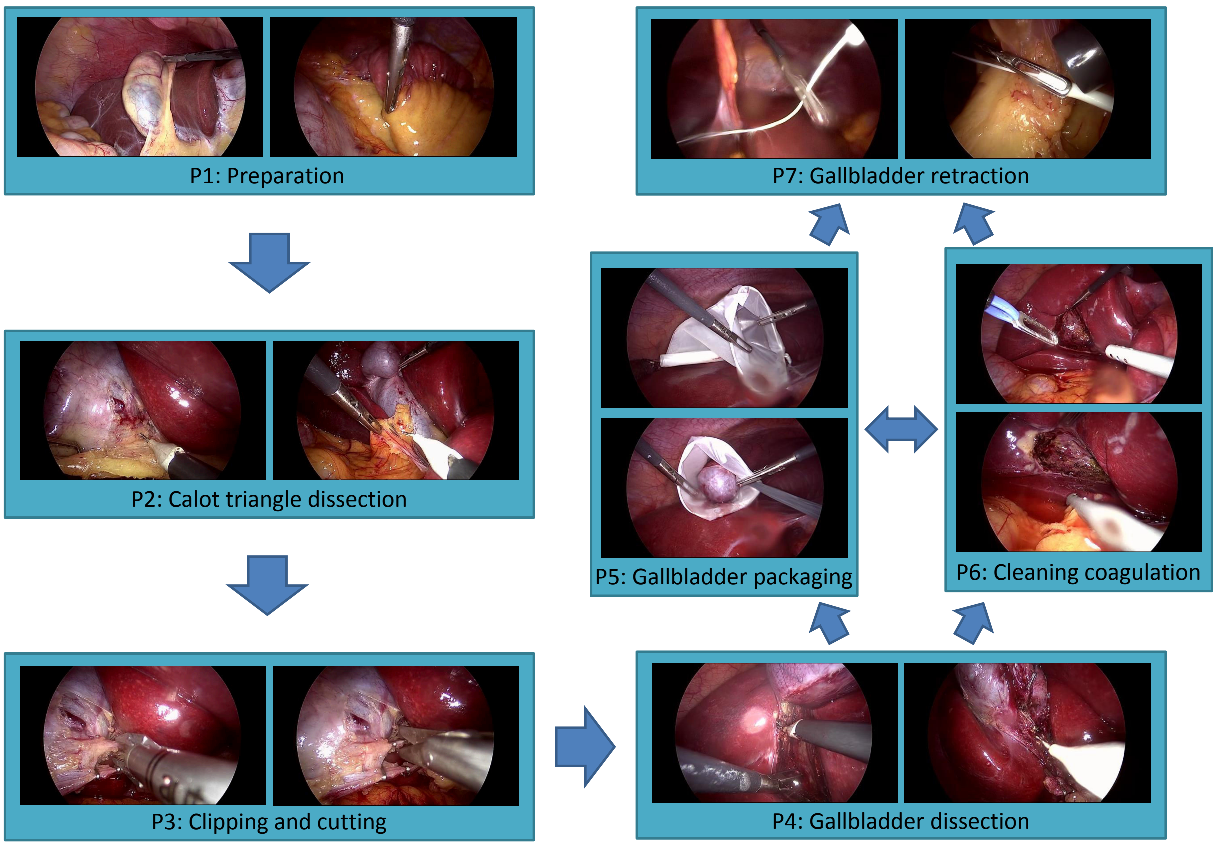

P1: Preparation
P2: Calot triangle dissection
P3: Clipping and cutting
P4: Gallbladder dissection
P5: Gallbladder packaging
P6: Cleaning coagulation
P7: Gallbladder retraction

Figure C.4: Workflow chirurgical des cholecystectomies défini dans le jeu de donnée Cholec80.

un jeu de données de 208 vidéos laparoscopiques, comprenant 8 chirurgies différentes.

Ensuite, nous nous interessons à la reconnaissance d'activités à un niveau de granularité plus fin : la reconnaissance des étapes chirurgicales [Twinanda 2017]. Il s'agit dans ce cas d'identifier l'étape de la chirurgie qui est exécutée à un moment donné. Dans ce travail, nous effectuons de la reconnaissance d'étapes chirurgicales lors des cholécystectomies, un type de chirurgie mini-invasive qui consiste à enlever la vésicule biliaire. Nous proposons d'analyser image par image : la tâche peut donc se résumer à classifier chaque image à l'étape de la procédure qui lui correspond. Puisque l'exécution d'une chirurgie suit un certain workflow, la classification peut bénéficier de contraintes temporelles qui doivent être respectées. Afin d'imposer ces contraintes temporelles, nous appliquons une méthode basée sur un Modèle de Markov Caché (HMM). Nous évaluons cette méthode sur deux jeux de données : Cholec80 [Twinanda 2017] et EndoVis [Stauder 2016]. Cholec80 contient 80 vidéos de cholécystectomies enregistrées à l'Hôpital de Strasbourg/IRCAD et EndoVis contient 7 vidéos de cholécystectomies collectées à la Klinikum Rechts der Isaar, à Munich. Nous montrons le workflow des cholécystectomies dans la figure C.4.

En utilisant les mêmes descripteurs visuels que pour la classification de chirurgies, les résultats montrent que ces descripteurs sont suffisamment discriminatoires pour

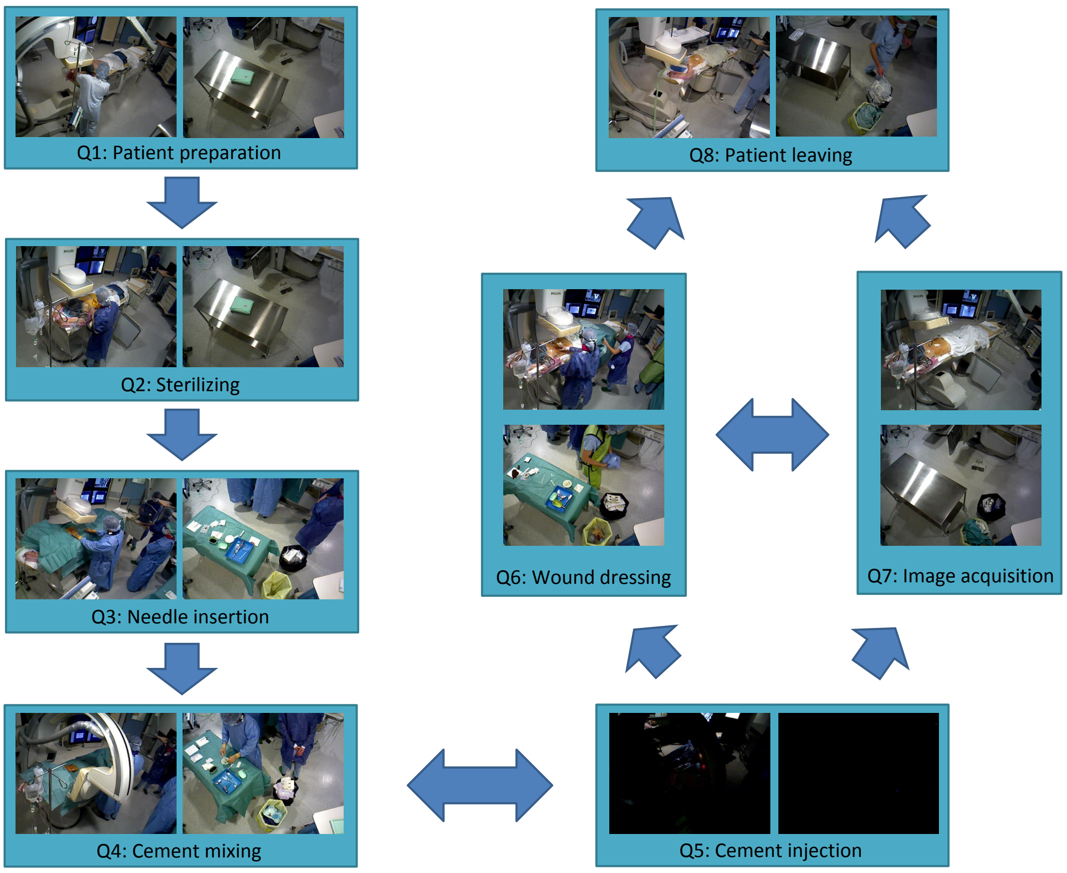


Figure C.5: Workflow chirurgical des vertébroplasties.

reconnaître les étapes de la chirurgie. Ces résultats peuvent encore être améliorés en incluant les signaux binaires sur la présence des instruments (obtenus par annotation manuelle).

