Fine-Grained Activity Modeling, Recognition, and Error Analysis in Robot-Assisted Surgery

A

Dissertation

Presented to

the faculty of the School of Engineering and Applied Science

University of Virginia

in partial fulfillment

of the requirements for the degree

Doctor of Philosophy

by

Kay Hutchinson

December 2023

APPROVAL SHEET

This

Dissertation

is submitted in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

[illegible]

Author: Kay Hutchinson

This Dissertation has been read and approved by the examining committee:

Advisor:	Homa Alemzadeh
Chair:	Scott Acton
Committee Member:	Noah Schenkman
Committee Member:	John A. Stankovic
Committee Member:	Afsaneh Doryab
Committee Member:	Todd DeLong

Accepted for the School of Engineering and Applied Science:

[illegible]

Jennifer L. West, School of Engineering and Applied Science

December 2023

Acknowledgements

First, I would like to take this opportunity to express my deepest gratitude to my advisor, Prof. Homa Alemzadeh, for her mentorship throughout the course of my doctoral research. Without her expertise and dedication, this dissertation would not have been possible.

I'm extremely grateful to the members of my committee, Prof. Scott Acton, Dr. Noah Schenkman, Prof. John A. Stankovic, Prof. Afsaneh Doryab, and Prof. Todd DeLong for their valuable support and guidance.

My research was funded by the National Science Foundation (NSF) through the NSF Research Traineeship (NRT) and Graduate Research Fellowship Program (GRFP) fellowships, as well as the UVA Dean's Scholar Fellowship and funding from the Engineering-in-Medicine (EIM) center.

I am thankful to the members of the UVA Dependable Systems and Analytics (UVA-DSA) group: Maryam Khorasgani, Xugui Zhou, Zongyu (Zoey) Li, Keshara Weerasinghe, Hamid Roodabeh, and Xueren Ge. Special appreciation goes to Ian Reyes and Mohammad Samin Yasar for their collaboration and encouragement. I'm also thankful for help from Harshneet Bhatia, Rohan Chandra, Devin Gardner, Meghana Guttikonda, Saahith Janapati, Eli Jelesko, Sion Kim, Casey Ladd, Sara Liu, Asha Maran, Devon Parikh, Joyce Park, Shaunak Sinha, Erick Tian, Ray Wang, and Shrisha Yapalparvi. I'm honored to have had the opportunity to work with every one of you.

In addition, I would like to extend my heartfelt appreciation to my collaborators Dr. Schenkman, Dr. Leigh Ann Cantrell, and Dr. Katherina (Tina) Chen whose expertise enabled us to approach problems from multiple perspectives and identify important insights. Their collaboration and support have significantly enriched the research presented in this dissertation.

Furthermore, I would like to thank the members of the Link Lab and the Electrical and Computer Engineering Department for providing an excellent academical environment. I greatly appreciate Beth Eastwood-Beatty, Travis Hite, Kelley Tobler, and Jennifer Burman for organizing engaging and educational events, and for their administrative support and assistance. I have many fond memories of fun times and friendship with the Link Lab community. Also, I sincerely thank Prof. Harry Powell for his mentorship and for sharing his sage advice and wisdom with me throughout my studies.

I'd like to acknowledge my dear friends Rachel and Hannah who are sharing this journey with me as they pursue their own doctorates. And I'd also like to recognize the friends I've made along the way, Liz and Ian. Your friendship means the world to me.

Finally, thank you Mom and Dad for your love and support. I'm so lucky to have been blessed with such great parents.

Abstract

Surgical robots are complex cyber-physical systems driving the development of new features and technologies to improve efficiency and patient safety in surgery. This dissertation focuses on three major challenges that hinder the development of surgical robots: small datasets with low diversity and incompatible activity labels, limited generalizability and interpretability of black box activity recognition models, and limited attention to identifying executional errors during surgeon training and skill assessment. We propose a formal framework for the fine-grained modeling of surgical tasks using context and motion primitives. This framework directly relates interactions among tools and objects within the surgical environment encoded as context, to surgical workflow described by motion primitives, and enables the modeling of tasks as finite state machines. Then, we develop a method for labeling context based on video data that results in objective, fine-grained annotations with near-perfect agreement among non-expert annotators and expert surgeons. Using our framework, we create the COntext and Motion Primitive Aggregate Surgical Set (COMPASS) containing kinematic and video data with consistent labels from six different surgical dry lab tasks that nearly triples the amount of data for modeling and analysis. To understand the relationship between different levels of granularity, we use this framework and dataset to develop models for the inference of surgical context from video data and the recognition of motion primitives and gestures from kinematic data. We propose the novel Leave-One-Task-Out (LOTO) cross validation setup that evaluates the generalizability of activity recognition models to unseen tasks. We find that aggregating data across datasets supports the task-generalization of motion primitive recognition models. Then, we develop novel activity-aware methods for the identification and analysis of errors based on the surgical task and activity recognition models created in the previous thrusts. We use these methods to identify interpretable patterns in fine-grained surgical activities that strongly correlate with measures of surgical skill and can be used to improve feedback in surgeon training and skill assessment.

Contents

List of Abbreviations and Acronyms

AP	Average Precision
COMPASS	COntext and Motion Primitive Aggregate Surgical Set
CPS	Cyber-Physical System
DCS	Data Collection System
DESK	DExterous Surgical SKills
DTW	Dynamic Time Warping
dV	da Vinci
dVRK	da Vinci Research Kit
dVSS	da Vinci Surgical System
FSM	Finite State Machine
GED	Graph Edit Distance
GES	Graph Edit Score
GRS	Global Rating Scale
HMM	Hidden Markov Model
IBIS	Independent Binocular Imaging System
IOU	Intersection Over Union
JIGSAWS	JHU-ISI Gesture and Skill Assessment Working Set
KL	Kullback-Liebler
KT	Knot Tying
LSTM	Long-Short Term Memory
MAE	Mean Absolute Error
mAP	Mean Average Precision
MAPE	Mean Absolute Percentage Error
MIS	Minimally Invasive Surgery
MP	Motion Primitive
MRCNN	Mask Region-based CNN

NP	Needle Passing
LOSO	Leave-One-Supertrial-Out
LOTO	Leave-One-Task-Out
LOUO	Leave-One-User-Out
LSTM	Long-Short Term Memory
OSATS	Objective Structured Assessments of Technical Skills
PaS	Post and Sleeve
PoaP	Pea on a Peg
PT	Peg Transfer
RAS	Robot-Assisted Surgery
RMSE	Root Mean Squared Error
ROI	Region of Interest
ROSMA	RObotic Surgical MAnuevers
S	Suturing
SPM	Surgical Process Model
TCN	Temporal Convolutional Network

Chapter 1

Introduction

1.1 Motivation

Surgical robots provide surgeons with increased flexibility and precision while reducing incision size, recovery time, and scarring for patients. They are driving the creation of new surgical techniques and technologies [1] and are now used in many surgical procedures across multiple specialties including urology, gynecology, and general surgery. Since their adoption, there has been an increase in the number of minimally invasive surgery (MIS) cases [2], and the da Vinci Surgical Systems by Intuitive Surgical [3] have been used to perform over 10 million procedures [4] as of 2021. Surgical robots are complex, human-in-the-loop cyber physical systems (CPS) [5] where a human surgeon with knowledge of the surgical workflow sits at the master console and operates the surgical robot. Commands from the control software in the cyber layer are sent to the surgical robot in the physical layer.

Surgical robots are transforming the way surgeons operate, and enabling the development and integration of features to make surgical robots dependable, safe, secure, and efficient [5]. Advancements in sensing and computing technology support the collection of system logs, kinematic data, and videos from robot-assisted surgery (RAS) systems and simulators during procedures. This has given rise to the field of surgical data science [6]. Surgical data science aims to increase the accessibility and quality of medical data, and leverage it to assist care providers, improve patient care, augment surgeon training, and enable the development of new surgical technologies. As shown in Figure 1.1, surgical data science is central to the analysis of data for surgical process modeling, surgical scene segmentation, activity recognition, surgical autonomy, and automated skill assessment.

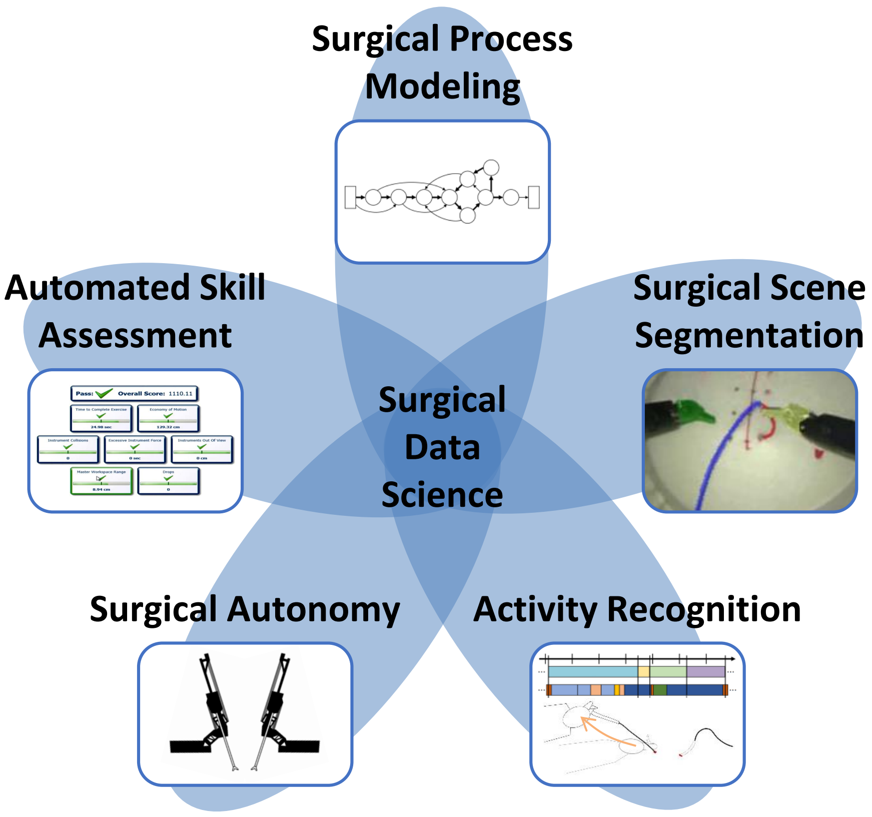

Surgical Process Modeling
Automated Skill Assessment
Surgical Scene Segmentation
Surgical Data Science
Surgical Autonomy
Activity Recognition

Figure 1.1: Surgical data science is central to the modeling and analysis of medical data for augmenting surgeon training, improving patient care, and enabling the development of new technologies.

As part of their training, surgeons perform surgical tasks using the surgical robot in both real dry lab and virtual simulated environments. These tasks allow surgeons to hone their skills on the surgical robot and practice the motions and gestures that are the building blocks for real surgical operations. Their performance on these tasks is manually or automatically assessed by a senior surgeon or training simulator to track skill development, provide feedback, and determine skill level. However, manual assessment, using standardized rubrics such as the Global Rating Scale (GRS) score [7], can be subjective and time consuming.

Automated skill assessment is being developed for more objective feedback and to reduce the burden on expert surgeons, but these methods are based on metrics such as time and path length, or data-driven models which lack specificity and interpretability. In addition, less attention has been paid to safety metrics and the analysis of errors which is an emerging subfield of research [8]. There has also been more interest in fine-grained and interpretable feedback during training based on the decomposition of surgical activities into smaller components for analysis by leveraging surgical process modeling. Surgical process modeling [9, 10] seeks to model surgical workflow by defining different levels of a surgical hierarchy and decomposing surgical procedures into smaller components such as tasks, gestures, and actions. Machine learning models for the automated recognition of these surgical activities are a fundamental component of systems for skill assessment, autonomy, and error detection. However, comprehensive and comparative analyses are hindered by the lack of a generalizable framework for modeling surgical tasks and the absence of annotated datasets that reflect the diversity of tasks in real surgery.

1.2 Challenges

This dissertation focuses on addressing three challenges in robotic surgery including limited datasets with incompatible activity labels, limited explainability and generalizability in activity recognition, and limited attention to executional errors in the surgical tasks during training and skill assessment.

1.2.1 Surgical Context and Activity Modeling

The formal specification of surgical interventions is part of planning, teaching, and evaluating surgical operations [10] and can involve top-down knowledge representation and bottom-up data-driven techniques [11]. These models are applied to datasets with real or dry lab tasks to label surgical activity at different granularities such as phases, tasks, gestures, and actions. However, a major limitation is that datasets contain only a few tasks, subjects, and trials resulting in low data diversity.

Much work has been done at the gesture level in skill assessment [12, 13], autonomy [14], and error detection [15–18] using these datasets, and they have also been the subject of activity recognition and image segmentation challenges [19–21]. The automated segmentation and classification of gestures is critical for these applications and an active field of research. However, the definitions of gestures vary between datasets which limits comparative analyses between datasets and models [8]. Surgical actions and action triplets [22,23] have also been introduced for the fine-grained description of surgical activity, but the number and types of actions used to label datasets still vary, and these datasets focus on video data of real surgeries. The absence of a standardized, multilevel framework for defining and labeling fine-grained surgical activities continues to be a challenge for surgical robotics research [8, 10]. These challenges are compounded by the difficulty of annotating for surgical workflow [24], the presence of errors in gesture labels [25], and differing gesture definitions among existing works.

1.2.2 Surgical Context Inference and Activity Recognition

Understanding the complex surgical environment is an important first step in augmenting surgical robots with systems to improve training, safety, and efficiency [26]. Towards these applications, it is important to know the tools, objects, and anatomical structures in the scene; the current step and task; and the activities the surgeon is performing to make progress in the operation. This is accomplished by analyzing video and kinematic data from the surgical robot using various data processing and machine learning methods. However, interpretability remains a challenge in deep learning models [27].

Previous works on surgical robot instrument or surgical scene segmentation used publicly available datasets of real surgical videos such as MICCAI EndoVis 17 [28], MICCAI EndoVis 18 [19], and Cata7 [29]. However, they did not explicitly relate the segmentation and interactions of tools and objects to the activities performed by the surgeon. Relating these tool and object interactions to surgical activity would provide a deeper understanding of the surgical environment as well as an alternative method of verifying activity labels. The complexity and number of interactions during individual gestures has been a challenge to directly relating segmented images to surgical workflow. In addition, video and/or kinematic data has also been used for surgical gesture recognition as summarized in [8]. Kinematic data is critical for safety and error detection applications, since occlusions and lens contamination are common [17, 30, 31], but is not always available in datasets. Furthermore, the limited number of tasks in each dataset and lack of standardized labels is a significant challenge to aggregating data for comparative analysis of models and evaluating their generalizability. Evaluating the generalizability of activity recognition models is particularly important to ensure that they can produce reliable results on new trials, subjects, and *tasks* that they have not seen during training. Previous works have focused on generalizability to unseen subjects or trials, but there has not yet been an assessment of the task-generalizability of activity recognition models.

1.2.3 Surgical Error Analysis

It is generally expected that a surgeon's skill will improve with experience, and studies of the learning curve for robotic surgical procedures found that between 30 and 175 operations are needed to "significantly reduce the overall rate of postoperative complications" [32]. There have been significant efforts towards developing standardized and validated training programs to help surgeons achieve the high level of experience necessary for safe operations [32]. During their training, surgeons are periodically evaluated by automated assessments from training simulators and manual assessments from expert surgeons. However, existing automated skill evaluation methods have limited interpretability, and manual methods are time-consuming and subjective. Many previous works have developed data-driven methods for skill assessment using statistical measures and machine learning models. However, there has been limited attention to identifying executional errors made by surgeons during training and skill assessment. These are important because a review of adverse event reports in the FDA's MAUDE (Manufacturer and User Facility Device Experience) database for the da Vinci Surgical System found that 81% of adverse events relating to pre-existing health conditions or surgeon errors resulted in injury or death [33].