C.3 Reconnaissance d'activités chirurgicales dans des vidéos RGBD acquises en salle opératoire

Comme pour les vidéos laparoscopiques, nous étudions la faisabilité d'une approche de reconnaissance d'activités dans des vidéos RGBD en essayant dans un premier temps de classifier les N actions différentes qui se déroulent dans des vidéos pré-segmentés [Twinanda 2015a]. Cette tâche diffère de la reconnaissance des étapes d'une chirurgie dans le sens où elle vise à identifier uniquement une action par vidéo, tandis que l'autre cherche plutôt à reconnaître une série d'actions dans la vidéo. Pour ceci, nous proposons d'utiliser des points d'intérêt spatiotemporels (STIP) sur les images couleur et profondeur. Afin de quantifier les descripteurs, nous proposons une extension de l'approche par « sac de mots » (BOW). L'approche proposée cherche à apprendre les informations spatiales et temporelles qui sont perdues pendant l'encodage des descripteurs. Evaluée sur un jeu de données composé de 1734 vidéos avec 15 actions différentes, cette approche présente des performances supérieures à la méthode de Pyramide Spatiale d'Histogrammes (SPM). Ceci montre qu'elle est vraiment capable de retenir plus d'informations que l'approche par SPM.

Dans un deuxième temps, comme dans le cas des vidéos laparoscopiques, nous nous interessons aussi au problème de la reconnaissance d'étapes de la chirurgie dans des vidéos RGBD. Nous nous concentrons sur des vidéos correspondant à des vertébroplasties, procédure qui consiste à injecter dans une vertèbre du ciment afin du soulager la douleur à la colonne vertébrale et restituer la mobilité. Nous montrons dans la figure C.5 le workflow chirurgical défini pour les vertébroplasties. Pour imposer les contraintes temporelles, nous proposons d'appliquer une méthode basée sur un Modèle de Markov Caché (HMM) combiné à une approche étendue de type « sac de mots ». Nous évaluons notre méthode sur un jeu de données nommé VerCArm24, composé de 24 vertébroplasties enregistrées dans une salle opératoire hybride. Les résultats montrent que l'approche étendue génère des meilleurs résultats que l'approche traditionnelle de « sac de mots ». La combinaison des caractéristiques visuelles extraites des images couleur et profondeur améliore d'autant plus les résultats.

C.4 Descripteurs « conçus manuellement » vs obtenus par apprentissage

L'ensemble des tâches mentionnées auparavant utilise des descripteurs dits « conçus manuellement », c'est-à-dire conçus empiriquement afin de capturer une certaine information provenant des images ou vidéos. Des nombreuses caractéristiques importantes peuvent alors se perdre lors de l'extraction de ces descripteurs. L'apprentissage automatique de ces descripteurs s'avère alors plus intéressant afin d'apprendre une représentation optimale des données. En Vision par Ordinateur, les méthodes de « Deep Learning » gagnent en popularité grâce à leur succès lorsqu'elles sont utilisées pour résoudre des diverses problématiques qui incluent aussi la reconnaissance d'activités. Néanmoins, leur application dans le domaine de la reconnaissance d'activités dans un environnement médical reste encore à être évaluée. Dans ce travail, nous proposons l'utilisation d'un réseau de neurones pour reconnaître les étapes chirurgicales dans des vidéos laparoscopiques ainsi que dans des vidéos enregistrées dans une salle opératoire.

C.4.1 « Deep Learning » appliqué à des vidéos laparoscopiques

Nous avons observé que, dans le cas des vidéos laparoscopiques, la combinaison des informations visuelles avec les informations binaires issues de la présence des instruments contribuait à améliorer les résultats de la détection des étapes chirurgicales. Cependant, l'information sur la présence des instruments est obtenue par un processus fastidieux d'annotation manuelle. Afin d'éliminer le besoin de ce processus manuel, nous proposons alors un réseau de neurones profond nommé EndoNet [Twinanda 2017], conçu pour faire simultanément de la reconnaissance d'étapes chirurgicales et de la détection

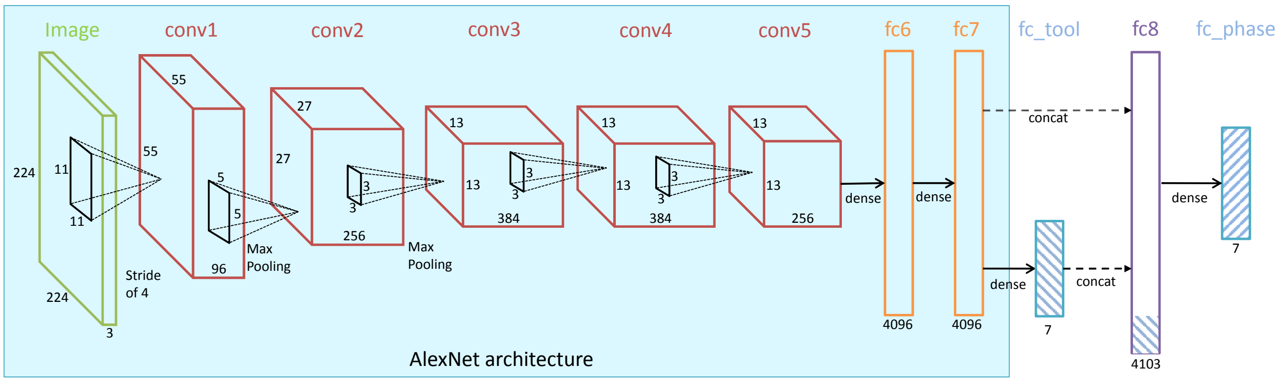


Figure C.6: Architecture de réseau de EndoNet.

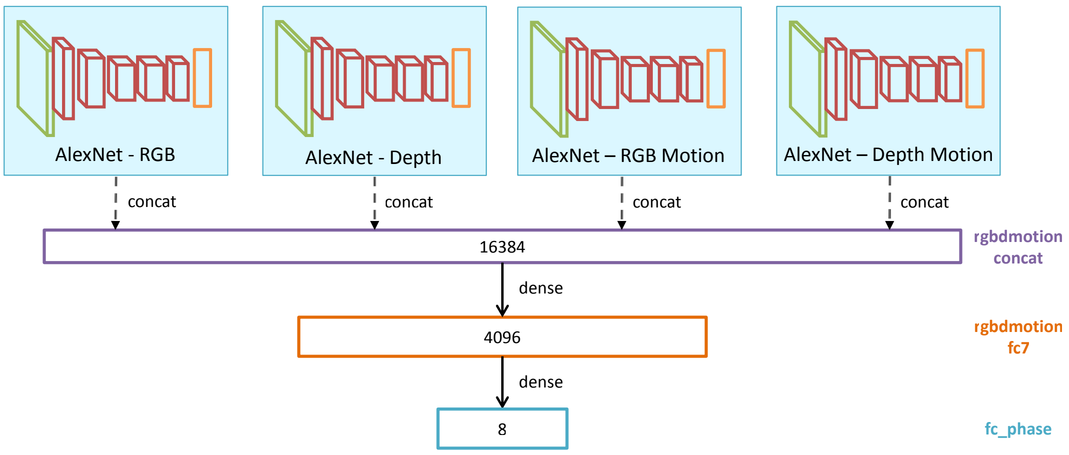


Figure C.7: Architecture à multi-flux de données.

d'instruments. Dans la figure C.6, nous montrons l'architecture de réseau de EndoNet, qui contient 5 couches de convolution (`conv1` – `conv5`), 4 couches « entièrement connectée » (`fc6` – `fc7`, `fc_tool`, `fc_phase`), et 1 couche de concaténation (`fc8`). Les résultats expérimentaux, affichés dans la table C.1, montrent que l'approche proposée (EndoNet) surpasse d'autres approches basées sur les descripteurs « conçus manuellement ».