However, error detection is still an emerging subfield [8] with significant challenges. The rarity of errors has led to a lack of datasets that are labeled for errors, and resulted in limited attention to developing methods for identifying and analyzing errors. Previous works that do consider errors have done so in the context of activity-aware skill assessment where the errors are associated with a particular fine-grained unit of surgical activity such as gestures or motions. Several works have found that some gestures are more indicative of skill than others [13, 34–36] and that there is a correlation between skill level and errors [15]. For example, [37] found that experts used fewer gestures, made fewer errors, and had more predictable transitions between gestures. Thus, methods for identifying and analyzing executional errors can improve the safety of robotic surgery and support surgeon training and skill assessment.

Monitoring systems that are aware of the tools, objects, and tissues in the surgical scene, the surgical process, and the activities the surgeon is performing can improve robotic surgery by providing interpretable feedback about skill and safety. Such safety monitoring systems would need to differentiate between deviations from the expected procedure due to individual surgeon preferences or patient anatomy, and those that are due to errors. A high false positive rate for error detection could lead to surgeons ignoring the alerts given by the safety monitoring system, but not detecting and correcting errors could be harmful to the patient. Furthermore, errors contribute to suboptimal performance and potentially safety-critical events, so they can affect the surgical process and workflow [33]. However, previous fine-grained surgical activity analysis has focused on statistical measures such as time and path length [34, 38], or used data-driven models such as Hidden Markov Models (HMMs) [12, 13] which are not specific or easily interpretable. On the other hand, works that aim to provide interpretable feedback have used metrics as inputs [39, 40], or techniques to visualize trajectories [41] or features [42], but have not considered patterns in fine-grained surgical activities.

1.3 Contributions

This dissertation addresses the above challenges by developing and evaluating methods for fine-grained activity modeling, recognition, and error analysis in robot-assisted surgery, organized around three research thrusts as shown in Figure 1.2.

First, we address the challenge of limited datasets with incompatible activity labels by proposing a formal framework for the fine-grained modeling of surgical tasks using context and motion primitives. This framework directly relates interactions between tools and objects (encoded as context) to surgical workflow (described by motion primitives) and enables the modeling of tasks as finite state machines. We also develop a method for labeling context that results in objective, fine-grained annotations with near-perfect agreement among annotators and expert surgeons. We apply our method to three publicly available multimodal datasets with both kinematic and video data to create the COntext and Motion Primitive Aggregate Surgical Set (COMPASS) with nearly three times as much data compared to JIGSAWS alone [43].

Second, we use this framework and dataset to develop models for surgical context inference and activity recognition with improved explainability and generalizability. We leverage image segmentation models for inferring context from surgical videos to improve the explainability of gesture recognition models. With the aggregated data from a variety of tasks in COMPASS, we propose a novel cross validation setup that evaluates the generalizability of activity recognition models to unseen tasks. We find that finer-grained models exhibit greater generalizability across tasks which motivates us to study the execution of lower-level surgical activities in these tasks.

Third, we use the surgical task and activity models developed in the previous thrusts to create activity-aware methods for the identification and analysis of errors. We use these novel methods to identify interpretable patterns in fine-grained surgical activities that strongly correlate with measures of surgical expertise and can be used to provide specific and interpretable feedback to surgeons during training and skill assessment.

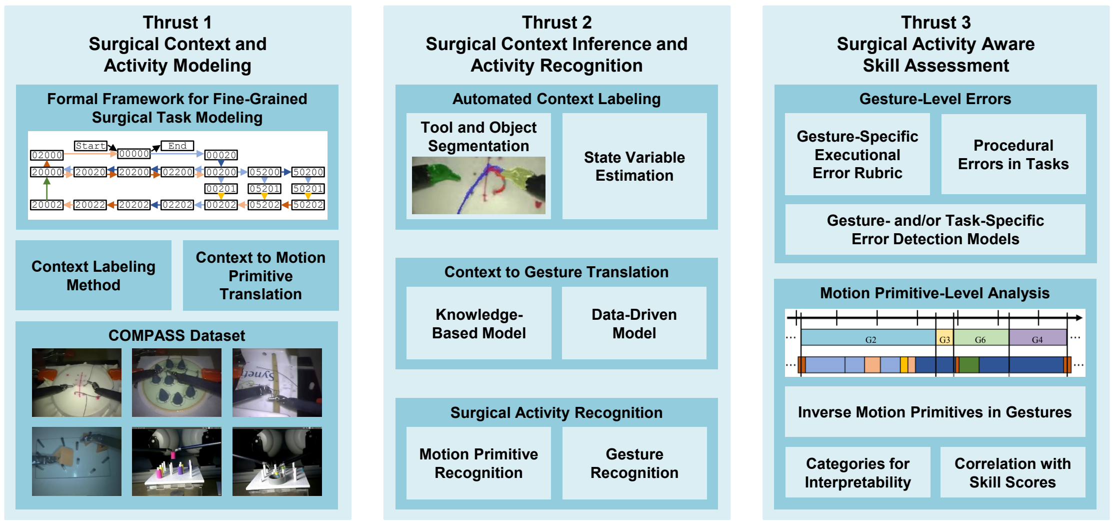

Thrust 1
Surgical Context and Activity Modeling
Formal Framework for Fine-Grained Surgical Task Modeling
Context Labeling Method
Context to Motion Primitive Translation
COMPASS Dataset
Thrust 2
Surgical Context Inference and Activity Recognition
Automated Context Labeling
Tool and Object Segmentation
State Variable Estimation
Context to Gesture Translation
Knowledge-Based Model
Data-Driven Model
Surgical Activity Recognition
Motion Primitive Recognition
Gesture Recognition
Thrust 3
Surgical Activity Aware Skill Assessment
Gesture-Level Errors
Gesture-Specific Executional Error Rubric
Procedural Errors in Tasks
Gesture- and/or Task-Specific Error Detection Models
Motion Primitive-Level Analysis
Inverse Motion Primitives in Gestures
Categories for Interpretability
Correlation with Skill Scores

Figure 1.2: Research thrusts for developing and evaluating methods for fine-grained activity modeling, recognition, and error analysis in robot-assisted surgery.

This work makes the following contributions:

- **Definition and labeling method for surgical context:** We formally define surgical context as a set of important state variables that represent the interactions between surgical instruments and objects. Context is a combination of general- and task-specific state variables and can be manually labeled using video data. We show that this definition can be applied to different surgical tasks by using it to label six standard dry lab training tasks. Our method for annotating surgical context results in a set of high-quality labels with near-perfect agreement between consensus labels from non-experts and labels from expert surgeons, as well as higher agreement among annotators than existing methods for labeling gestures with descriptive definitions.

- **Modeling of surgical tasks with motion primitives enabling dataset aggregation:** We propose a formal framework for modeling surgical tasks with motion primitives, as atomic units of surgical activity, whose performance results in changes in surgical context. This enables the modeling of tasks as finite state machines using context and motion primitives. Using our methods for labeling context and the automated translation of context to motion primitives, we label six dry lab tasks from three publicly available datasets (JIGSAWS [43], DESK [44], and ROSMA [45]) to create an aggregate dataset called COMPASS (COntext and Motion Primitive Aggregate Surgical Set) that contains both kinematic and video data along with standardized context and motion primitive labels. This nearly triples the amount of data available for analysis and comparison, and for training machine learning models.

- **Automated inference of surgical context based on image segmentation:** We propose an algorithm for the detection of surgical states using image segmentation that results in the automated inference of surgical context based on only video data. This leverages improvements in surgical tool and object segmentation to support surgical workflow annotation.

- **Methods for translating surgical context to gestures:** We design two models for translating surgical context to existing gesture definitions using a knowledge-based finite state machine and a data-driven machine learning model. This enables the comparison of gestures derived from context to the gesture labels from the JIGSAWS dataset and helps bridge the gap between image segmentation and surgical workflow annotation.

- **Evaluation of the task-generalizability of activity recognition models:** We propose the Leave-One-Task-Out (LOTO) cross validation method to assess the generalizabilty of a surgical activity recognition model to an unseen task since models will see new tasks and subjects when deployed. This enables us to perform the first evaluation of a surgical activity recognition model on data from multiple tasks and datasets, and show that our framework enables comparisons between models and datasets. We show preliminary results that dataset aggregation can enhance the task-generalization of motion primitive recognition models.

- **Rubric for the identification of gesture-specific errors:** We create a gesture-specific rubric for identifying executional and procedural errors in robotic surgery and apply it to the Suturing and Needle Passing tasks from the JIGSAWS dataset to obtain a set of error labels that support error analysis and detection. We find that gestures have different predominant error modes and some kinematic parameters can be better indicators of executional error occurrences. In addition, the total number of executional errors correlates with longer trial durations and lower skill.

- **Identification of inverse motion primitives towards interpretable skill assessment:** We analyze surgical gestures and motion primitives and the relationship between them towards improving the interpretability of skill assessment. We identify inverse motion primitives that are often used as recovery actions to correct the position or orientation of objects, or may be indicators of other issues such as with depth perception. Their occurrence and total duration are strongly correlated with lower GRS scores (provided with the JIGSAWS dataset) and can be used to identify the portions of trials containing specific motions that contribute to lower scores.

1.4 Dissertation Organization

This dissertation is organized as follows. Chapter 2 presents a novel formal framework to describe the surgical scene using context and models of surgical tasks using a set of standardized motion primitives. Chapter 3 evaluates models for the recognition of surgical context, motion primitives, and gestures. Chapter 4 proposes methods for identifying and analyzing gesture-level errors and motion-level patterns in dry lab surgical tasks. Chapter 5 gives the conclusion and directions for future research.

Chapter 2

Surgical Context and Activity Modeling

2.1 Background

Surgical process models (SPMs) [9, 10] decompose surgical operations into smaller units of activity such as steps, tasks, and gestures as shown in Figure 2.1. While modeling procedures with phases and steps enables standardization and supports teaching [47], finer-grained modeling is needed for automated assistance, skill assessment, and autonomy in robot-assisted surgery. Gestures, defined as intentional activities with meaningful outcomes [43], are an important analytical unit for skill evaluation [8, 48] and error detection [15]. Finer-grained activities called actions or motions [49–51] have also been proposed to improve the understanding of tool-tissue interactions and their relationship to different granularity levels. The recognition of motions can increase the explainability of gesture-level analyses, improve error detection by identifying the exact erroneous parts of gestures [40–42, 48], and enable autonomy or error recovery through the execution of motion primitives [11, 14].

The decomposition of tasks into gestures has been done with models such as graphs [37, 52], statecharts [11], hybrid automata [53], and behavior trees [54] for

This chapter contains material from [46] coauthored with I. Reyes, Z. Li, and H. Alemzadeh. Reproduced with permission from Springer Nature.

cooperative, autonomous, and supervisory systems in robotic surgery [55]. However, prior work has not explicitly modeled or formalized surgical context, which is characterized by the status and interactions among surgical tools and tissues/objects, and its relationship to motions, gestures, and surgical workflow.

Additionally, while research has focused on standardized surgical ontologies [56], labeling methods [57], and action triplets [49], there is still a need for a common surgical language and larger multimodal datasets to support comparative analysis of activity recognition and error detection models [8]. Section 2.6 provides a detailed summary of related work on gesture and action definitions and datasets. As shown in Tables 2.14 and 2.15, the definitions, numbers, and types of activity labels vary in the existing datasets. The most commonly used dataset is JIGSAWS [43], which contains kinematic data, videos, gesture labels, and surgical skill scores for three dry lab surgical tasks. However, only two of its tasks are labeled with similar sets of gestures. Other recently developed datasets such as DESK [58] and V-RASTED [59] have defined their own sets of gestures while ROSMA [45] is not labeled. Datasets such as these with both kinematic and video data from a surgical robot or simulator are small and contain only a handful of trials of a few simulated or dry lab training tasks performed by a limited number of subjects leading to low data diversity. This scarcity of data hinders the training and generalization of machine learning models. Also, most datasets on finer-grained actions have focused on only video data from real surgery. While video data is required for labeling, the inclusion of kinematic data is valuable for safety analysis [16–18], improved recognition accuracy through multimodal analysis [60, 61], or when video data is not available or is noisy due to smoke or occlusions [30] or lens contamination.

Furthermore, annotating surgical workflow is costly and requires guidance from expert surgeons [24], and the resulting labels may contain errors and inconsistencies such as those identified by [25] in JIGSAWS. Label quality and inter-rater agreement have not been examined when creating the existing datasets. Also, the labels in these datasets do not capture multiple levels of granularity which can be useful for detecting errors and preventing their propagation to higher levels of the surgical hierarchy. In addition, they do not differentiate between activities performed by the left and right hands, which is important for detailed skill assessment and the analysis of bimanual coordination [62].

We address these challenges by making the following contributions:

- We propose a novel formal framework for modeling surgical dry lab tasks with finite state machines using a standardized set of motion primitives whose execution leads to changes in important state variables that make up the surgical context. Context characterizes the physical interactions among surgical tools and objects, and motion primitives represent basic surgical actions across different surgical tasks and procedures.

- We develop a method for labeling surgical context based on the video data of dry lab tasks that achieves near-perfect agreement between crowdsourced labels and expert surgeon labels, higher agreement among annotators than using existing gesture definitions, and such that the context labels can be automatically translated into motion primitive labels.

- We apply our framework and labeling method to create an aggregate dataset, called COMPASS (COntext and Motion Primitive Aggregate Surgical Set), which includes kinematic and video data as well as context and motion primitive labels for a total of six dry lab tasks from the JIGSAWS [43], DESK [58], and ROSMA [45] datasets.

The tools for labeling surgical context based on video data, automated translation of context to motion primitive labels, and the aggregated dataset with context and motion primitive labels are publicly available at `https://github.com/UVA-DSA/COMPASS`.

This chapter is organized as follows. Section 2.2 introduces the surgical hierarchy, defines context and the surgical activities at each level, and describes the tasks that will be studied. Section 2.3 describes the context and motion primitive models for the tasks and the translation of context to motion primitives. Section 2.4

presents the method for labeling surgical context and the evaluation of the method for labeling and translating context to motion primitives. Section 2.5 describes the standardized features of the COMPASS dataset. Section 2.6 reviews related work on datasets and recognition models for gestures and actions. Section 2.7 gives the conclusion for this chapter.

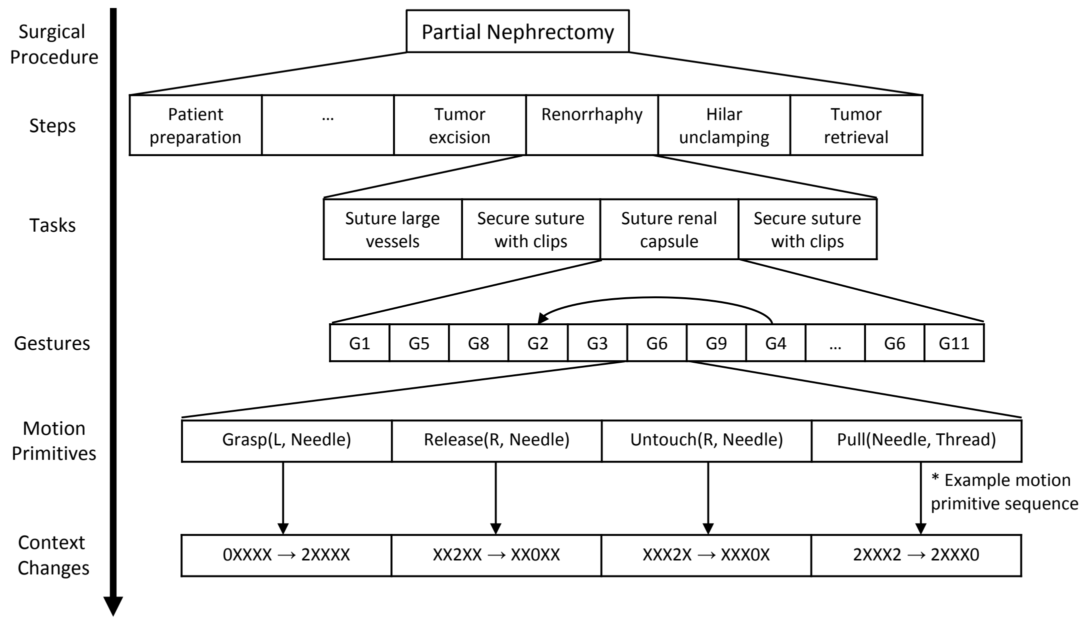


Figure 2.1: Surgical hierarchy showing the decomposition of a surgical procedure into steps, tasks, and gestures. Our proposed framework adds the motion primitive and context levels that relate surgical workflow to physical interactions in the surgical environment which can be identified in video data.

2.2 Surgical Hierarchy and Activity Definitions

Surgical procedures follow the hierarchy of levels defined in [10] which relate the fine-grained activities a surgeon performs to objects and progress in the operation as shown in Figure 2.1. In this dissertation, surgical **activities** will refer in general to units at any level of the surgical hierarchy, but more specifically to both the gesture and motion levels. When a patient undergoes a surgical **operation**, they may have one or more **procedures** performed which are divided into **steps**. Each step (or phase) results in the completion of a major objective during the surgery such as the removal of a tumor. This is accomplished by performing a series of **tasks** which achieve smaller goals such as suturing. These tasks are comprised of **gestures** (also called sub-tasks or surgemes) as shown in grammar graphs [52] which are defined as units of surgical activity with a specific intent [43] with semantic meaning of both the activity and underlying physical context. Below this level, **motion primitives** (also called motions, actions, or dexemes) are atomic surgical activities involving only one hand and without the semantics of physical context [9, 10] such as moving an instrument or closing the graspers. These motion primitives effect changes in important states that define the overall surgical **context** which describes the interactions between surgical tools, objects, and anatomical structures. Further, motion primitives can be decomposed into the basic actuations of translate, rotate, and open/close graspers that represent commands sent to the surgical robot.

Our new representation of surgical tasks extends surgical process modeling to include context and motion primitives which is missing from previous work where lower-level activities beneath gestures are obscured by hidden states such as in [52].

2.2.1 Context

A surgical environment (in either dry lab or real surgical procedures) can be modeled by a set of state variables that characterize the status and interactions among surgical instruments (e.g., graspers, scissors, and electrocautery tools) and objects (e.g., needles, threads, blocks, balls, sleeves, and rings) or anatomical structures (e.g., organs, tissues, and tumors) at a given time in the physical environment. We define surgical context using two sets of variables that can be observed or measured using kinematic and/or video data from a surgical scene: (i) general state variables relating to the contact and hold interactions between the tools and objects in the environment, and (ii) task-specific state variables describing the states of objects critical to the current task. We define independent state variables for the left and right tools to enable the generation of separate label sets for each side to support side-specific skill assessment, hand coordination analysis, and improved motion primitive recognition.

We propose the following definition of surgical context with general and task-specific state variables. The four general state variables are: Left Holding, Left Contact, Right Holding, and Right Contact where the value of the state variable represents the interacting object as shown in Figure 2.2. An additional task-specific state variable is appended to the right of the general state variables to describe progress in the task. The Suturing and Needle Passing tasks involve throwing four sutures and passing a needle through four rings, respectively, so the needle, if held, can be "not touching", "touching", or "in" the fabric or ring. The Knot Tying task involves tying two knots, so while a knot is being tied, the thread can be "wrapped" around the opposite grasper, in a "loose" knot, or in a "tight" knot. The Peg Transfer and Post and Sleeve tasks involve picking up a block and placing it on another post, so the block can be "on" or "off" the peg. The Pea on a Peg task involves picking up a pea and placing it on a post, so the pea, if held, can be "in the cup", "stuck to other peas", "not stuck to other peas", or "on the peg". For example in Figure 2.2, the state 50202 indicates that the left grasper is holding a ring, the right grasper is holding the needle, and the needle is in the ring.

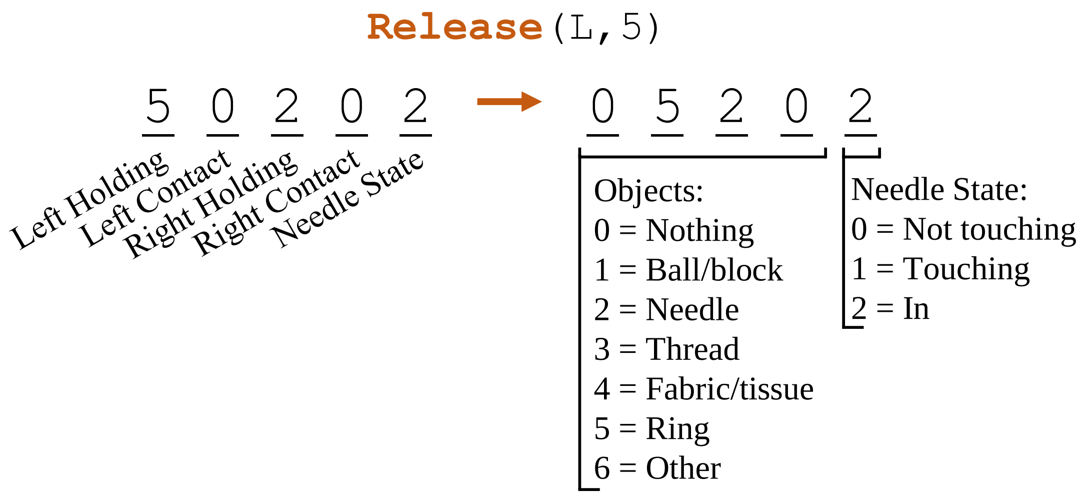


Figure 2.2: Object and needle state encodings for the state variables of the context in the Needle Passing task.

Changes in the surgical context happen as the result of performing a set of basic motion primitives with the robot (either controlled by the surgeon or autonomously). So, each surgical task can be modeled as a finite state machine (FSM) with the states representing the surgical context and the transitions representing the motion primitives as presented in Section 2.3.2. We focus on dry lab training tasks and manually construct FSMs to model each task after reviewing the videos to understand the general activities in the task. In these models, states represent context and transitions represent MPs. This representation of surgical tasks incorporates surgical context into procedure modeling which is missing from previously proposed models such as grammar graphs and Hidden Markov Models [52] where hidden states obscure lower-level actions.

2.2.2 Motion Primitives (MPs)

We define a unified set of six modular and programmable surgical motion primitives (MPs) to model the basic surgical activities that lead to changes in the physical context and thus relate surgical context to workflow. As shown in Equation 2.1, each MP is characterized by its type (e.g., Grasp), the specific tool which is used (e.g., left grasper), the object with which the tool interacts (e.g., block), and a set of constraints that define the functional (e.g., differential equations characterizing typical trajectories [14]) and safety requirements (e.g., virtual fixtures and no-go zones [17, 63, 64]) for the execution of MPs:

$$\text{MP(tool, object, constraints)} \tag{2.1}$$

The performance of an MP results in changes in the state variables of the surgical context. Table 2.1 shows the set of MPs and corresponding changes to surgical context applicable to all tasks. Table 2.2 shows the set of MPs and corresponding changes to surgical context applied to specific dry lab tasks.

In this framework, tools and objects are considered classes as in object-oriented programming and can have attributes such as the specific type of tool and current

Table 2.1: General motion primitives for changes in context: “L” and “R” represent the left and right graspers as tools, “a” is a generic object as listed in Figure 2.2, and “X” can be any value.

Motion Primitive	Context Change
Touch(L, a)	X0XX $\rightarrow$ XaXX
Touch(R, a)	XXX0 $\rightarrow$ XXXa
Grasp(L, a)	0aXX $\rightarrow$ aXXX
Grasp(R, a)	XX0a $\rightarrow$ XXaX
Release(L, a)	aXXX $\rightarrow$ 0aXX
Release(R, a)	XXaX $\rightarrow$ XX0a
Untouch(L, a)	XaXX $\rightarrow$ X0XX
Untouch(R, a)	XXXa $\rightarrow$ XXX0

position. The MPs can be further decomposed into the fundamental transformations of move/translate, rotate, and open/close grasper that characterize the low-level kinematic commands. These can be used for the programming and execution of motions on a robot for semi-autonomous surgery [11], which is the subject of future work.

The segmentation of tasks into MPs allows the separation of activities performed by the left and right hands and the generation of separate sets of labels. This can support more detailed skill assessment, the analysis of bimanual coordination [62], and surgical automation [14]. To generate separate left and right label sets, MPs performed by each hand or arm of the robot are split into new transcripts. MPs performed with a held object such as a needle are assigned to the transcript of the arm holding the object. Then, the Idle MP is used to fill in the gaps created by the separation so that every kinematic sample and video frame has a label and the labels are continuous.

MPs are similar to the action triplets proposed by [49] for surgical action recognition based on video data in real surgical tasks. But, we focus on developing a standardized framework and applying it to a variety of tasks and datasets with *both kinematic and video data* to enable comparative analyses between datasets and tasks. Kinematic data can support analysis for safety and skill evaluation, and be used to develop dynamic motion primitives (DMPs) for automating surgery [14]. Context and MPs can be extended to new tasks and real surgical procedures by defining task-specific state variables for the context and adding task- or tool-specific verbs perhaps similar to those proposed in [49] (e.g., Cut for scissors) that change the value of those state variables and result in progress in the task.

Table 2.2: Task-specific motion primitives for changes in context: “L” and “R” represent the left and right graspers as tools, objects are encoded as in Figure 2.2, “b” is a value greater than 0, and “X” can be any value.

Motion Primitive	Context Change
Suturing/Needle Passing	
Touch(2, 4/5)	2XXX0 → 2XXX1
Touch(2, 4/5)	XX2X0 → XX2X1
Push(2, 4/5)	2XXX1 → 2XXX2
Push(2, 4/5)	XX2X1 → XX2X2
Push(2, 4/5)	XXXX0 → XXXX2
Pull(L, 2)	2XXX2 → 2XXX0
Pull(R, 2)	XX2X2 → XX2X0
Pull(2, 4/5)	XXXX2 → XXXX1
Knot Tying	
Pull(L, 3)	3XXX0 ↔ 3XXX1
Pull(R, 3)	XX3X0 ↔ XX3X1
Pull(L, 3) Pull(R, 3)	3X3X1 → 3X3X2
Pull(L, 3) Pull(R, 3)	3X3X2 → 3X3X3
Peg Transfer and Post and Sleeve	
Untouch(1, Post)	XXXX0 → XXXX1
Touch(1, Post)	XXXX1 → XXXX0
Pea on a Peg	
Grasp(L, 1)	0XXX0 → 1XXX1
Grasp(R, 1)	XX0X0 → XX1X1
Pull(L, 1)	1XXX1 → 1XXX2
Pull(R, 1)	XX1X1 → XX1X2
Pull(L, 1)	1XXX1 → 1XXX3
Pull(R, 1)	XX1X1 → XX1X3
Touch(1, 1)	XXXX3 → XXXX2
Untouch(1, 1)	XXXX2 → XXXX3
Touch(1, Peg)	XXXX3 → XXXX4
Untouch(1, Peg)	XXXX4 → XXXX3
Release(L, 1)	1XXXb → 0XXX0
Release(R, 1)	XX1Xb → XX0X0
Push(L, 1)	1XXX2 → 1XXX1
Push(R, 1)	XX1X2 → XX1X1

2.2.3 Gestures

Surgical gestures are defined as modular and recognizable units of surgical activity that result in a meaningful outcome [43, 44]. This dissertation does not propose new gestures, but instead relates context and motion primitives to the existing definitions from JIGSAWS since it is the most commonly used dataset for gesture-related research. Section 2.6 presents a detailed summary of related work on gesture and action definitions and datasets including and beyond JIGSAWS. The set of 15 gestures defined by [43] and used to label surgical activity in the Suturing, Needle Passing, and Knot Tying tasks of the JIGSAWS dataset are listed in Table 2.3.

Table 2.3: JIGSAWS gesture indices and definitions from [43].

Index	Description
G1	Reaching for needle with right hand
G2	Positioning needle
G3	Pushing needle through tissue
G4	Transferring needle from left to right
G5	Moving to center with needle in grip
G6	Pulling suture with left hand
G7	Pulling suture with right hand
G8	Orienting needle
G9	Using right hand to help tighten suture
G10	Loosening more suture
G11	Dropping suture at end and moving to end points
G12	Reaching for needle with left hand
G13	Making C loop around right hand
G14	Reaching for suture with right hand
G15	Pulling suture with both hands

2.2.4 Tasks

The performance of a task achieves a goal in a surgical step, and the exercises on robotic simulators and in dry lab settings are common examples of this unit of activity. By practicing these tasks, surgeons improve their manipulation of the robot's instruments and learn the basic components of surgical operations such as object manipulation, suturing, knot tying, and electrocautery. In this dissertation, we use six dry lab tasks from the JIGSAWS [43], DESK [44], and ROSMA [45] datasets and describe each task below.

Suturing

Suturing is one of the most common tasks surgeons perform. An example of suturing as a dry lab exercise is shown in Figure 2.3 from the JIGSAWS dataset. In this task, the surgeon retrieves the needle and throws four sutures with the entry and exit points for the needle marked with red dots on the fabric. We abbreviate this task as "S".

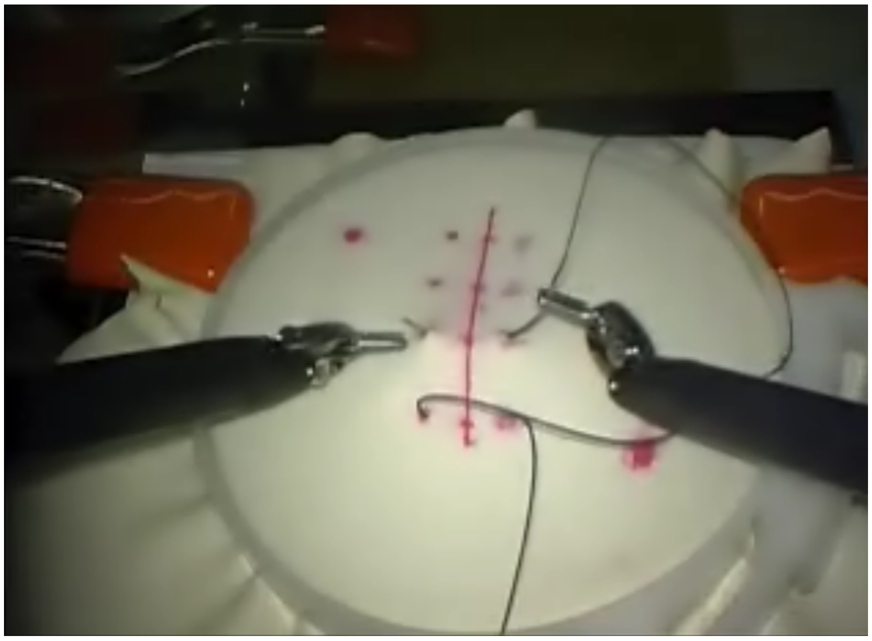


Figure 2.3: Example of the Suturing (S) task from the JIGSAWS dataset [43].

Needle Passing

The Needle Passing task of the JIGSAWS dataset is similar to the Suturing task in that it involves manipulating a needle. The dry lab setup for this task is shown in Figure 2.4 and in this task, the surgeon threads the needle through four of the rings. We abbreviate this task as “NP”.