C.4.2 « Deep Learning » appliqué à des vidéos multi-vues RGBD

Pour la reconnaissance automatique des étapes de la chirurgie dans des vidéos multi-vues RGBD, nous présentons une architecture adaptée qui profite de la nature multimodale des données (illustrée dans la figure C.7). Nous avons conçu une architecture à plusieurs flux de données qui prend en entrée les images couleur et profondeur ainsi que des cartes de mouvement. Nous proposons aussi un processus de réglage fin à deux étapes afin d'entraîner le réseau. Dans la table C.2, nous montrons les résultats de reconnaissance à partir du jeu de données VerCArm24.

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	68.4±24.1	75.7±13.6	69.2±8.0	54.5±32.3	60.2±23.8	47.5±2.6
Handcrafted	40.3±20.4	40.0±17.8	36.7±7.8	31.7±20.2	38.4±19.2	32.6±6.4
H.Crafted+CCA	54.6±23.8	57.2±21.2	61.3±8.3	39.4±31.0	41.5±21.6	38.2±5.1
H.Crafted+Tool	79.5±13.5	82.1±10.2	84.9±6.4	67.9±24.4	73.8±10.9	71.4±8.0
EndoNet	**84.8±9.1**	**88.3±5.5**	**92.0±1.4**	**73.7±16.1**	**79.6±7.9**	**81.7±4.2**

(a) Cholec80

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
Binary tool	81.4±16.1	79.5±12.3	73.0±21.5	80.3±18.1	77.5±18.8	69.8±21.7
Handcrafted	49.7±15.6	33.2±21.5	46.5±24.6	46.6±16.2	48.0±18.5	43.4±21.6
H.Crafted+CCA	66.1±22.3	64.7±22.1	61.1±17.3	52.3±22.2	49.4±21.5	44.0±22.3
H.Crafted+Tool	87.4±13.5	85.5±9.4	82.3±8.9	80.6±13.6	80.1±13.8	77.1±9.3
EndoNet	**91.0±7.7**	**87.4±10.3**	**86.0±6.3**	**83.0±12.5**	**79.2±17.5**	**76.3±5.1**

(b) EndoVis

Table C.1: Résultats de reconnaissance des étapes chirurgicales (mean ± std): (a) Cholec80 and (b) EndoVis. Le meilleur résultat de chaque métrique d'évaluation est indiqué en gras.

C.5 Réseau de type « Long-Short Term Memory » (LSTM)

Les résultats obtenus pour les vidéos laparoscopiques et pour les vidéos RGBD montrent que les descripteurs extraits grâce aux réseaux de neurones profonds ont des performances supérieures aux descripteurs « conçus manuellement » pour la tâche de reconnaissance d'étapes chirurgicales. Néanmoins, jusqu'à ce point, les contraintes temporelles du workflow chirurgical sont appliquées par HMM qui présente un désavantage: l'information temporelle qui peut être passée le long de la séquence est limitée à cause de l'hypothèse Markovienne d'ordre un. Afin d'atténuer cette limitation, nous proposons d'imposer les contraintes temporelles du workflow chirurgical dans la reconnaissance des étapes de la chirurgie grâce à un réseau LSTM, au lieu de l'approche conventionnelle par Modèle de Markov Caché (HMM). L'utilisation de LSTM nous permet aussi de concevoir une architecture de bout en bout. Toutefois, l'entraînement d'un réseau de bout en bout avec LSTM reste difficile puisqu'il implique des séquences longues qui nécessitent des grandes quantités de mémoire. Afin de s'attaquer à cette limitation, nous réalisons l'entraînement du réseau de neurones et du réseau LSTM de façon séparée. Dans la figure C.8, nous montrons l'architecture de bout en bout pour la reconnaissance des étapes chirurgicales appliquée à des vidéos laparoscopiques.

Dans la table C.3, nous affichons les résultats de reconnaissance obtenus à partir du jeu de donnée Cholec80. L'expérience montre que les résultats du réseau LSTM sont

Feature	Offline			Online		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
STIP	76.9±18.1	76.3±19.1	79.0±22.1	64.5±16.1	65.1±15.8	66.6±18.8
DSTIP	83.4±14.6	83.6±15.6	85.8±18.3	77.4±16.3	80.5±15.6	80.4±19.5
Comb.	85.0±18.0	86.3±18.2	87.2±21.9	78.5±18.0	81.0±17.8	81.3±21.6
STIP + Enc.	78.0±19.8	77.4±18.4	81.0±22.1	72.5±19.9	72.9±17.4	75.9±20.7
DSTIP + Enc.	85.3±16.9	85.7±17.5	85.9±22.3	77.6±17.0	80.3±16.9	81.9±21.5
Comb. + Enc.	84.7±18.7	86.0±17.7	86.9±22.2	80.5±17.8	82.2±16.7	82.5±21.1
Multi-stream	**93.3±6.9**	**91.7±5.8**	**96.0±2.8**	**91.2±5.7**	**89.6±5.7**	**93.7±3.0**

Table C.2: Résultats de reconnaissance des étapes chirurgicales (mean ± std) à partir du jeu de donnée VerCArm24. Le meilleur résultat de chaque métrique d'évaluation est indiqué en gras.

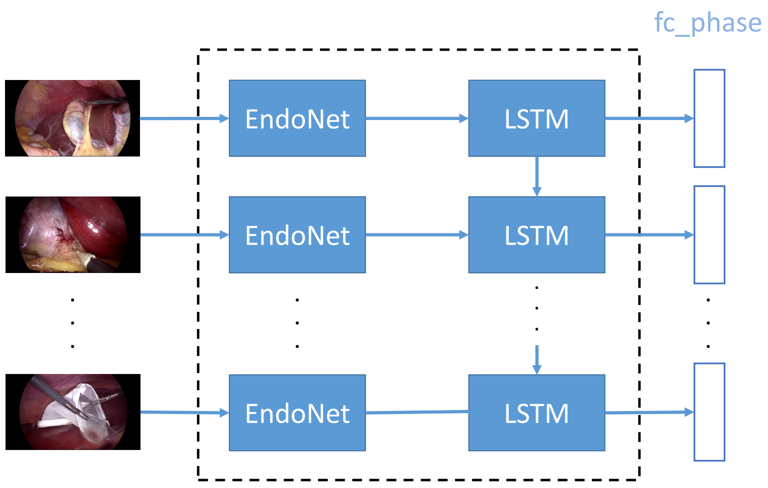


Figure C.8: Architecture de bout en bout pour la reconnaissance des étapes chirurgicales appliqué à des videos laparoscopiques.

supérieures à ceux obtenus avec l'approche de HMM et il peut alors être utilisé pour imposer les contraintes temporelles pour la reconnaissance d'étapes chirurgicales.

C.6 Perspectives

Dans cette thèse, nous avons présenté des méthodes pour la reconnaissance automatique d'activités chirurgicales en utilisant des approches de Vision sur des vidéos laparoscopiques et des vidéos d'une salle opératoire. Cependant, même si ces deux types différents de vidéos contiennent des informations sur les activités qui se déroulent dans la salle et qui peuvent être complémentaires, ils sont encore exploités de façon séparée. La combinaison des informations provenant de ces deux types de vidéos fait alors partie des perspectives de ce travail.

Le niveau de granularité plus fin qui a été abordé dans cette thèse est l'étape de la

Feature	Offline (%)			Online (%)		
	Precision	Recall	Accuracy	Precision	Recall	Accuracy
EndoNet+HMM	83.1±9.9	85.1±5.8	91.0±3.7	74.4±16.2	77.6±6.7	82.0±4.6
EndoNet+LSTM*	**88.1±6.5**	**89.0±5.7**	**92.2±7.7**	**84.4±7.9**	**84.7±7.9**	**88.6±9.6**

Table C.3: Comparaison entre des approaches de HMM et de LSTM pour la reconnaissance des étapes chirurgicales (mean ± std) à partir du jeu de donnée Cholec80. Le meilleur résultat de chaque métrique d'évaluation est indiqué en gras.

chirurgie. Dans le futur, il serait intéressant d'effectuer une analyse plus fine des activités, des interactions du personnel, et aussi de leurs interactions avec les objets présents dans la salle. De même, l'identification de l'emplacement où chaque activité a lieu est une perspective intéressante à étudier dans le futur.

References

[Aggarwal 2011] J.K. Aggarwal and M.S. Ryoo. *Human Activity Analysis: A Review.* ACM Comput. Surv., vol. 43, no. 3, pages 16:1–16:43, April 2011. (cited on page 15)

[Aggarwal 2014] J.K. Aggarwal and Lu Xia. *Human activity recognition from 3D data: A review.* Pattern Recognition Letters, vol. 48, pages 70 – 80, 2014. (cited on page 15)

[Ahmadi 2006] Seyed-Ahmad Ahmadi, Tobias Sielhorst, Ralf Stauder, Martin Horn, Hubertus Feussner and Nassir Navab. *Recovery of Surgical Workflow Without Explicit Models.* In MICCAI, pages 420–428, 2006. (cited on page 20)

[Allan 2015] Max Allan, Ping-Lin Chang, Sebastien Ourselin, David J. Hawkes, Ashwin Sridhar, John Kelly and Danail Stoyanov. *Image Based Surgical Instrument Pose Estimation with Multi-class Labelling and Optical Flow.* In MICCAI, volume 9349 of *LNCS*, pages 331–338. 2015. (cited on pages 19 and 102)

[Atasoy 2012] Selen Atasoy, Diana Mateus, Alexander Meining, Guang-Zhong Yang and Nassir Navab. *Endoscopic Video Manifolds for Targeted Optical Biopsy.* IEEE Trans. Med. Imaging, vol. 31, no. 3, pages 637–653, 2012. (cited on page 51)

[Bardram 2004] Jakob E. Bardram. *Applications of Context-aware Computing in Hospital Work: Examples and Design Principles.* In Proceedings of the 2004 ACM Symposium on Applied Computing, SAC '04, pages 1574–1579, 2004. (cited on page 3)

[Bay 2008] Herbert Bay, Andreas Ess, Tinne Tuytelaars and Luc Van Gool. *Speeded-Up Robust Features (SURF).* Comput. Vis. Image Underst., vol. 110, no. 3, pages 346–359, June 2008. (cited on page 15)

[Béjar Haro 2012] Benjamín Béjar Haro, Luca Zappella and René Vidal. *Surgical Gesture Classification from Video Data.* In MICCAI, pages 34–41, 2012. (cited on page 25)

[Bendersky 2014] Michael Bendersky, Lluis Garcia Pueyo, Vanja Josifovski, Jeremiah J. Harmsen and Dima Lepikhin. *Up Next: Retrieval Methods for Large Scale Related Video Suggestion.* In Proceedings of KDD 2014, pages 1769–1778, New York, NY, USA, 2014. (cited on page 14)

[Bengio 1994] Y. Bengio, P. Simard and P. Frasconi. *Learning Long-term Dependencies with Gradient Descent is Difficult.* Trans. Neur. Netw., vol. 5, no. 2, pages 157–166, March 1994. (cited on pages 18 and 46)

[Bharathan 2013] Rasiah Bharathan, Rajesh Aggarwal and Ara Darzi. *Operating room of the future.* Best Practice and Research Clinical Obstetrics and Gynaecology, vol. 27, no. 3, pages 311 – 322, 2013. Advances in Gynaecological Surgery. (cited on page 2)

[Bhatia 2007] Beenish Bhatia, Tim Oates, Yan Xiao and Peter Fu-Ming Hu. *Real-Time Identification of Operating Room State from Video.* In AAAI, pages 1761–1766, 2007. (cited on pages 4 and 22)

[Bishop 2006] Christopher M. Bishop. Pattern recognition and machine learning (information science and statistics). Springer-Verlag New York, Inc., 2006. (cited on pages 29, 33, and 37)

[Blum 2008] Tobias Blum, Nicolas Padoy, Hubertus Feußner and Nassir Navab. *Workflow mining for visualization and analysis of surgeries.* International Journal of Computer Assisted Radiology and Surgery, vol. 3, no. 5, pages 379–386, 2008. (cited on pages 19 and 20)

[Blum 2010] Tobias Blum, Hubertus Feussner and Nassir Navab. *Modeling and Segmentation of Surgical Workflow from Laparoscopic Video.* In MICCAI, volume 6363 of *LNCS*, pages 400–407, 2010. (cited on pages 4, 9, 19, 20, 24, 25, 62, 67, 100, and 106)

[Boser 1992] Bernhard E. Boser, Isabelle M. Guyon and Vladimir N. Vapnik. *A Training Algorithm for Optimal Margin Classifiers.* In Proceedings of the Fifth Annual Workshop on Computational Learning Theory, COLT '92, pages 144–152, 1992. (cited on page 30)

[Bouget 2015] David Bouget, Rodrigo Benenson, Mohamed Omran, Laurent Riffaud, Bernt Schiele and Pierre Jannin. *Detecting Surgical Tools by Modelling Local Appearance and Global Shape.* Trans. Medical Imaging, vol. 34, no. 12, pages 2603–2617, 2015. (cited on page 102)

[Chakraborty 2013] Ishani Chakraborty, Ahmed Elgammal and Randall S. Burd. *Video based activity recognition in trauma resuscitation.* 10th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), vol. 0, pages 1–8, 2013. (cited on pages ix, 4, 23, and 25)

[Charriere 2016] K. Charriere, G. Quellec, M. Lamard, D. Martiano, G. Cazuguel, G. Coatrieux and B. Cochener. *Real-time multilevel sequencing of cataract surgery videos.* In 2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI), pages 1–6, June 2016. (cited on pages 21, 24, and 25)

[Chaudhry 2009] R. Chaudhry, A. Ravichandran, G. Hager and R. Vidal. *Histograms of oriented optical flow and Binet-Cauchy kernels on nonlinear dynamical systems for the recognition of human actions.* In CVPR, pages 1932–1939, June 2009. (cited on page 15)

[Cheng 2007] Shinko Y. Cheng, Sangho Park and Mohan M. Trivedi. *Multi-spectral and multi-perspective video arrays for driver body tracking and activity analysis.* Computer Vision and Image Understanding, vol. 106, no. 2–3, pages 245 – 257, 2007. Special issue on Advances in Vision Algorithms and Systems beyond the Visible Spectrum. (cited on page 14)

[Choi 2008] Jaesik Choi, Won J. Jeon and Sang-Chul Lee. *Spatio-temporal Pyramid Matching for Sports Videos.* In MIR, pages 291–297, 2008. (cited on page 79)

[Cleary 2004] Kevin Cleary, Ho Young Chung and Seong Ki Mun. *OR2020 workshop overview: operating room of the future.* In CARS, volume 1268 of *International Congress Series*, pages 847–852, 2004. (cited on page 2)