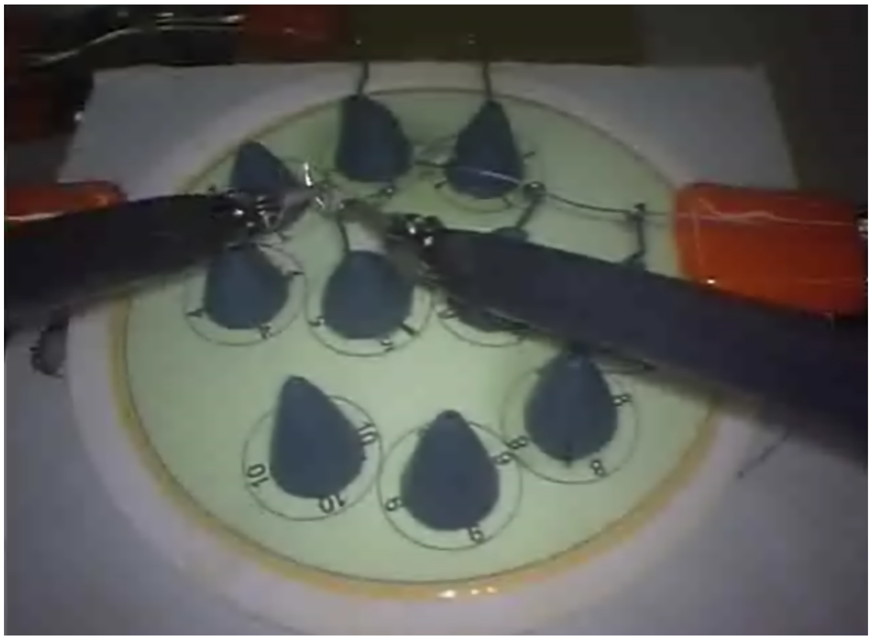


Figure 2.4: Example of the Needle Passing (NP) task from the JIGSAWS dataset [43].

Knot Tying

Knot Tying is another basic skill that surgeons must master. In this task, the surgeon ties two knots with the suture around the rubber tubing as shown in Figure 2.5 from the JIGSAWS dataset. Each knot is tied by grasping the right end of the suture with the left grasper, wrapping the suture around the jaws of the right grasper, grabbing the left end of the suture with the right grasper and pulling it through the loop to create a knot. We abbreviate this task as "KT".

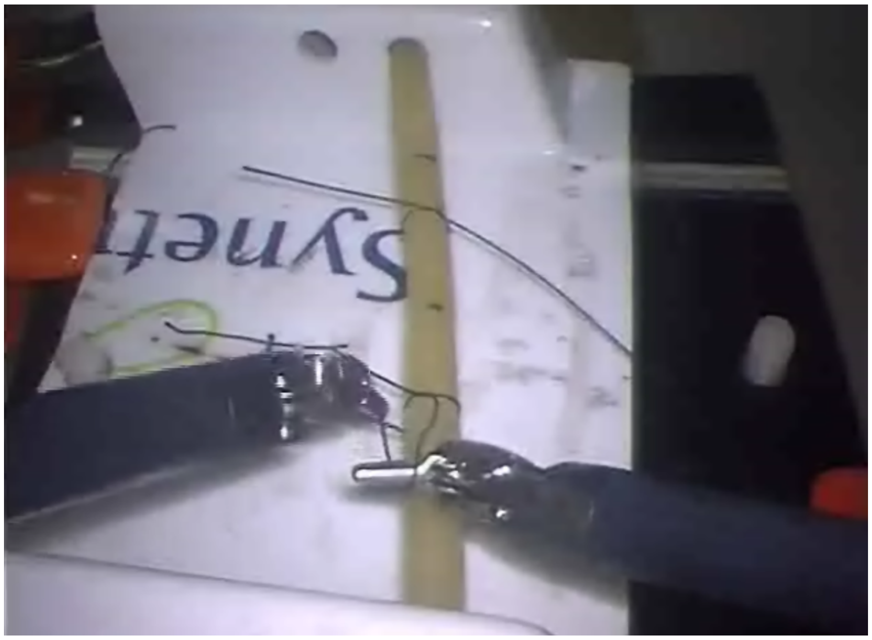


Figure 2.5: Example of the Knot Tying (KT) task from the JIGSAWS dataset [43].

Peg Transfer

Peg Transfer from the DESK dataset is a basic object manipulation task using rubber blocks and a set of vertical pegs as shown in Figure 2.6. In this task, each of the three blocks is grasped and lifted from its peg, passed to the other instrument, and placed on a different peg. We abbreviate this task as “PT”.

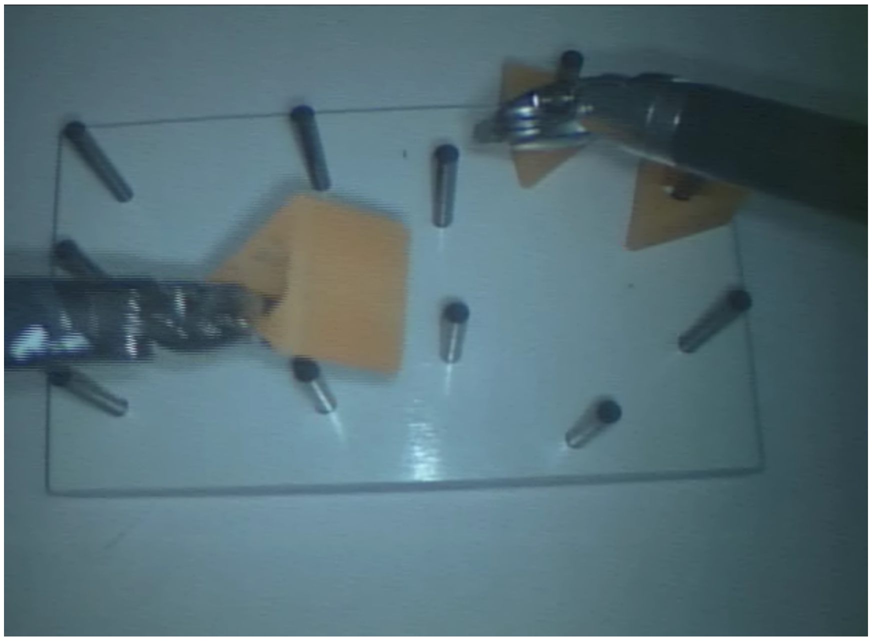


Figure 2.6: Example of the Peg Transfer (PT) task from the DESK dataset [44].

Post and Sleeve

The Post and Sleeve task of the ROSMA dataset is similar to the Peg Transfer task but uses sleeves and posts instead of blocks and pegs as shown in Figure 2.7. Tasks like this and Peg Transfer are also called pick-and-place tasks since they involve the picking up, transfer, and placing of objects. In the Post and Sleeve task, each sleeve is grasped, lifted from its post, passed to the other instrument, and placed on a different post. We abbreviate this task as “PaS”.

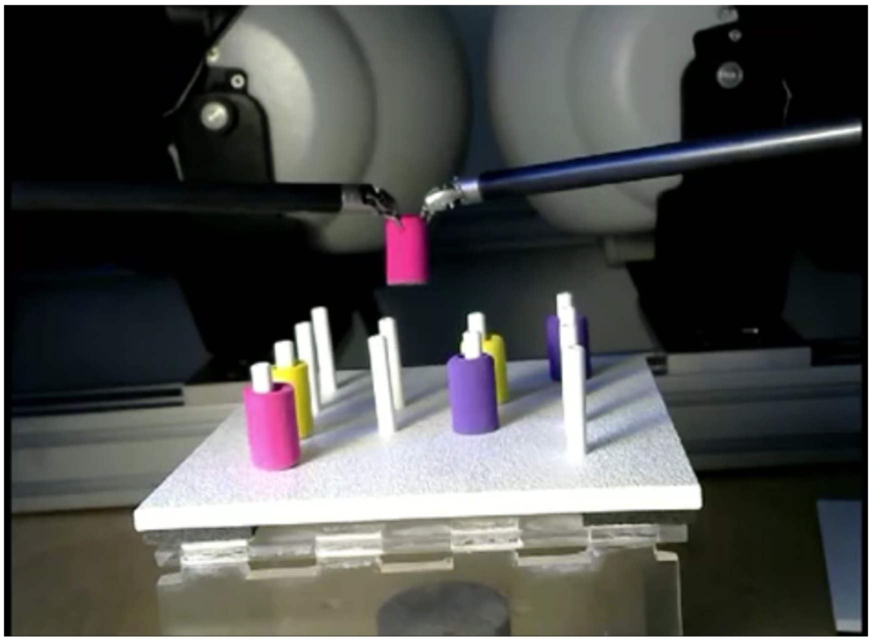


Figure 2.7: Example of the Post and Sleeve (PaS) task from the ROSMA dataset [45].

Pea on a Peg

Pea on a Peg from the ROSMA dataset is another pick-and-place task where fuzzy balls called peas are contained in a cup surrounded by pegs as shown in Figure 2.8. In this task, the surgeon picks up a pea from the cup and places it on one of the pegs. This task is challenging since the peas stick together and must be separated. Also, careless maneuvers may knock previously placed peas off of their pegs. We abbreviate this task as “PoaP”.

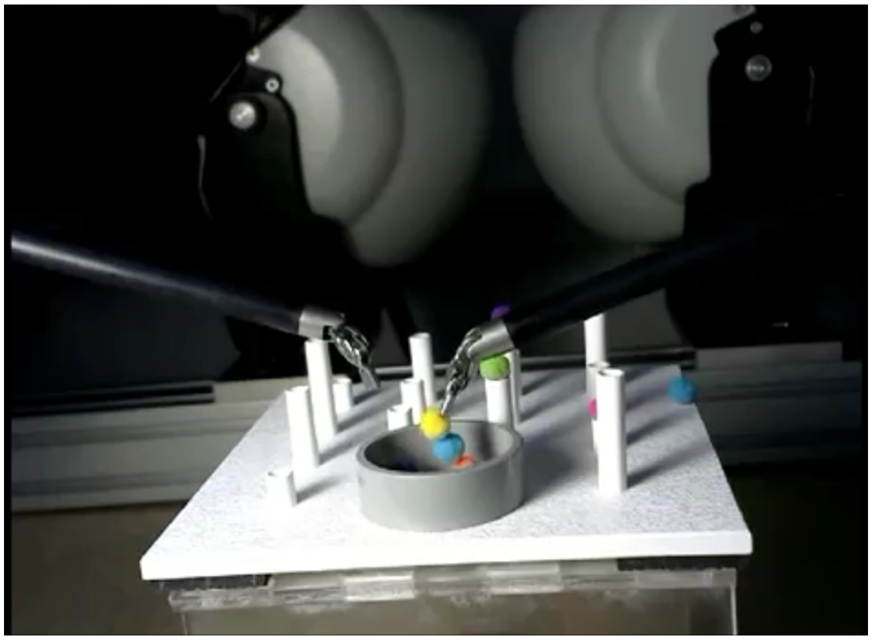


Figure 2.8: Example of the Pea on a Peg (PoaP) task from the ROSMA dataset [45].

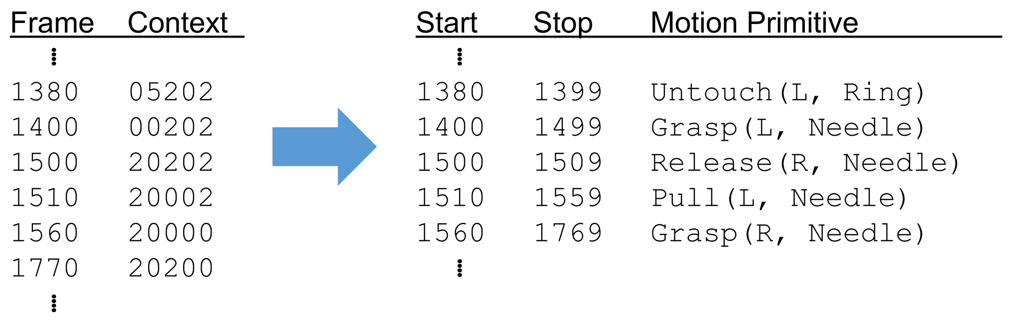
2.3 Surgical Task Models

2.3.1 Context to Motion Primitive Translation

The translation of context to MPs is an important step in leveraging high-quality context labels in creating surgical workflow annotations that enable the aggregation of data from different surgical tasks and datasets. The context labels are translated automatically into MP labels using the finite state machine (FSM) models for each task shown in Section 2.3.2. Given an input sequence of context labels, the corresponding sequence of MP labels are generated based on the transitions described in Tables 2.1 and 2.2 for the general and task-specific state variables respectively. Table 2.1 shows the set of universal MPs and corresponding changes to surgical context applicable to all tasks which enables the generalizability of this framework and allows activity recognition models to leverage these similarities across tasks. Table 2.2 shows the sets of MPs and corresponding changes to surgical context applied to specific dry lab tasks. We focus on dry lab tasks where the tools are graspers, but do not model or analyze the MP-specific functional and safety constraints here.

For each change of context in the input sequence, the specific changes to the state variables are identified and translated to the corresponding MPs. If multiple states changed between labeled frames, then Grasp and Release MPs would have a higher priority than Touch and Untouch MPs (if they were performed on the same object by the same tool). Otherwise, all MPs were listed in the MP transcript so that separate MP transcripts for the left and right sides could be generated. Context was labeled at 3 Hz and the context to MP translation assumes that states persist until the next context label in order to generate an MP label for each kinematic sample. Figure 2.9 shows an example of a sequence of context translated into MPs. This rule-based translation method also assumes that changes in context can be completely described by the definitions in Tables 2.1 and 2.2. Alternatively, data-driven and learning-from-demonstration approaches can be used for more realistic and personalized modeling of the tasks and label translations. To generate separate left and right label sets, MPs performed by each hand or arm of the robot are split into new transcripts.

MPs performed with a held object such as a needle are assigned to the transcript of the arm holding the object. Then, the Idle MP is used to fill in the gaps created by the separation so that every kinematic sample and video frame has a label and the labels are continuous.



Frame	Context
⋮	
1380	05202
1400	00202
1500	20202
1510	20002
1560	20000
1770	20200
⋮	

Start	Stop	Motion Primitive
⋮		
1380	1399	Untouch(L, Ring)
1400	1499	Grasp(L, Needle)
1500	1509	Release(R, Needle)
1510	1559	Pull(L, Needle)
1560	1769	Grasp(R, Needle)
⋮		

Figure 2.9: Example sequence of context translated into motion primitives.

2.3.2 Context and Motion Primitive Models of Tasks

We model the ideal performance of each task using context and MPs as a finite state machine (FSM) where the states are specific contexts and the transitions between states are MPs. The FSM models for the tasks are shown in Figures 2.10, 2.11, 2.12, 2.13, and 2.14.

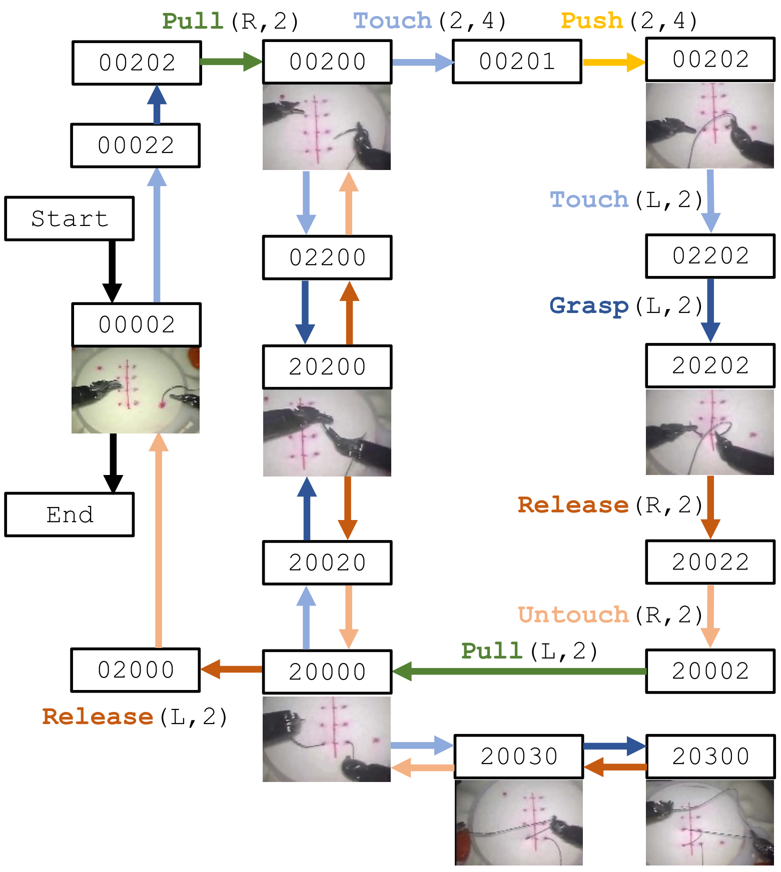


Figure 2.10: Finite state machine model representing the ideal performance of a Suturing trial with context and MPs.