[Cortes 1995] Corinna Cortes and Vladimir Vapnik. *Support-Vector Networks.* Mach. Learn., vol. 20, no. 3, pages 273–297, September 1995. (cited on page 29)

[Dalal 2005] Navneet Dalal and Bill Triggs. *Histograms of Oriented Gradients for Human Detection.* In CVPR, pages 886–893, 2005. (cited on pages 15 and 52)

[Dergachyova 2016] Olga Dergachyova, David Bouget, Arnaud Huaulmé, Xavier Morandi and Pierre Jannin. *Automatic data-driven real-time segmentation and recognition of surgical workflow.* International Journal of Computer Assisted Radiology and Surgery, vol. 11, no. 6, pages 1081–1089, 2016. (cited on pages 21, 24, 25, 52, 100, 101, and 102)

[Dollár 2005] P. Dollár, V. Rabaud, G. Cottrell and S. Belongie. *Behavior Recognition via Sparse Spatio-Temporal Features.* In VS-PETS, October 2005. (cited on pages 14, 16, 81, 86, and 88)

[Donahue 2013] Jeff Donahue, Yangqing Jia, Oriol Vinyals, Judy Hoffman, Ning Zhang, Eric Tzeng and Trevor Darrell. *DeCAF: A Deep Convolutional Activation Feature for Generic Visual Recognition.* CoRR, vol. abs/1310.1531, 2013. (cited on pages 44 and 106)

[Donahue 2015] Jeff Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko and Trevor Darrell. *Long-*

term Recurrent Convolutional Networks for Visual Recognition and Description. In CVPR, 2015. (cited on page 18)

[Felzenszwalb 2010] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester and Deva Ramanan. *Object Detection with Discriminatively Trained Part Based Models.* Trans. PAMI, vol. 32, no. 9, pages 1627–1645, 2010. (cited on pages 104 and 106)

[Fine 1998] Shai Fine, Yoram Singer and Naftali Tishby. *The Hierarchical Hidden Markov Model: Analysis and Applications.* Machine Learning, vol. 32, no. 1, pages 41–62, 1998. (cited on page 35)

[Gao 2014] Yixin Gao, S. Swaroop Vedula, Carol E. Reiley, Narges Ahmidi, Balakrishnan Varadarajan, Henry C. Lin, Lingling Tao, Luca Zappella, Benjamin Bejar, David D. Yuh, Chi Chiung Grace Chen, Rene Vidal, Sanjeev Khudanpur and Gregory D. Hager. *The JHU-ISI Gesture and Skill Assessment Dataset (JIGSAWS): A Surgical Activity Working Set for Human Motion Modeling.* In M2CAI - MICCAI Workshop, 2014. (cited on page 18)

[Girshick 2014] Ross B. Girshick, Jeff Donahue, Trevor Darrell and Jitendra Malik. *Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation.* In CVPR, pages 580–587, 2014. (cited on pages 16, 25, and 116)

[Golub 1979] Gene H. Golub, Michael Heath and Grace Wahba. *Generalized Cross-Validation as a Method for Choosing a Good Ridge Parameter.* Technometrics, vol. 21, no. 2, pages 215–223, 1979. (cited on page 30)

[Gönen 2011] Mehmet Gönen and Ethem Alpaydin. *Multiple Kernel Learning Algorithms.* J. Mach. Learn. Res., vol. 12, pages 2211–2268, July 2011. (cited on page 31)

[Hajj 2016] Hassan Al Hajj, Gwénolé Quellec, Mathieu Lamard, Guy Cazuguel and Béatrice Cochener. *Coarse-to-fine Surgical Instrument Detection for Cataract Surgery Monitoring.* CoRR, vol. abs/1609.05619, 2016. (cited on page 102)

[Han 2005] Ju Han and B. Bhanu. *Human Activity Recognition in Thermal Infrared Imagery.* In CVPR - Workshops, pages 17–17, June 2005. (cited on page 14)

[Harris 1988] Chris Harris and Mike Stephens. *A combined corner and edge detector.* In Proc. of Fourth Alvey Vision Conference, pages 147–151, 1988. (cited on page 15)

[Hinton 2012] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. *Improving neural networks by preventing co-adaptation of feature detectors.* CoRR, vol. abs/1207.0580, 2012. (cited on page 43)

[Hochreiter 1997] Sepp Hochreiter and Jürgen Schmidhuber. *Long Short-Term Memory.* Neural Comput., vol. 9, no. 8, pages 1735–1780, November 1997. (cited on pages 18, 46, and 124)

[Hwang 2016] Sangheum Hwang and Hyo-Eun Kim. *Self-Transfer Learning for Weakly Supervised Lesion Localization.* In MICCAI, pages 239–246, 2016. (cited on page 135)

[Islam 2013] Gazi Islam, Baoxin Li and Kanav Kahol. *An Affordable Real-time Assessment System for Surgical Skill Training.* In Proceedings of the Companion Publication of the 2013 International Conference on Intelligent User Interfaces Companion, IUI '13 Companion, pages 111–112, 2013. (cited on page 19)

[Jain 1996] Anil K. Jain and Aditya Vailaya. *Image retrieval using color and shape.* Pattern Recognition, vol. 29, no. 8, pages 1233 – 1244, 1996. (cited on page 15)

[Jannin 2001] P. Jannin, M. Raimbault, X. Morandi and B. Gibaud. *Modeling Surgical Procedures for Multimodal Image-Guided Neurosurgery.* In MICCAI, pages 565–572, 2001. (cited on page 19)

[Jannin 2007] P. Jannin and X. Morandi. *Surgical models for computer-assisted neurosurgery.* Neuro Image, vol. 37, no. 3, pages 783 – 791, 2007. (cited on pages 19 and 25)

[Jia 2014] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama and Trevor Darrell. *Caffe: Convolutional Architecture for Fast Feature Embedding.* arXiv preprint arXiv:1408.5093, 2014. (cited on pages 105, 119, and 126)

[Katić 2014] Darko Katić, Anna-Laura Wekerle, Fabian Gartner, Hannes Kenngott, BeatPeter Müller-Stich, Rüdiger Dillmann and Stefanie Speidel. *Knowledge-Driven Formalization of Laparoscopic Surgeries for Rule-Based Intraoperative Context-Aware Assistance.* In IPCAI, volume 8498 of *LNCS*, pages 158–167. 2014. (cited on pages 19, 25, and 135)

[Klank 2008] Ulrich Klank, Nicolas Padoy, Hubertus Feussner and Nassir Navab. *Automatic feature generation in endoscopic images.* IJCARS, vol. 3, no. 3-4, pages 331–339, 2008. (cited on page 22)

[Kranzfelder 2009] Michael Kranzfelder, Armin Schneider, Gerhard Blahusch, Hansgeorg Schaaf and Hubertus Feussner. *Feasibility of opto-electronic surgical instrument identification.* Minimally Invasive Therapy & Allied Technologies, vol. 18, no. 5, pages 253–258, 2009. (cited on page 3)

[Kranzfelder 2013] Michael Kranzfelder, Armin Schneider, Adam Fiolka, Elena Schwan, Sonja Gillen, Dirk Wilhelm, Rebecca Schirren, Silvano Reiser, Brian Jensen and Hubertus Feussner. *Real-time instrument detection in minimally invasive surgery using radiofrequency identification technology.* Journal of Surgical Research, vol. 185, no. 2, pages 704 – 710, 2013. (cited on page 102)

[Krapac 2011] Josip Krapac, Jakob Verbeek and Frédéric Jurie. *Modeling Spatial Layout with Fisher Vectors for Image Categorization.* In ICCV, pages 1487–1494, 2011. (cited on page 80)

[Krizhevsky 2012] Alex Krizhevsky, Ilya Sutskever and Geoffrey E. Hinton. *ImageNet Classification with Deep Convolutional Neural Networks.* In Advances in Neural Information Processing Systems 25, pages 1097–1105. 2012. (cited on pages xii, 17, 25, 41, 42, 43, 100, 101, and 116)

[Kuehne 2014] H. Kuehne, A. Arslan and T. Serre. *The Language of Actions: Recovering the Syntax and Semantics of Goal-Directed Human Activities.* In CVPR, pages 780–787, June 2014. (cited on page 17)

[Kushwaha 2012] A. K. S. Kushwaha, O. Prakash, A. Khare and M. H. Kolekar. *Rule based human activity recognition for surveillance system.* In Intelligent Human Computer Interaction (IHCI), 2012 4th International Conference on, pages 1–6, Dec 2012. (cited on page 14)

[Lalys 2011] Florent Lalys, Laurent Riffaud, Xavier Morandi and Pierre Jannin. *Surgical Phases Detection from Microscope Videos by Combining SVM and HMM.* In Medical Computer Vision. Recognition Techniques and Applications in Medical Imaging: International MICCAI Workshop., pages 54–62, 2011. (cited on page 20)

[Lalys 2012] Florent Lalys, Laurent Riffaud, David Bouget and Pierre Jannin. *A Framework for the Recognition of High-Level Surgical Tasks From Video Images for Cataract Surgeries.* Trans. Biomed. Engineering, vol. 59, no. 4, pages 966–976, 2012. (cited on pages 4, 19, 21, 22, 24, 25, 52, 100, and 102)

[Lalys 2014] Florent Lalys and Pierre Jannin. *Surgical process modelling: a review.* IJCARS, vol. 9, no. 3, pages 495–511, 2014. (cited on pages ix, 9, and 20)

[Laptev 2005] Ivan Laptev. *On Space-Time Interest Points.* Int. J. Comput. Vision, vol. 64, no. 2-3, pages 107–123, September 2005. (cited on pages xi, 14, 15, and 81)

[Laptev 2008] Ivan Laptev, Marcin Marszałek, Cordelia Schmid and Benjamin Rozenfeld. *Learning Realistic Human Actions from Movies.* In CVPR, 2008. (cited on page 79)

[Lazebnik 2006] Svetlana Lazebnik, Cordelia Schmid and Jean Ponce. *Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories.* In CVPR, pages 2169–2178, 2006. (cited on pages 25, 79, 82, 92, and 134)

[Lea 2013] Colin Lea, James C Facker, Gregory D. Hager, Russell H. Taylor and Suchi Saria. *3D Sensing Algorithms Towards Building an Intelligent Intensive Care Unit.* In AMIA Summits on Translational Science, 2013. (cited on pages ix, 4, 23, and 24)

[Lea 2016] Colin Lea, Austin Reiter, René Vidal and Gregory D. Hager. *Segmental Spatiotemporal CNNs for Fine-Grained Action Segmentation.* In ECCV, 2016. (cited on pages 22 and 26)

[LeCun 1998] Yann LeCun, Léon Bottou, Genevieve B. Orr and Klaus-Robert Müller. *Effiicient BackProp.* In Neural Networks: Tricks of the Trade, This Book is an Outgrowth of a 1996 NIPS Workshop, pages 9–50, 1998. (cited on pages 41 and 42)

[Lin 2006] Henry C Lin, I Shafran, David D Yuh and Gregory D. Hager. *Towards Automatic Skill Evaluation: Detection and Segmentation of Robot-Assisted Surgical Motions.* vol. 11, pages 220–230, 2006. (cited on page 9)

[Lin 2008] Weiyao Lin, Ming-Ting Sun, R. Poovandran and Zhengyou Zhang. *Human activity recognition for video surveillance.* In 2008 IEEE International Symposium on Circuits and Systems, pages 2737–2740, May 2008. (cited on page 14)

[Lin 2011] Yuanqing Lin, Fengjun Lv, Shenghuo Zhu, Ming Yang, Timothee Cour, Kai Yu, Liangliang Cao and Thomas Huang. *Large-scale image classification: Fast feature extraction and SVM training.* In CVPR, pages 1689–1696, June 2011. (cited on page 17)

[Liu 2015] Qingfeng Liu and Chengjun Liu. *A novel locally linear KNN model for visual recognition.* In CVPR, pages 1329–1337, June 2015. (cited on page 17)

[Lo 2003] Benny P.L. Lo, Ara Darzi and Guang-Zhong Yang. *Episode Classification for the Analysis of Tissue/Instrument Interaction with Multiple Visual Cues.* In MICCAI, volume 2878 of *LNCS*, pages 230–237, 2003. (cited on page 20)

[Lowe 2004] David G. Lowe. *Distinctive Image Features from Scale-Invariant Keypoints.* International Journal of Computer Vision, vol. 60, no. 2, pages 91–110, 2004. (cited on pages 15 and 52)

[Lucas 1981] Bruce D. Lucas and Takeo Kanade. *An Iterative Image Registration Technique with an Application to Stereo Vision.* In Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2, pages 674–679, 1981. (cited on page 81)

[Ma 2016] Shugao Ma, Leonid Sigal and Stan Sclaroff. *Learning Activity Progression in LSTMs for Activity Detection and Early Detection.* In CVPR, June 2016. (cited on pages 18 and 26)

[Mahasseni 2016] Behrooz Mahasseni and Sinisa Todorovic. *Regularizing Long Short Term Memory With 3D Human-Skeleton Sequences for Action Recognition.* In CVPR, June 2016. (cited on pages 18 and 26)

[Malpani 2016] Anand Malpani, Colin Lea, Chi Chiung Grace Chen and Gregory D. Hager. *System events: readily accessible features for surgical phase detection.*

International Journal of Computer Assisted Radiology and Surgery, vol. 11, no. 6, pages 1201–1209, 2016. (cited on page 3)

[Manjunath 1996] B. S. Manjunath and W. Y. Ma. *Texture features for browsing and retrieval of image data.* IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 8, pages 837–842, Aug 1996. (cited on page 15)

[Meyer 2007] Mark A. Meyer, Wilton C. Levine, Marie T. Egan, Brett J. Cohen, Gabriel Spitz, Patricia Garcia, Henry Chueh and Warren S. Sandberg. *A computerized perioperative data integration and display system.* International Journal of Computer Assisted Radiology and Surgery, vol. 2, no. 3, pages 191–202, 2007. (cited on page 3)

[Muenzer 2013] Bernd Muenzer, Klaus Schoeffmann and Laszlo Boszormenyi. *Relevance Segmentation of Laparoscopic Videos.* In IEEE International Symposium on Multimedia, pages 84–91, 2013. (cited on page 51)

[Murthy 2010] G. R. S. Murthy and R. S. Jadon. *Hand gesture recognition using neural networks.* In Advance Computing Conference (IACC), 2010 IEEE 2nd International, pages 134–138, Feb 2010. (cited on page 14)

[Nara 2011] Atsushi Nara, Kiyoshi Izumi, Hiroshi Iseki, Takashi Suzuki, Kyojiro Nambu and Yasuo Sakurai. Surgical workflow monitoring based on trajectory data mining, pages 283–291. 2011. (cited on page 3)

[Neumuth 2012] Thomas Neumuth and Christian Meissner. *Online recognition of surgical instruments by information fusion.* IJCARS, vol. 7, no. 2, pages 297–304, 2012. (cited on page 102)

[Ni 2013] Bingbing Ni, Gang Wang and Pierre Moulin. *RGBD-HuDaAct: A Color-Depth Video Database for Human Daily Activity Recognition.* In Consumer Depth Cameras for Computer Vision: Research Topics and Applications, pages 193–208, 2013. (cited on page 15)

[Padoy 2007a] N. Padoy, T. Blum, I. Essa, Hubertus Feussner, M. O. Berger and Nassir Navab. *A Boosted Segmentation Method for Surgical Workflow Analysis.* In MICCAI, pages 102–109, 2007. (cited on page 20)