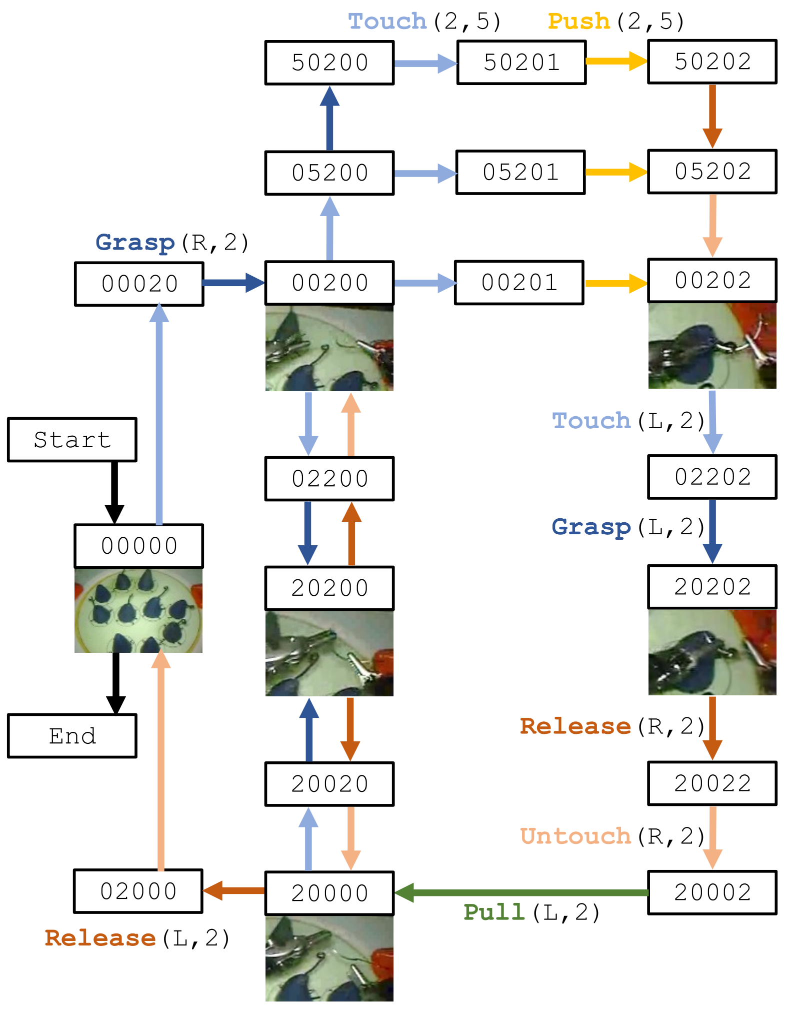


Figure 2.11: Finite state machine model representing the ideal performance of a Needle Passing trial with context and MPs.

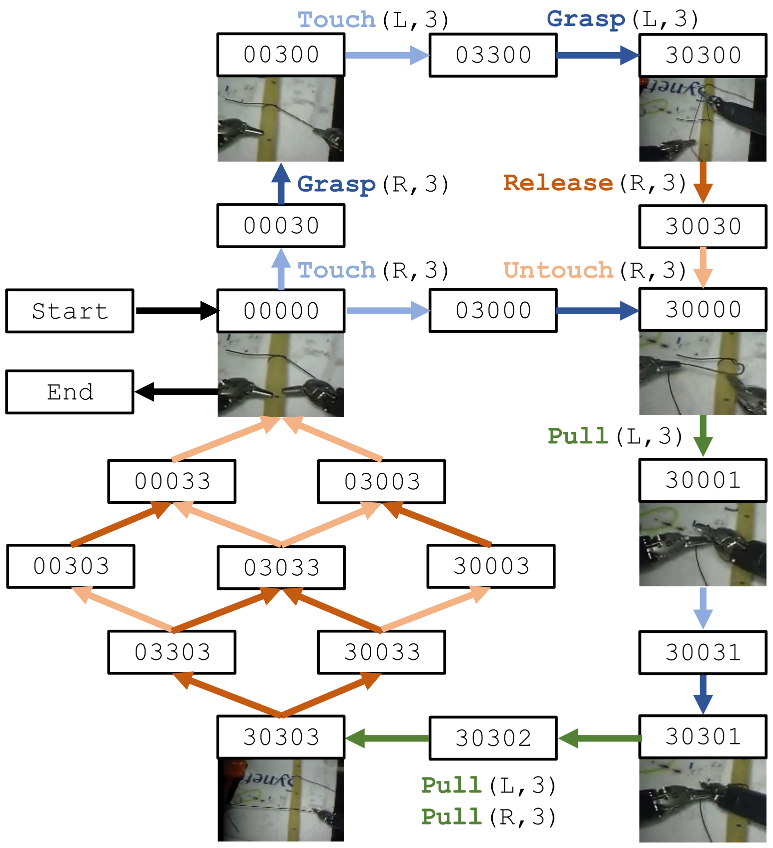


Figure 2.12: Finite state machine model representing the ideal performance of a Knot Tying trial with context and MPs.

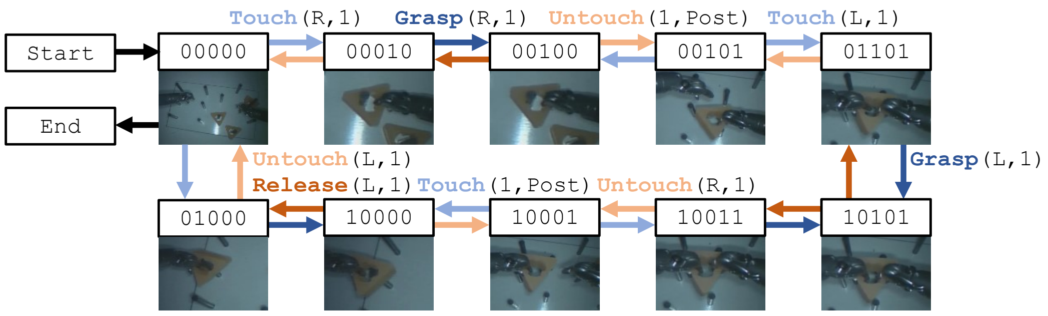


Figure 2.13: Finite state machine model representing the ideal performance of a Peg Transfer or Post and Sleeve trial with context and MPs.

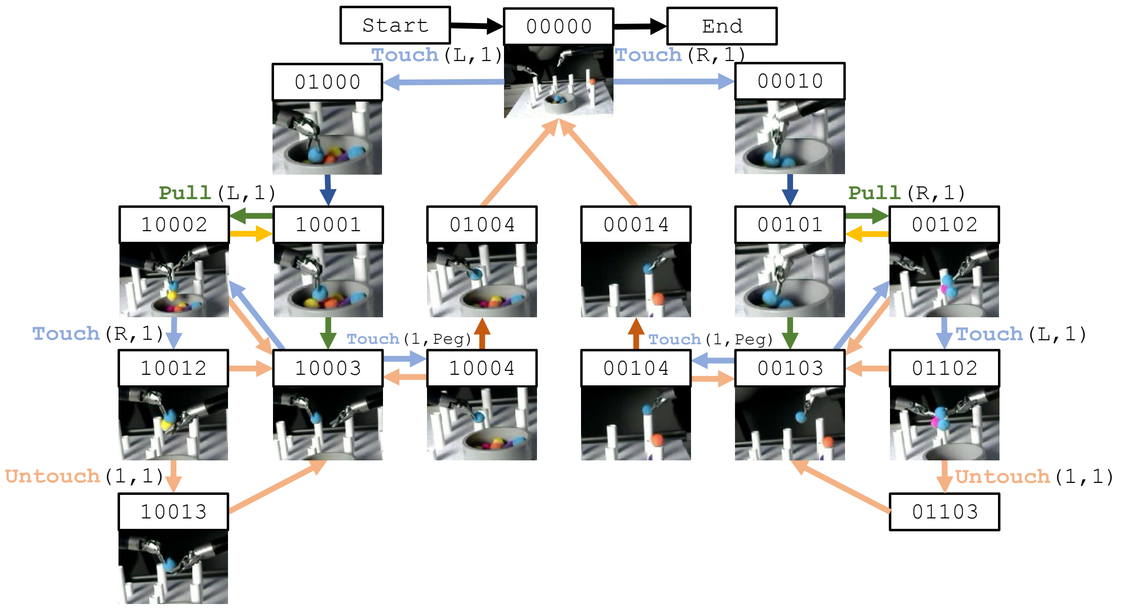


Figure 2.14: Finite state machine model representing the ideal performance of a Pea on a Peg trial with context and MPs.

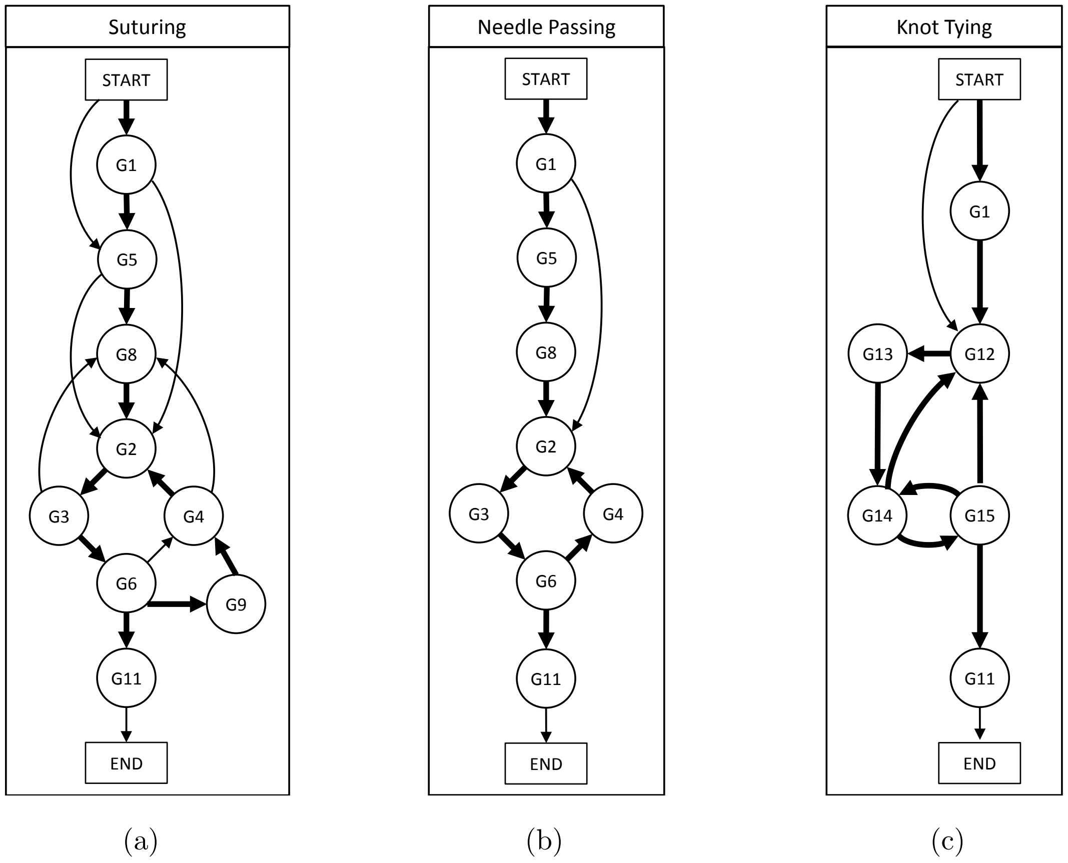


Figure 2.15: Grammar graphs showing transitions between gestures for the (a) Suturing, (b) Needle Passing, and (c) Knot Tying tasks. Adapted from [52].

These models differ from previous task models, such as grammar graphs which simply represent the typical sequences of gestures (e.g., grammar graphs for Suturing, Needle Passing, and Knot Tying from [52] shown in Figure 2.15).

The definition of MPs based on the changes in the surgical context could enable the translation of context and MPs to existing gesture labels and facilitate the aggregation of different datasets labeled with different gesture definitions. However, translation from context and MP labels to existing gesture definitions is complicated. This is because surgical process and workflow are affected by suboptimal performance and safety-critical events [33], so executional and procedural errors in gestures can affect the MP sequences for each gesture. Additional modeling is needed to develop and evaluate the MP to gesture translation which is future work beyond this dissertation.

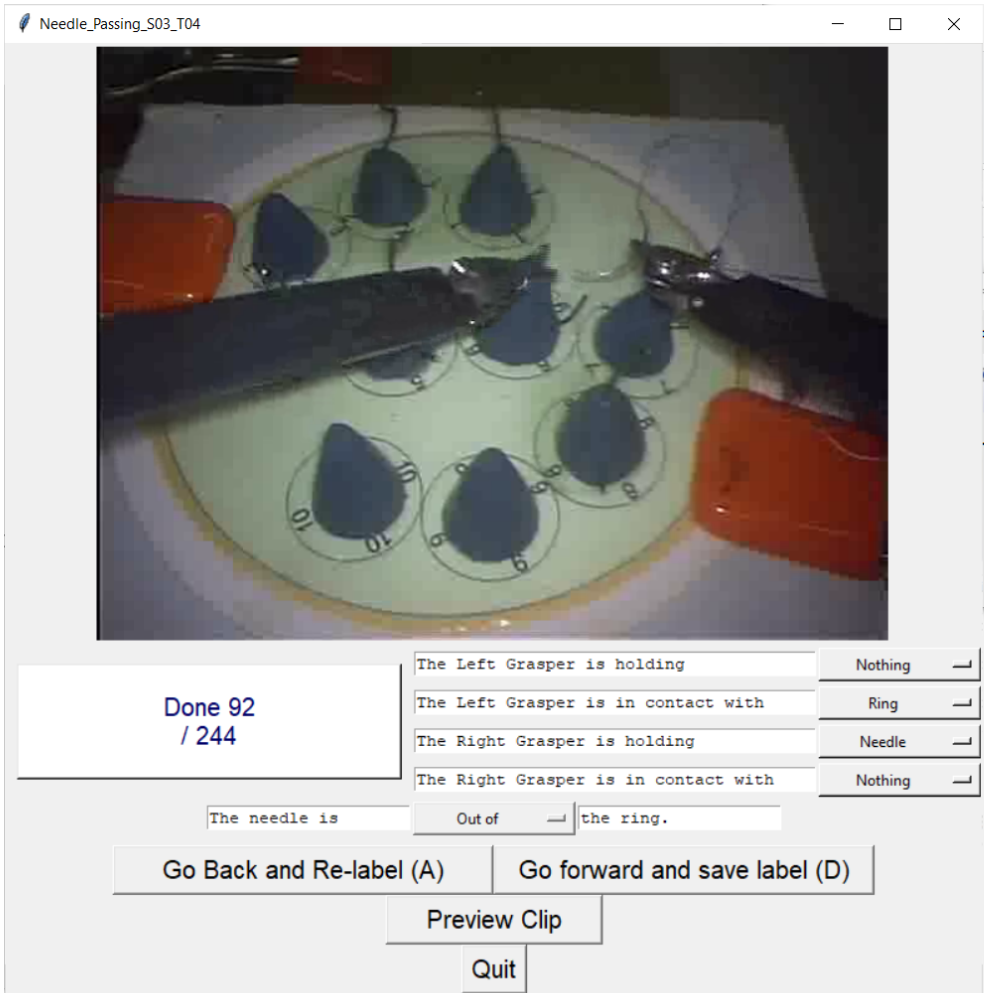
2.4 Surgical Context Labeling

2.4.1 Context Labeling Method

Surgical activity recognition models using supervised learning techniques require a large number of annotated video sequences [65]. However, the manual labeling of gestures is time consuming and subjective which can lead to labeling errors [25]. To address this, we have developed a tool for the manual annotation of surgical context (states of the objects and instruments) based on video data and used it to label all trials in six tasks from the JIGSAWS [43], DESK [44], and ROSMA [45] datasets.