[Padoy 2007b] Nicolas Padoy, Martin Horn, Hubertus Feussner, Marie-Odile Berger and Nassir Navab. *Recovery of Surgical Workflow: a Model-based Approach.* In 21st International Congress and Exhibition - Computer Assisted Radiology and Surgery - CARS 2007, Berlin, Germany, June 2007. (cited on page 10)

[Padoy 2008] Nicolas Padoy, Tobias Blum, Hubertus Feussner, Marie-Odile Berger and Nassir Navab. *On-line Recognition of Surgical Activity for Monitoring in the Operating Room.* In IAAI, pages 1718–1724, 2008. (cited on pages 20 and 21)

[Padoy 2009] Nicolas Padoy, Diana Mateus, Daniel Weinland, Marie-Odile Berger and Nassir Navab. *Workflow monitoring based on 3D motion features.* In VOEC Workshop at ICCV, pages 585–592, 2009. (cited on pages 19, 22, 25, and 105)

[Padoy 2012] Nicolas Padoy, Tobias Blum, Seyed-Ahmad Ahmadi, Hubertus Feussner, Marie-Odile Berger and Nassir Navab. *Statistical modeling and recognition of surgical workflow.* Medical Image Analysis, vol. 16, no. 3, pages 632–641, 2012. (cited on pages 3, 9, 13, 62, 68, 101, 102, and 163)

[Parlak 2012] Siddika Parlak, Aleksandra Sarcevic, Ivan Marsic and Randall S. Burd. *Introducing {RFID} technology in dynamic and time-critical medical settings: Requirements and challenges.* Journal of Biomedical Informatics, vol. 45, no. 5, pages 958 – 974, 2012. Text Mining and Natural Language Processing in Pharmacogenomics. (cited on page 3)

[Primus 2016] M. J. Primus, K. Schoeffmann and L. Böszörmenyi. *Temporal segmentation of laparoscopic videos into surgical phases.* In 2016 14th International Workshop on Content-Based Multimedia Indexing (CBMI), pages 1–6, June 2016. (cited on page 102)

[Quellec 2014a] Gwénolé Quellec, Katia Charrière, Mathieu Lamard, Zakarya Droueche, Christian Roux, Béatrice Cochener and Guy Cazuguel. *Real-time recognition of surgical tasks in eye surgery videos.* Medical Image Analysis, vol. 18, no. 3, pages 579–590, 2014. (cited on page 21)

[Quellec 2014b] Gwenole Quellec, Mathieu Lamard, Beatrice Cochener and Guy Cazuguel. *Real-Time Segmentation and Recognition of Surgical Tasks in Cataract Surgery Videos.* Trans. Medical Imaging, vol. 33, no. 12, pages 2352–2360, Dec 2014. (cited on page 21)

[Quellec 2015] Gwenole Quellec, Mathieu Lamard, Beatrice Cochener and Guy Cazuguel. *Real-Time Task Recognition in Cataract Surgery Videos Using Adaptive Spatiotemporal Polynomials.* Trans. Medical Imaging, vol. 34, no. 4, pages 877–887, 2015. (cited on pages 4, 9, and 21)

[Rabiner 1990] Lawrence R. Rabiner. *Readings in Speech Recognition.* chapter A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pages 267–296. 1990. (cited on page 33)

[Rapantzikos 2009] K. Rapantzikos, Y. Avrithis and S. Kollias. *Dense saliency-based spatiotemporal feature points for action recognition.* In CVPR, pages 1454–1461, June 2009. (cited on page 14)

[Rautaray 2010] Siddharth Swarup Rautaray and Anupam Agrawal. *A Novel Human Computer Interface Based on Hand Gesture Recognition Using Computer Vision Techniques.* In Proceedings of the First International Conference on Intelligent

Interactive Technologies and Multimedia, IITM '10, pages 292–296, 2010. (cited on page 14)

[Reiter 2012] Austin Reiter, Peter K. Allen and Tao Zhao. *Feature Classification for Tracking Articulated Surgical Tools.* In MICCAI, volume 7511 of *LNCS*, pages 592–600. 2012. (cited on page 102)

[Reiter 2016] Austin Reiter, Andy Ma, Nishi Rawat, Christine Shrock and Suchi Saria. *Process Monitoring in the Intensive Care Unit: Assessing Patient Mobility Through Activity Analysis with a Non-Invasive Mobility Sensor.* In MICCAI, pages 482–490, 2016. (cited on pages 23 and 24)

[Rieke 2015] Nicola Rieke, David Joseph Tan, Mohamed Alsheakhali, Federico Tombari, Chiara Amat di San Filippo, Vasileios Belagiannis, Abouzar Eslami and Nassir Navab. *Surgical Tool Tracking and Pose Estimation in Retinal Microsurgery.* In MICCAI, volume 9349 of *LNCS*, pages 266–273, 2015. (cited on page 102)

[Rumelhart 1988] David E. Rumelhart, Geoffrey E. Hinton and Ronald J. Williams. *Neurocomputing: Foundations of Research.* chapter Learning Representations by Back-propagating Errors, pages 696–699. 1988. (cited on page 18)

[Russakovsky 2015] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg and Li Fei-Fei. *ImageNet Large Scale Visual Recognition Challenge.* IJCV, vol. 115, no. 3, pages 211–252, 2015. (cited on page 100)

[Ryoo 2009] M. S. Ryoo and J. K. Aggarwal. *Semantic Representation and Recognition of Continued and Recursive Human Activities.* International Journal of Computer Vision, vol. 82, no. 1, pages 1–24, 2009. (cited on page 17)

[Sakoe 1978] H. Sakoe and S. Chiba. *Dynamic programming algorithm optimization for spoken word recognition.* IEEE Transactions on Acoustics, Speech, and Signal Processing, vol. 26, no. 1, pages 43–49, Feb 1978. (cited on page 17)

[Sanchez 2011] Jorge Sanchez and Florent Perronnin. *High-dimensional Signature Compression for Large-scale Image Classification.* In CVPR, pages 1665–1672, Washington, DC, USA, 2011. (cited on page 17)

[Schuster 1997] Mike Schuster and Kuldip K. Paliwal. *Bidirectional Recurrent Neural Networks.* Trans. Sig. Proc., vol. 45, no. 11, pages 2673–2681, November 1997. (cited on page 47)

[Simonyan 2014a] Karen Simonyan and Andrew Zisserman. *Two-Stream Convolutional Networks for Action Recognition in Videos.* CoRR, 2014. (cited on page 116)

[Simonyan 2014b] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition.* CoRR, vol. abs/1409.1556, 2014. (cited on page 22)

[Speidel 2009] Stefanie Speidel, Julia Benzko, Sebastian Krappe, Gunther Sudra, Pedram Azad, Beat Peter Müller-Stich, Carsten Gutt and Rüdiger Dillmann. *Automatic classification of minimally invasive instruments based on endoscopic image sequences.* In SPIE, volume 7261, pages 72610A–72610A–8, 2009. (cited on pages 19 and 102)

[Srivastava 2014] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever and Ruslan Salakhutdinov. *Dropout: A Simple Way to Prevent Neural Networks from Overfitting.* J. Mach. Learn. Res., vol. 15, no. 1, pages 1929–1958, January 2014. (cited on page 43)

[Stauder 2014] Ralf Stauder, Aslı Okur, Loïc Peter, Armin Schneider, Michael Kranzfelder, Hubertus Feussner and Nassir Navab. *Random Forests for Phase Detection in Surgical Workflow Analysis.* In IPCAI, volume 8498 of *LNCS*, pages 148–157. 2014. (cited on pages 3, 13, 62, 65, 101, and 102)

[Stauder 2016] Ralf Stauder, Daniel Ostler, Michael Kranzfelder, Sebastian Koller, Hubertus Feußner and Nassir Navab. *The TUM LapChole dataset for the M2CAI 2016 workflow challenge.* CoRR, vol. abs/1610.09278, 2016. (cited on pages 11, 143, and 166)