Labeling video data for surgical context provides a more objective way of recognizing surgical activities and can thus lead to a higher level of agreement among annotators. As noted in [24], labels for surgical workflow require guidance from surgeons while annotations for surgical instruments do not. Since context labels document the objects held by or in contact with the left and right instruments, they rely less on surgical knowledge than gestures which require anticipating the next activities in a task to mark when a gesture has ended. Figure 2.16 shows a snapshot of the tool for the manual labeling of context based on video data. The annotators indicate the values of different state variables for frames in the video data and have the option of copying over the same values of the state variables to future frames until a change in context is observed. This differs from other labeling methods where annotators mark the start and end of each segment and assign a label to it.

To ensure reliable and high-quality annotations, three full sets of context labels were obtained using our context labeling tool for all trials. Two annotators, with extensive experience with the datasets and the dry lab robotic surgery tasks, each produced a full set of labels for all trials. The third set of labels was crowdsourced to 22 engineering students, who had no previous experience but were given a training module on the definitions of context, MPs, and their relationship, and how to use the labeling tool. The "Consensus" labels were then created using majority voting for each state variable.


Needle_Passing_S03_T04

Done 92 / 244

The Left Grasper is holding Nothing

The Left Grasper is in contact with Ring

The Right Grasper is holding Needle

The Right Grasper is in contact with Nothing

The needle is Out of the ring.

Figure 2.16: App for context labeling based on video data.

2.4.2 Evaluation of Labeling Method

In order to evaluate our context labeling and context to motion primitive translation methods, we obtain two more sets of labels, in addition to the Consensus set. A group of expert surgeons labeled a set of six trials (one from each task) for context, referred to as the "Surgeon" set, against which we evaluate the quality of the context labels. Then, three independent annotators also labeled a subset of trials in the JIGSAWS tasks for context, MPs, and gestures to assess and compare the different labeling methods and the context to motion primitive translation. This set is referred to as the "Multilevel" set.

We assess the quality of context labels generated using our labeling tool by measuring the agreement among the annotators in the Consensus set as well as the agreement between the Surgeon and Consensus sets of context labels using Krippendorff's Alpha. Then, we compare context-, MP-, and gesture-level labeling methods using labels in the Multilevel set.

Krippendorff's Alpha

Krippendorff's Alpha, α, is a commonly used statistical measure of inter-rater reliability. It indicates how much the data from two or more methods can be trusted to represent the real phenomenon [66]. As shown in Equation 2.2, it is calculated by considering the probability D_e that two labelers produced the same annotation due to change rather than agreement on the data to label, and the observed disagreement D_o between each labeler's annotations:

$$\alpha = 1 - \frac{D_o}{D_e} \tag{2.2}$$

The coefficient α is a value ranging from -1 to 1, where larger positive values indicate greater agreement, $\alpha = 0$ indicates no agreement other than by chance, and negative values indicate greater disagreement. Table 2.4 lists the thresholds of α for the interpretation of different levels of agreement based on [67] and [68].

Table 2.4: Interpretation of Krippendorff's Alpha (α) from [69].

Range	Interpretation
$\alpha > 0.8$	Near-perfect
$0.6 < \alpha \leq 0.8$	Substantial
$0.4 < \alpha \leq 0.6$	Moderate
$0.2 < \alpha \leq 0.4$	Fair
$\alpha \leq 0.2$	Slight

The sequence of states annotated by each of the labelers were encoded as numbers and did not have numerical significance, so they can be best described as categorical data. Thus, the nominal distance (also called the difference function) is best suited to quantify the agreement between labelers annotating for context. The nominal distance function shown in Equation 2.3 is used to calculate D_e and D_o [69] as given in Equation 2.4 where n_l is the number of labelers and n_u is the total number of frames annotated by two or more labelers.

$$d_{\text{nominal}}(\text{label}_1\ \text{label}_2) = \begin{cases} 0 & \text{if label}_1 = \text{label}_2 \\ 1 & \text{if label}_1 \neq \text{label}_2 \end{cases} \tag{2.3}$$

$$\begin{aligned} D_o &= \frac{1}{2n_u n_l(n_l - 1)} \sum_{l_1, l_2 \in \text{all labels}} d_{\text{nominal}}(l_1, l_2) \\ D_e &= \frac{1}{2n_u n_l(n_u n_l - 1)} \sum_{l_1, l_2 \in \text{all labels}} d_{\text{nominal}}(l_1, l_2) \end{aligned} \tag{2.4}$$

Consensus Context Labels

The quality of the context labels is measured by calculating Krippendorff's Alpha among the annotators in the Consensus set as well as the agreement between the Surgeon and Consensus sets of context labels.

Agreement among annotators We report the agreement among crowdsourced annotators using the average Krippendorff's Alpha in the second column of Table 2.5. Four of the tasks have α above 0.8 indicating near-perfect agreement and the other two tasks have α above 0.6 indicating substantial agreement among annotators. The average for all tasks, weighted for the number of frames in each task, was 0.84 indicating near-perfect agreement in context labeling overall. We also observed that long segments of near-perfect agreement were punctuated by disagreements at the transitions between context groups. However, this disagreement is limited to a few context states instead of the gesture label for a specific frame which results in much greater agreement between annotators when labeling for context than for gestures. This shows that our method for labeling context results in a high-quality set of fine-grained labels.

Table 2.5: Krippendorff's Alphas among annotators and between Consensus and Surgeon context labels.

Task	Among annotators	Between Consensus and Surgeon
S	0.69	0.86
NP	0.85	0.90
KT	0.79	0.94
PT	0.90	0.94
PoaP	0.83	0.93
PaS	0.89	0.97

Agreement between Consensus and Surgeon context labels Krippendorff's Alpha between the Consensus and Surgeon context labels for each task is shown in the third column of Table 2.5. All tasks had an α of at least 0.8 and the average for all tasks, weighted for the number of frames in each task, was 0.92 indicating near-perfect agreement between crowdsourced context labels from non-experts and those given by expert surgeons.

Multilevel Labels

The agreement among annotators for labels at different granularities is shown in Table 2.6. There is the least agreement when labeling using the descriptive gesture definitions since the Krippendorff's Alpha values were 0.24, 0.08, and 0.20 for the Suturing, Needle Passing, and Knot Tying tasks, respectively, which is only slight to fair agreement according to Table 2.4. The agreement when annotating for MPs directly ranges from 0.26 to 0.41 which is fair to moderate agreement according to Table 2.4. Thus, directly labeling MPs is also difficult, likely due to their short durations. The greatest agreement is seen for context which has substantial to near-perfect agreement. This is likely because the context labels are based on well-defined interactions among surgical tools and objects that can be observed in video data.

The agreement of the Multilevel context and gesture labels with the Surgeon context labels and the JIGSAWS gesture labels, respectively, are shown in Table 2.6.

Table 2.6: Krippendorff's Alphas among annotators for context-, MP-, and gesture-level labels in the Multilevel set.

	Multilevel			Multilevel vs. Surgeon Context	Multilevel vs. JIGSAWS Gestures
Task	Context	MPs	Gestures		
S	0.72	0.33	0.24	0.86	0.34
NP	0.91	0.41	0.08	0.90	0.04
KT	0.89	0.26	0.20	0.89	0.06

There is much greater agreement when labeling for context than for gestures and the existing JIGSAWS gesture labels are difficult to reproduce since the Krippendorff's Alpha for the Needle Passing and Knot Tying tasks are only 0.04 and 0.06, respectively. This may be because the JIGSAWS gesture labels were generated more subjectively by only one annotator by watching the videos and developing the gesture definitions in consultation with a surgeon [43]. Thus, we again see that crowdsourcing context labels results in high-quality annotations that are comparable to those from surgeons and are thus representative of expert knowledge.

2.4.3 Evaluation of Context to MP Translation

Context to MP translation allows us to leverage high-quality context labels in creating surgical workflow annotations and aggregating different surgical datasets. Context labels are translated automatically into MPs using the FSMs for each task. To evaluate our framework, we obtain MP graphs (similar to grammar graphs but with nodes representing MPs) from two expert robotic surgeons describing the execution of S, NP, and KT, and compare our proposed models to expert knowledge. To evaluate our context labeling and context to MP translation methods, we obtain two more sets of labels, in addition to the Consensus set. Two trios of independent annotators labeled subsets of trials in the JIGSAWS and DESK tasks, respectively, for context, MPs, and gestures to assess and compare the different labeling methods and the context to MP translation. We refer to these labels as the Multilevel set and their gesture labels can be compared to the gesture labels from JIGSAWS and DESK. The surgeons also labeled a set of six trials (one from each task to capture task-diversity, and overlapping the trials in the Multilevel set) for context, referred to as the Surgeon set, against which we evaluate label quality. To assess the performance of the context to MP translation, we translate the context labels in the Multilevel annotations set and compare the resulting translated MP transcripts to the ground truth MP labels from each annotator. We calculate the accuracy and edit score for each task as described below.

Accuracy

Given the lists of predicted and ground truth labels, the accuracy is the number of correctly predicted samples divided by the total number of samples in the trial, and reported as a percentage. Accuracy is a standard frame-wise metric, but doesn't consider sequential relations or oversegmentation [8].

Edit Score

Given sequences of predicted labels, P, and ground truth labels, G, the edit score is calculated as defined in [70] by normalizing the Levenshtein edit distance, $\text{edit}(G, P)$. The edit distance measures the number of insertions, deletions, and replacements needed to transform the sequence of predicted labels, P, to match the sequence of ground truth labels, G, and is normalized by the maximum length of the predicted and ground truth sequences as in Equation 2.5. The edit score ranges from 0 to 100 where a higher score indicates greater similarity between the predicted and ground truth sequences. Edit score is a segmental metric for the order and segmentation of activity predictions, but doesn't consider the timing of boundaries between activities [8].

$$\text{Edit Score} = \left(1 - \frac{\text{edit}(G, P)}{\max(\text{len}(G), \text{len}(P))}\right) \times 100 \tag{2.5}$$

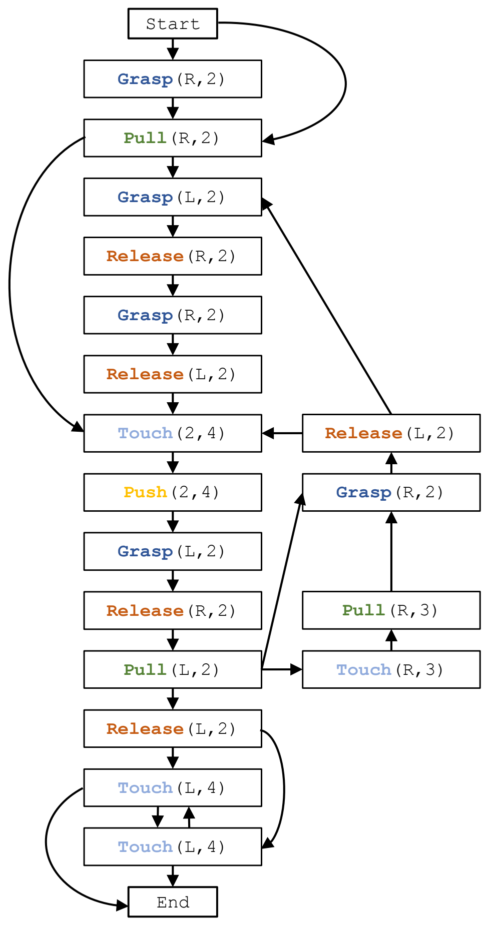
Graph Edit Score

Graph edit score (GES) is the normalized graph edit distance (GED) calculated using Equation 2.6 by dividing the minimum cost of transforming A to B by the maximum GED (cost of deleting all nodes and edges in A and inserting all nodes and edges in B, where C represents an empty graph). GED is implemented using networkx [71] with the Start nodes as the root and a timeout of 18 hours.

$$\text{GES} = \left(1 - \frac{\text{GED}(A, B)}{(\text{GED}(A, C) + \text{GED}(B, C))}\right) \times 100 \tag{2.6}$$

Task Modeling

We evaluate our framework by comparing the MP graphs for S, NP, and KT to MP graphs defined by expert surgeons as shown in Figures 2.17, 2.18, and 2.19 using graph edit score. Surgeons may not be available to verify future models, so it is important to check that those for surgically-relevant tasks represent expert knowledge. The surgeon-defined graphs were created by obtaining the MP sequences for each gesture and substituting them into the grammar graphs in Figure 2.15. In order to compare the FSMs with the MP graphs defined by expert surgeons, the MP transitions from the FSMs were converted to MP graphs. Touch and Untouch MPs that immediately preceded or followed Grasp and Release MPs, respectively, were removed since the surgeons assumed they were combined when creating their MP graphs.



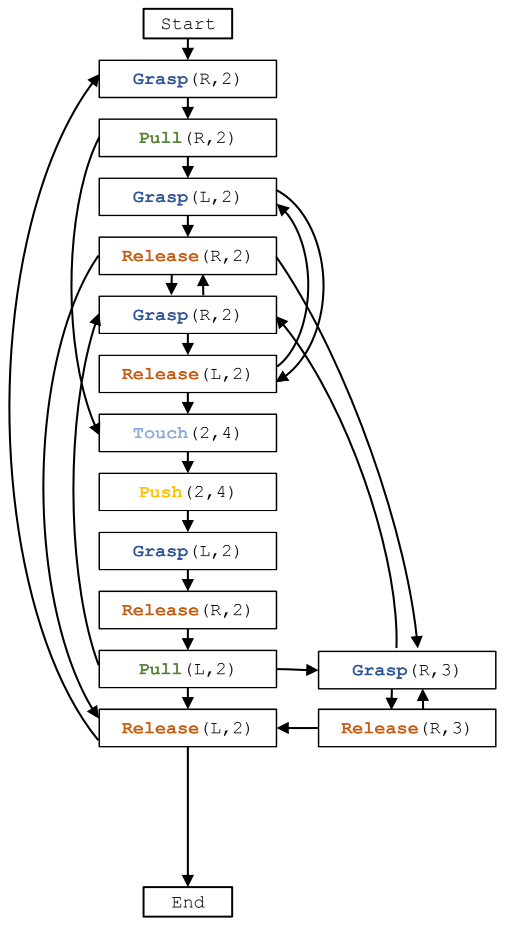
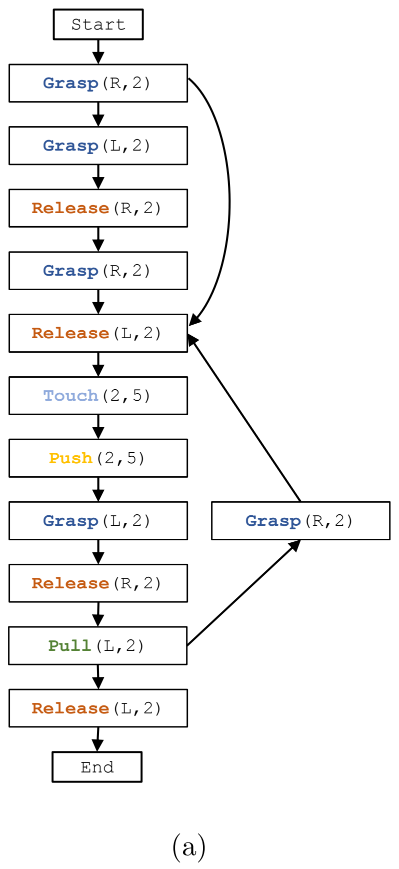


Figure 2.17: (a) Surgeon-defined and (b) proposed MP graph models for the Suturing task.