[Sung 2012] Jaeyong Sung, C. Ponce, B. Selman and A. Saxena. *Unstructured human activity detection from RGBD images.* In ICRA, pages 842–849, May 2012. (cited on page 15)

[Sutherland 2006] J. Sutherland and W. J. van den Heuvel. *Towards an Intelligent Hospital Environment: Adaptive Workflow in the OR of the Future.* In Proceedings of the 39th Annual Hawaii International Conference on System Sciences, volume 5, page 100b, January 2006. (cited on page 8)

[Szegedy 2015] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke and Andrew Rabinovich. *Going Deeper with Convolutions.* In CVPR, 2015. (cited on page 43)

[Sznitman 2014] Raphael Sznitman, Carlos J. Becker and Pascal Fua. *Fast Part-Based Classification for Instrument Detection in Minimally Invasive Surgery.* In MICCAI, pages 692–699, 2014. (cited on pages 19 and 102)

[Tao 2013] Lingling Tao, Luca Zappella, Gregory D. Hager and René Vidal. *Surgical Gesture Segmentation and Recognition.* In MICCAI, pages 339–346, 2013. (cited on page 9)

[Toshev 2014] A. Toshev and C. Szegedy. *DeepPose: Human Pose Estimation via Deep Neural Networks.* In CVPR, pages 1653–1660, June 2014. (cited on page 16)

[Toti 2015] G. Toti, M. Garbey, V. Sherman, B. L. Bass and B. J. Dunkin. *A smart trocar for automatic tool recognition in laparoscopic surgery.* Surg Innov, vol. 22, no. 1, pages 77–82, Feb 2015. (cited on page 3)

[Tran 2015] D. Tran, L. Bourdev, R. Fergus, L. Torresani and M. Paluri. *Learning Spatiotemporal Features with 3D Convolutional Networks.* In ICCV, pages 4489–4497, Dec 2015. (cited on pages 16, 17, 25, and 116)

[Twinanda 2014a] Andru Putra Twinanda, Michel De Mathelin and Nicolas Padoy. *Fisher Kernel Based Task Boundary Retrieval in Laparoscopic Database with Single Video Query.* In MICCAI, pages 409–416. 2014. (cited on page 11)

[Twinanda 2014b] Andru Putra Twinanda, Jacques Marescaux, Michel De Mathelin and Nicolas Padoy. *Towards Better Laparoscopic Video Database Organization by Automatic Surgery Classification.* In IPCAI, pages 186–194, 2014. (cited on pages 12 and 165)

[Twinanda 2015a] Andru Putra Twinanda, Emre O. Alkan, Afshin Gangi, Michel de Mathelin and Nicolas Padoy. *Data-driven spatio-temporal RGBD feature encoding for action recognition in operating rooms.* Int. J. Computer Assisted Radiology and Surgery, vol. 10, no. 6, pages 737–747, 2015. (cited on pages 12 and 167)

[Twinanda 2015b] Andru Putra Twinanda, Jacques Marescaux, Michel de Mathelin and Nicolas Padoy. *Classification approach for automatic laparoscopic video database organization.* Int. J. Computer Assisted Radiology and Surgery, vol. 10, no. 9, pages 1449–1460, 2015. (cited on pages 12 and 165)

[Twinanda 2016] Andru Putra Twinanda, Pramita Winata, Afshin Gangi, Michel De Mathelin and Nicolas Padoy. *Multi-Stream Deep Architecture for Surgical Phase Recognition on Multi-View RGBD Videos.* In M2CAI Workshop at MICCAI, 2016. (cited on page 12)

[Twinanda 2017] Andru Putra Twinanda, Sherif Shehata, Didier Mutter, Jacques Marescaux, Michel de Mathelin and Nicolas Padoy. *EndoNet: A Deep Architecture for Recognition Tasks on Laparoscopic Videos.* IEEE Transactions on Medical Imaging, vol. 36, no. 1, pages 86–97, Jan 2017. (cited on pages 12, 166, and 168)

[Varma 2009] Manik Varma and Rakesh Bodla Babu. *More Generality in Efficient Multiple Kernel Learning.* In ICML, pages 1065–1072, 2009. (cited on page 53)

[Vedaldi 2010] Andrea Vedaldi and Brian Fulkerson. *Vlfeat: An Open and Portable Library of Computer Vision Algorithms.* In ICM, pages 1469–1472, 2010. (cited on page 54)

[Viterbi 1967] Andrew J. Viterbi. *Error bounds for convolutional codes and an asymptotically optimum decoding algorithm.* Trans. on Information Theory, vol. 13, no. 2, pages 260–269, 1967. (cited on page 33)

[Voros 2007] Sandrine Voros, Jean-Alexandre Long and Philippe Cinquin. *Automatic Detection of Instruments in Laparoscopic Images: A First Step Towards High-level Command of Robotic Endoscopic Holders.* Int. J. Rob. Res., vol. 26, no. 11-12, pages 1173–1190, November 2007. (cited on page 19)

[Wallhoff 2007] F. Wallhoff, M. RuB, G. Rigoll, J. Gobel and H. Diehl. *Surveillance and Activity Recognition with Depth Information.* In 2007 IEEE International Conference on Multimedia and Expo, pages 1103–1106, July 2007. (cited on page 14)

[Wang 2015] A. Wang, J. Cai, J. Lu and T. J. Cham. *MMSS: Multi-modal Sharable and Specific Feature Learning for RGB-D Object Recognition.* In ICCV, pages 1125–1133, Dec 2015. (cited on page 116)

[Wu 2007] J. Wu, A. Osuntogun, T. Choudhury, M. Philipose and J. M. Rehg. *A Scalable Approach to Activity Recognition based on Object Use.* In ICCV, pages 1–8, Oct 2007. (cited on page 15)

[Xia 2013] L. Xia and J. K. Aggarwal. *Spatio-temporal Depth Cuboid Similarity Feature for Activity Recognition Using Depth Camera.* In CVPR, pages 2834–2841, June 2013. (cited on pages 16, 17, 81, 86, and 88)

[Xia 2015] Lu Xia, Ilaria Gori, Jake K. Aggarwal and Michael S. Ryoo. *Robot-centric Activity Recognition from First-Person RGB-D Videos.* In WACV, pages 357–364, 2015. (cited on page 15)

[Xiao 2005] Yan Xiao, Peter Hu, Hao Hu, Danny Ho, Franklin Dexter, Colin F. Mackenzie, F. Jacob Seagull and Richard P. Dutton. *An algorithm for processing vital sign monitoring data to remotely identify operating room occupancy in real-time.* Anesthesia and Analgesia, vol. 101, no. 3, pages 823–829, 9 2005. (cited on pages 3, 13, 22, and 163)

[Xu 2012] Ran Xu, Priyanshu Agarwal, Suren Kumar, Venkat N. Krovi and Jason J. Corso. *Combining Skeletal Pose with Local Motion for Human Activity Recognition.* In Articulated Motion and Deformable Objects: 7th International Conference, AMDO 2012, pages 114–123, 2012. (cited on pages 15 and 16)

[Yosinski 2015] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs and Hod Lipson. *Understanding Neural Networks Through Deep Visualization.* In Deep Learning Workshop, International Conference on Machine Learning (ICML), 2015. (cited on pages x, 43, and 44)

[Zappella 2013] Luca Zappella, Benjamín Béjar, Gregory Hager and René Vidal. *Surgical gesture classification from video and kinematic data.* Medical Image Analysis, vol. 17, no. 7, pages 732 – 745, 2013. (cited on pages 9, 20, 22, 25, and 100)

[Zelnik-Manor 2001] L. Zelnik-Manor and M. Irani. *Event-based analysis of video.* In CVPR, volume 2, pages II–123–II–130 vol.2, 2001. (cited on page 14)