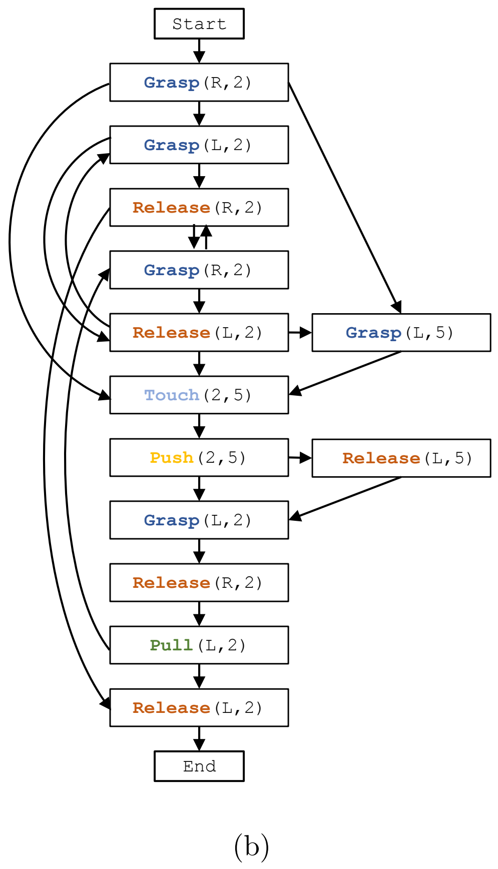
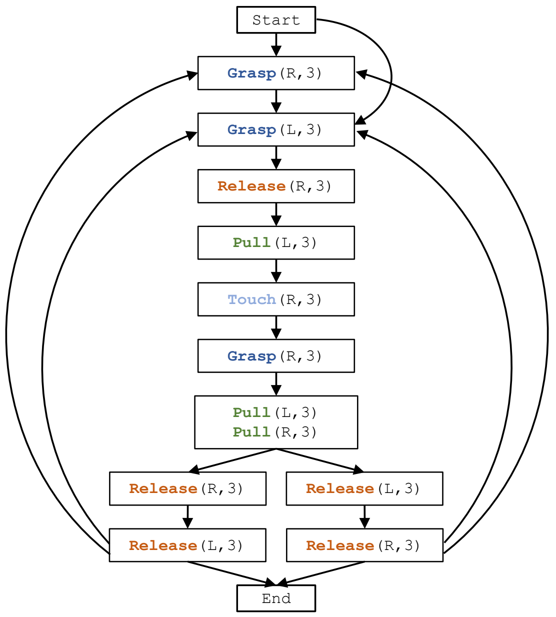
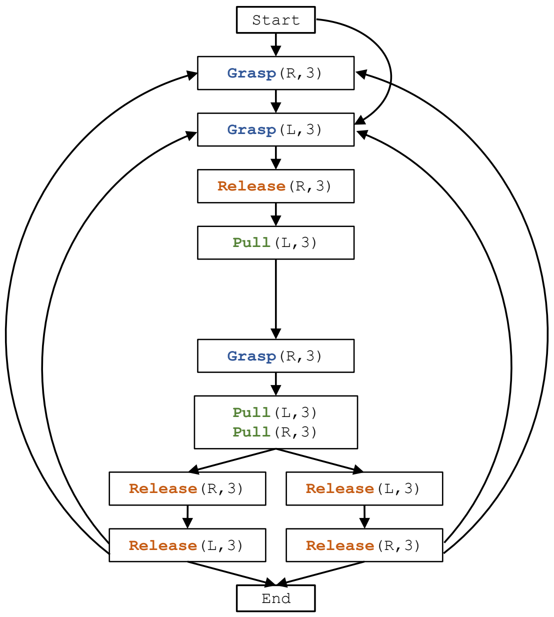


Figure 2.18: (a) Surgeon-defined and (b) proposed MP graph models for the Needle Passing task.



(a)



(b)

Figure 2.19: (a) Surgeon-defined and (b) proposed MP graph models for the Knot Tying task. The Touch(R, 3) in (a) refers to a different thread than that of the following Grasp(R, 3).

GES was lowest for S in Table 2.7 because it was the most complex task and the surgeons had additional MPs to represent passing the needle from left to right while the proposed model represented it as the inverse of passing the needle from right to left. Also, although physically possible, several transitions in our proposed graph for S were not in the surgeons' graph since they may not represent an efficient execution of the task. Comparatively, KT was a simpler task, and overall the proposed models are good representations of expert knowledge.

Table 2.7: Graph edit scores between the proposed and surgeon-defined MP graphs.

Task	Graph Edit Score
Suturing	76.4
Needle Passing	83.6
Knot Tying	93.3

Quality of Multilevel Labels

We assess context label quality by measuring the agreement among annotators in the Consensus set and the agreement between the Surgeon and Consensus sets of context labels using Krippendorff's Alpha [65]. Then, we compare context-, MP-, and gesture-level labeling methods using labels in the Multilevel set. The context and MP labels in the Multilevel set show variability with the type of task (S, NP, KT) and with the skill of the annotator, both of which can affect the resulting translated MP labels and their evaluation. So, we first assessed the quality of each annotator based on their agreement with the Surgeon context labels and JIGSAWS/DESK gesture labels with respect to the label for each frame and the overall label sequence.

Table 2.8 shows α, and Table 2.9 shows accuracies and edit scores for each annotator when labeling for context (compared to Surgeons) and gestures (compared to JIGSAWS/DESK). For JIGSAWS, annotator 3 was the most reliable annotator overall with annotator 2 almost as reliable for context labels. Less variation was seen among the annotators for DESK.

Table 2.8: Krippendorff's Alphas of Multilevel labels compared to Surgeon context and JIGSAWS/DESK gesture labels.

	Context			Gestures		
Task	Annotator 1	Annotator 2	Annotator 3	Annotator 1	Annotator 2	Annotator 3
S	0.87	0.89	0.91	0.06	0.05	0.76
NP	0.88	0.88	0.85	0.09	0.07	0.47
KT	0.72	0.85	0.87	0.08	0.39	0.48
PT	0.93	0.89	0.93	0.40	0.41	0.45

Table 2.9: Accuracies and edit scores of Multilevel labels compared to Surgeon context and JIGSAWS/DESK gesture labels.

	Context						Gestures					
	Annotator 1		Annotator 2		Annotator 3		Annotator 1		Annotator 2		Annotator 3	
Task	Acc	Edit	Acc	Edit	Acc	Edit	Acc	Edit	Acc	Edit	Acc	Edit
S	67.6	69.2	72.3	73.0	77.3	78.0	25.5	49.1	26.1	42.0	85.6	88.7
NP	74.1	74.9	76.0	76.6	74.7	74.9	34.0	56.8	27.7	49.5	47.3	54.5
KT	32.5	32.5	60.7	60.7	62.2	62.2	24.0	51.5	52.3	61.0	62.5	64.1
PT	83.3	84.1	83.7	87.2	83.7	84.0	49.8	52.7	50.6	52.3	54.1	55.8

Context to Motion Primitive Translation Accuracy

We translate the context labels given by each annotator into MP labels and compare them with the corresponding ground truth MP labels that each annotator labeled directly. The accuracies between the translated MP labels and ground truth MP labels are shown in Table 2.10. We find that the context to MP translation accuracy was higher for annotators 1 and 3 for JIGSAWS, and annotators 2 and 3 for DESK. There is inter-rater variability across tasks, where KT and PT generally had higher metrics while S had lower metrics likely due to task complexity.

Table 2.10: Accuracies and edit scores between ground truth and translated MPs for Multilevel and Surgeon labels.

	Annotator 1		Annotator 2		Annotator 3		Surgeon	
Task	Acc	Edit	Acc	Edit	Acc	Edit	Acc	Edit
S	0.27	33.9	0.18	26.9	0.23	30.1	0.29	30.8
NP	0.64	67.4	0.27	45.1	0.45	46.4	0.22	25.4
KT	0.50	53.8	0.31	56.1	0.53	56.8	0.42	50.3

However, the ground truth MP labels used in this evaluation had very low agreement among annotators compared to the context labels and assessing their reliability is the subject of future work. Collaborations with surgeons could facilitate the generation of high-quality annotations using multilevel labeling methods to better evaluate these different methods and improve the automated translation between levels.

2.5 COMPASS Dataset

We create the COMPASS (COntext and Motion Primitive Aggregate Surgical Set) dataset by aggregating data from 39 trials of Suturing (S), 28 trials of Needle Passing (NP), and 36 trials of Knot Tying (KT) performed by eight subjects from the JIGSAWS dataset [43]; 47 trials of Peg Transfer (PT) performed by eight subjects from the DESK dataset [44]; and 65 trials of Post and Sleeve (PaS) and 71 trials of Pea on a Peg (PoaP) performed by 12 subjects from the ROSMA dataset [45]. Thus, COMPASS contains a total of 286 trials by 28 different subjects which is about three times the number of trials and subjects in JIGSAWS. Tables 2.11, 2.12, and 2.13 list the numbers of MPs and gestures of each type in each task and dataset. By using standardized definitions, COMPASS has fewer classes and more examples than other datasets.

Table 2.11: Number of motion primitives (MPs) in each task and the COMPASS dataset: Suturing (S), Needle Passing (NP), and Knot Tying (KT) from JIGSAWS [43]; Peg Transfer (PT) from DESK [58]; and Pea on a Peg (PoaP) and Post and Sleeve (PaS) from ROSMA [45].

	JIGSAWS			DESK	ROSMA		COMPASS
MP	S	NP	KT	PT	PoaP	PaS	All
Grasp	471	373	283	323	577	824	2851
Release	441	365	247	313	556	776	2698
Touch	518	330	135	539	1782	1598	4902
Untouch	314	206	111	364	1261	1131	3387
Pull	194	114	235	0	525	0	1068
Push	179	119	0	0	2	0	300

This aggregate dataset contains video and kinematic data along with context and motion primitive labels, and gesture labels when available from the original dataset. The videos are at 30 fps for the stereoscopic JIGSAWS and DESK tasks and 15 fps for the single camera for the ROSMA tasks. The kinematic data have been downsampled to 30 Hz and contain position, velocity, orientation (in quaternions), and gripper angle variables. Since linear velocity data was not available for all tasks, it was derived from the position data using a rolling average over five samples. The ROSMA dataset did not contain gripper angle, so a separate round of manually labeling video data was performed to approximate the gripper angle as open or closed. The consensus context labels are at 3 Hz and the automatically generated motion primitive labels are interpolated to 30 Hz for both arms of the robot so that every kinematic sample has an MP label. The original gesture label files from the JIGSAWS and DESK datasets are renamed under the new naming convention and included to promote comparisons between data and label sets. To uniquely identify each individual file, a naming system that includes the task, subject number, and trial number was applied to facilitate matching kinematic, video, and label files. For example, the name Pea_on_a_Peg_S02_T05 identifies the fifth trial of Pea on a Peg performed by subject two. The dataset is organized into different tasks with directories for the kinematic and video data, and each type of label. The subject and trial numbers from the original datasets are retained so that the LOSO and LOUO setups from [52] can be extended to COMPASS.

The tools for labeling surgical context based on video data, the automated translation of context to motion primitive labels, as well as the aggregated dataset with context and motion primitive labels are made publicly available at `https://github.com/UVA-DSA/COMPASS` to facilitate further research and collaboration in this area.

Table 2.12: Number of gestures in each JIGSAWS task and dataset.

Gesture	Suturing	Needle Passing	Knot Tying	JIGSAWS
G1	29	30	19	78
G2	166	117	0	283
G3	164	111	0	275
G4	119	83	0	202
G5	37	31	0	68
G6	163	112	0	275
G8	48	28	0	76
G9	24	1	0	25
G10	4	1	0	5
G11	39	25	36	100
G12	0	0	70	70
G13	0	0	75	75
G14	0	0	98	98
G15	0	0	73	73

Table 2.13: Number of gestures in DESK for Peg Transfer performed on the da Vinci surgical robot.

Gesture	Peg Transfer
S1	146
S2	147
S3	137
S4	146
S5	146
S6	135
S7	135

2.6 Related Work

The "Language of Surgery" project [72] models surgical procedures as a language and uses grammar to dictate how gestures are combined to perform tasks, and a hierarchical framework has been proposed to model surgical procedures [10]. Available datasets primarily focus on the task, gesture, and action levels as summarized in Tables 2.14 and 2.15. Within this hierarchical framework, tasks consist of a sequence of gestures, and gestures can be decomposed into actions.

2.6.1 Surgical Gestures

Gestures are defined as units of surgical motion with a specific intent (e.g., insert needle through tissue) and semantic meaning. The JIGSAWS dataset [43], provides gesture-level labels for the Suturing, Needle Passing, and Knot Tying tasks in a dry lab setting. Recent works have introduced new datasets as shown in Table 2.14, but differing gesture definitions limit comparisons between them as well as their generalizability to other tasks. Specifically, [73], [59], and [74] all performed gesture recognition based on kinematic data, but used different datasets and gesture definitions making comparisons between these works difficult. [60] fused kinematic, video, and event data to recognize and predict data, and [61] explored using multimodal attention for gesture recognition. But, previous works have not combined data from multiple sets since the gesture labels are incompatible.

2.6.2 Action Triplets

Action triplets, defined as <surgical tool/instrument, action verb, target anatomy>, are used to describe tool-tissue interactions (TTI) in surgical process modeling [22]. One of the early works formalizes laparoscopic adreanectomies, cholecystectomies and pancreatic resections [23] with surgical activities in the form of action triplets for surgical phase inference. [76] annotates two robotic surgery datasets of MICCAI robotic scene segmentation and Transoral Robotic Surgery (TORS) in the form of action

Table 2.14: Datasets and definitions for gesture recognition.

Paper	Dataset	Tasks	Gestures	
Gao 2014 [43] [52]	JIGSAWS	Suturing	G1 - Reaching for needle with right hand	
	• Video	Needle Passing	G2 - Positioning needle	
	• Kinematics	Knot Tying	G3 - Pushing needle through tissue	
			G4 - Transferring needle from left to right	
			G5 - Moving to center with needle in grip	
			G6 - Pulling suture with left hand	
			G7 - Pulling suture with right hand	
			G8 - Orienting needle	
			G9 - Using right hand to help tighten suture	
			G10 - Loosening more suture	
			G11 - Dropping suture at end and moving to end points	
			G12 - Reaching for needle with left hand	
			G13 - Making C loop around right hand	
			G14 - Reaching for suture with right hand	
			G15 - Pulling suture with both hands	
DiPietro 2019 [73]	MISTIC-SL	Suturing	G1-G12 & G14	
	• Video	Needle Passing	G13 - Grab suture using 2nd needle driver	
	• Kinematics	Knot Tying	G15 - Rotate suture twice using 1st needle driver around 2nd needle driver	
			G16 - Grab suture tail using 2nd needle driver in knot tying	
			G17 - Pull suture tail using 2nd needle driver through knot	
			G18 - Pull ends of suture taut	
			G19 - Rotate suture once using 2nd needle driver around 1st needle driver	
			G20 - Grab suture tail using 1st needle driver in knot tying	
			G21 - Pull suture tail using 1st needle driver through knot	
			G22 - Grab suture using 1st needle driver	
Gonzalez 2020 [44]	DESK	Peg Transfer	S1 - Approach peg	S5 - Transfer peg - Exchange
	• Video		S2 - Align & grasp	S6 - Approach pole
	• Kinematics		S3 - Lift peg	S7 - Align & place
			S4 - Transfer peg - Get together	
Menegozzo 2019 [59]	V-RASTED	Pick and Place	1 – Collecting ring	4 – Failing 1
	• Video		2 – Passing ring R to L	5 – Failing 2
	• Kinematics		3 – Posing ring on pole	6 – Failing 3
Goldbraikh 2022 [74]	own (open)	Suturing	No gesture	Instrument tie
	• Video	(not robotic)	Needle passing	Lay the knot
	• Kinematics		Pull the suture	Cut the suture
Qin 2020 [75]	RIOUS	Ultrasonic	S1 Probe released, out of view	S5 Lifting probe up
	• Kinematics	probing	S2 Probe released, in view	S6 Carrying probe to tissue surface
	• Video		S3 Reaching for probe	S7 Sweeping
	• Events		S4 Grasping probe	S8 Releasing probe

triplets to generate surgical reports. In the SARAS Endoscopic Surgeon Action Detection (ESAD) challenge [77], instead of action triplets, actions are described by both the verb and the anatomy. In the CholecTriplet2021 benchmark challenge for surgical action triplet recognition [21], the challenge dataset, CholecT50, consists of 50 video recordings of laparoscopic cholecystectomy labeled for 100 action triplet classes composed from 6 instruments, 10 verbs, and 15 targets. Despite being more descriptive of the surgical scene, the number of action triplets in the form of verbs, instruments, and targets can grow exponentially compared to a more limited number of gestures.

Table 2.15: Datasets and models for surgical actions.

Paper	Dataset	Tasks	Actions		
Nwoye 2022 [49]	CholecT50	Laparoscopic	Aspirate	Dissect	Pack
	• Video	cholecystectomy	Clip	Grasp	Retract
			Coagulate	Irrigate	
			Cut	Null	
Li 2022 [78]	EndoVis 2018	Nephrectomy	Cauterization	Looping	Clipping
	• Video		Suction	Idle	Retraction
			Staple	Tool manipulation	
			Ultrasound sensing	Suturing	
Meli 2021 [55]	own (dVRK)	Ring Transfer	Move		
	• Video		Grasp		
	• Kinematics		Release		
			Extract		
Forestier 2012 [79]	own (open)	Lumbar disk	Right:	Left:	
	• Video	herniation	Sew	Hold	
		(not robotic)	Install	Install	
			Hold	Remove	
			Remove		
			Coagulate		
			Swab		
			Irrigate		
Wagner 2021 [20]	EndoVis 2019	Laparoscopic	Grasp		
	• Video	cholecystectomy	Hold		
			Cut		
			Clip		
De Rossi 2021 [53]	own	Pick and place	A01 – MS moves to ring		
	(dV and SARAS)	(semi-	A02 – MS picks ring		
	• Video	autonomous	A03 – MS moves ring to exchange area		
		and cooperative)	A04 – AS moves to ring		
			A05 – AS grasps ring and MS leaves ring		
			A06 – AS moves ring to delivery area		
			A07 – AS drops ring on target		
			A08 – AS moves to starting position		
Valderrama	PSI-AVA	Radical	Cauterize	Open	Still
2022 [50]	• Video	prostatectomy	Close	Open Something	Suction
			Close Something	Pull	Travel
			Cut	Push	Wash
			Grasp	Release	
			Hold	Staple	
Ma 2021 [80]	own (dV)	Renal hilum	Single blunt dissection:	Single sharp dissection:	Combination:
	• Video	dissection	Spread	Cold cut	Pedicalize
		(Partial	Peel/push	Hot cut	2-hand spread
		nephrectomy)	Hook	Burn dissect	Coagulate then cut
Huaulmè 2021 [51]	MISAW	Suturing	Catch	Loosen completely	Position
	• Kinematic	Knot Tying	Give slack	Loosen partially	Pull
	• Video		Hold	Make a loop	
			Insert	Pass through	

2.6.3 Surgical Actions

Surgical actions are generally referred to as the level of the surgical hierarchy below gestures. Motions, motion primitives, and the action verb in action triplets are surgical actions. In surgical process modeling, [10] and [9] define motions as an activity performed by only one hand and without semantic meaning. Many other works define actions as atomic units as listed in Table 2.15.

The gesture and action label datasets are mostly proposed for surgical workflow segmentation [20,50,81–83], and gesture [59,60,73,74], action, or action triplet recognition [20,49,78]. However, different datasets have varying definitions of gestures and actions. This makes combining data from multiple sets challenging. Besides varying definitions, prior datasets with surgical actions mainly contain video data, and none of the previous datasets examine the process of labeling and the labeling agreement. Existing datasets also do not differentiate between activities performed by the left and right hands, which is important for detailed skill assessment and the analysis of bimanual coordination. In our framework, we formally define motion primitives for the left and right hands. Our motion primitives are sets of action definitions that are generalizable across different datasets. We also look into how motion primitives relate to surgical context and task progress. Our proposed dataset contains both kinematic and video data along with context and motion primitives labels for a total of six dry lab tasks. Our dataset will facilitate the development of recognition, skill assessment, and error detection models using both vision and kinematic data.

2.7 Conclusion

In this chapter, we defined surgical context to represent the interactions between surgical instruments and objects in the surgical environment and presented a framework for modeling dry lab surgical tasks as finite state machines where motion primitives cause changes in surgical context, characterized by tool and object/tissue interactions. We then presented and analyzed methods for labeling context, MPs, and gestures and found that our method for labeling context achieves substantial to near-perfect agreement between non-expert annotators and expert surgeons. By applying our framework to three publicly available datasets (JIGSAWS, DESK, and ROSMA), we created the COMPASS dataset containing video and kinematic data and labeled with unified context and motion primitive labels. COMPASS has nearly three times as much data as JIGSAWS alone and has greater task and subject diversity. Our framework enables dataset aggregation with multiple consistent levels, greater objectivity and agreement in labeling, and the modeling of tasks with finite state machines.

Furthermore, surgical data collection and aggregation would benefit from platform-independent and multimodal methods of recording data from surgical robots. Towards this goal, we present preliminary work on the development of a Data Collection System (DCS) and a proof of concept of its operation in Appendix A.

Future work includes extending the framework to tasks from real surgical procedures which would be accomplished by defining additional tools (e.g., scissors), objects (e.g., vessels and tissues), task-specific state variables to augment the context labels, and the associated MPs (e.g., Cut from [49]) that change those state variables. The context to MP translation could be improved by using data-driven and learning-from-demonstration approaches for more realistic and personalized task modeling. The definition of MPs based on changes in the surgical context could also enable translations between existing gestures and MPs. But this is complicated due to executional and procedural errors affecting the order and performance of MPs that comprise gestures, and is thus the subject of future work. These relationships could provide insight into the functional and safety constraints of MPs and enable autonomous operations, safety monitoring, and recovery. Our standardized set of context and motion primitive labels enables the generalized modeling and comparison of surgical activities between datasets and tasks. This supports the development of models for surgical activity recognition, skill analysis, error detection, and surgical automation [14] by providing examples of fine-grained motions, and multi-granularity models for improved fine-grained activity recognition [51].

Chapter 3

Surgical Context Inference and Activity Recognition

3.1 Background

Surgical robots for minimally invasive surgery (MIS) enable surgeons to operate with greater flexibility and precision, thus reducing incision size, recovery time, and scarring. Their widespread adoption into surgical specialties such as urology, gynecology, and general surgery has opened up new fields of interdisciplinary research and surgical data science [6]. Many of these areas depend on the modeling and decomposition of surgical procedures into smaller units based on surgical process modeling [9, 10]. Automatic segmentation of the surgical scene and workflow are active areas of research [50], and developed methods are used for skill assessment [12, 13, 28], error detection [17, 18], and autonomy [86]. Methods and models for detecting and understanding the surgical environment and surgeon's activities will enable improvements in the safety, dependability, security, and efficiency of surgical robots.

Surgical scene segmentation has benefited from challenges such as the 2017 and 2018 EndoVis workshops at MICCAI which presented challenges to perform robotic instrument and scene segmentation using images from a da Vinci Xi robot in porcine

This chapter contains material from the works [84] and [85], coauthored with I. Reyes, Z. Li, and H. Alemzadeh, copyrighted by IEEE.

procedures [19, 28]. High-performing models from these challenges have advanced video data analysis, and datasets for instrument and object segmentation in MIS procedures are more plentiful.

Video data has also been used for finer-grained surgical workflow segmentation such as recognizing action triplets [49,55,78] based on the interactions between robotic tools and objects in the surgical environment. However, an explicit relation between these interactions and units of surgical activity such as gestures or motion primitives has not been defined.

There has also been much work in recognizing gestures and actions as summarized in [8]. The JIGSAWS dataset [43] with its surgical gesture labels has been the foundation of many advancements in surgical gesture recognition [8], surgical process modeling [52], skill assessment [12, 13], error detection [16–18], and autonomy [86]. Both supervised [52, 70, 73, 87–89] and unsupervised [90–94] approaches to gesture segmentation and classification have been developed. Also, previous work has shown that models trained for multilevel granularity recognition perform better than models for action recognition alone [51], so incorporating knowledge of higher surgical process levels can improve finer-grained recognition. However, these approaches either rely on black box deep learning models that are hard to verify and need extensive training data, or do not capture the human interpretable contextual information of the gestures. In addition, previous works and comparisons between them have been restricted by datasets with differing gesture definitions [8] and limited diversity in the numbers of subjects, trials, and *tasks*.

Furthermore, while gesture recognition has been done with kinematic and/or video data [8], recent work on action triplet recognition has mainly focused on video data of surgical procedures [49,78]. But, kinematic data is important for improved recognition accuracy using multimodal data [61, 75] and for error detection [8, 18], since it is unaffected by common camera issues such as occlusions, lens contamination, and smoke [17,30,31]. Thus, fine-grained surgical activity recognition using only kinematic data should be investigated.

In this chapter, we use the framework and COMPASS dataset from Chapter 2 to develop models for the automated inference of surgical context from video data and the recognition of surgical motion primitives and gestures from kinematic data with improved explainability and generalizability. Our method leverages improvements in image segmentation models to automatically detect surgical tool and object interactions and infer context. We develop knowledge-based and data-driven models for translating this context to gestures in order to relate these tool-object interactions to surgical workflow, and we compare the performance of our proposed pipeline to existing methods for unsupervised gesture recognition. These translations could also be used to facilitate dataset aggregation and check labeling accuracy. With the aggregated data from a variety of tasks in COMPASS, we propose a novel cross validation method that evaluates the generalizability of activity recognition models to unseen tasks. We implement a state-of-the-art surgical activity recognition model created by [70] and evaluate it as a case study using our novel cross validation method to quantify how well an activity recognition model will generalize to an unseen task when it is deployed. This enables us to perform the first evaluation of an activity recognition model trained on data from multiple tasks and datasets and show the limitations of using the current gesture definitions compared to motion primitives. We find that finer-grained models exhibit greater generalizability across tasks which motivates us to study the execution of lower-level surgical activities in these tasks.

This chapter makes the following contributions:

- We present a method for the automated inference of surgical context based on detecting important surgical tool and object interactions using image segmentation.

- We propose two methods for the automated translation of context labels to gesture labels based on a knowledge-based finite state machine and a data-driven machine learning model.

- We use the JIGSAWS dataset as a case study to demonstrate that our proposed approach results in shorter labeling time using the segmentation masks.

- We compare the performance of existing activity recognition models in a case study of TCNs (Temporal Convolutional Networks) using only kinematic data at different levels of the surgical hierarchy, specifically the gesture and motion primitive levels, and for separate left and right sides of the robot vs. both sides combined.
- We introduce the Leave-One-Task-Out (LOTO) cross validation method to measure the ability of surgical activity recognition models to generalize to an unseen task, since current datasets do not include all of the surgical tasks that a model may see when it is deployed.
- We perform the first evaluation of a surgical activity recognition model trained on multiple tasks with data combined from different datasets by comparing model performance using the existing LOUO method as well as our proposed LOTO cross validation method.

This chapter is organized as follows. Section 3.2 describes our methods for the automated inference of context and its translation to gestures. Section 3.3 evaluates the performance of surgical activity recognition models at different levels of the surgical hierarchy and their generalizability across tasks. Section 3.4 reviews related work on surgical scene and workflow segmentation. Section 3.5 gives the conclusion for this chapter.

3.2 Automated Context Inference

Our goal is to develop an automated, independent, and explainable way of generating gesture transcripts based on video data that does not rely on expensive training data for gestures. Such a method would be easier to verify by humans and experts and can be used as the ground truth for evaluating the black box gesture recognition models that directly detect gestures from kinematic data.

Previous works defined context in terms of the current surgical task or procedure [17, 18] and performed context-aware error detection by using different error detection models for different tasks and gestures. However, they did not explicitly detect context as defined in Chapter 2 and their definitions of context did not capture fine-grained activities or the interactions between the surgical instruments and objects in the environment. The interactions encoded in our definition of context could be used to detect some types of errors in robotic surgery such as a tool coming into contact with an inappropriate anatomical structure or accidentally dropping a needle. In this way, error detection models could be improved by including context information which necessitates the development of a method for the automated inference of surgical context.

Furthermore, despite the limited availability of datasets that include kinematic data from surgical robots, datasets for instrument and object segmentation in minimally invasive procedures are plentiful and have been the subject of image segmentation competitions [19, 28]. Unlike annotations for surgical instrument segmentation, annotations for surgical workflow such as gestures require guidance from surgeons [24]. Labeling using descriptive gesture definitions is tedious and subjective, leaving uncertainty as to exactly when gestures start and end, and gesture labels can have annotation errors, such as those noted in [25], that adversely impact machine learning models and analyses [8]. Recent studies using the JIGSAWS dataset have found errors in the gesture labels including a dozen amendments listed in [25] for the Suturing task. Thus, we propose methods that leverage the abundance of data with image annotations for surgical instruments and important surgical objects to address the challenges of manual labeling and relate surgical context to gestures. We aim to integrate data-driven segmentation with knowledge-based context inference and context to gesture translation to perform gesture recognition. This can enable improvements in gesture recognition by integrating human input.

In order to automatically generate context labels, we not only need to segment and identify both tools and objects, but we also need to determine if the identified objects are in contact with or held by the surgical instruments. Figure 3.1 shows our proposed approach for automatically inferring context where binary segmentation models are used to generate masks for the objects relevant to the task, and the intersections and overlaps of the contours of these masks are used to deduce the context state variables. We use multiple binary segmentation models instead of a single multi-class segmentation model in order to maximize performance in identifying each specific object class, and so that we can calculate the intersections between the object masks and thus infer their interactions. The automated context inference step takes the predicted masks from the DeepLab V3 binary segmentation models and first extracts the contours of the masks to reduce noise and computation time. Then, the distances, D(), and intersections, Inter(), between tool (e.g., Left Grasper "LG" and Right Grasper "RG") and object masks (e.g., Needle "N" and Thread "T" and Ring/Tissue Points "Ts"), and the calculated open (α) or closed ($\neg\alpha$) pose of the grasper jaws are used to label context according to Equations 3.1, 3.2, 3.3, 3.4, and 3.5.

3.2.1 Methods

This section presents our overall pipeline for the automated inference of surgical context and translation to gesture labels based on video data as shown in Figure 3.1. Surgical context can be inferred from video or kinematic data by estimating the values of the state variables. In this work, we specifically focus on context inference based solely on video data as an independent method of verifying gestures predicted from kinematic data or when kinematic data is not available. Our methods are presented for a case study of the JIGSAWS dataset [43] using the context labels from the COMPASS dataset in Chapter 2, but are applicable to other datasets and sets of gestures.

Tool and Object Segmentation

The detection of general and task-specific state variables for surgical context requires identifying the status and relative distances of the instruments and the objects of interest in a task. As shown in Figure 3.2a for the JIGSAWS tasks, these include the left and right graspers, needle, thread, and rings.

We modified the DeepLab V3 model [95] to perform binary segmentation and classify the background vs. one object class in the video frames of a task trial. Specifically,

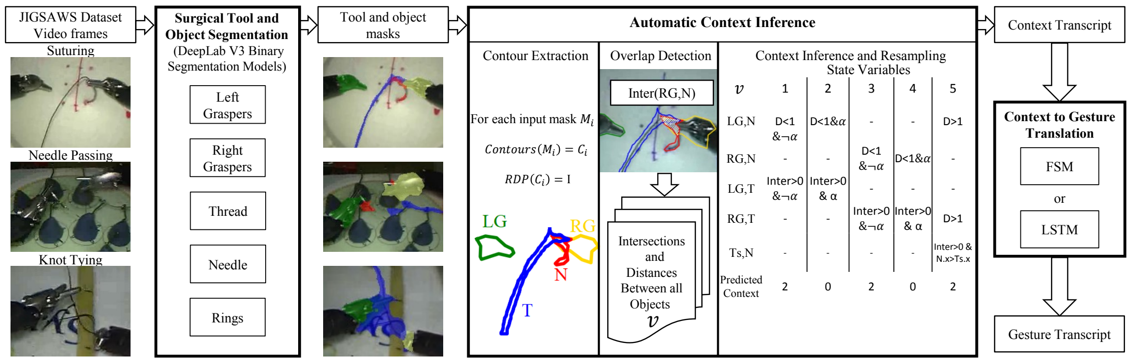
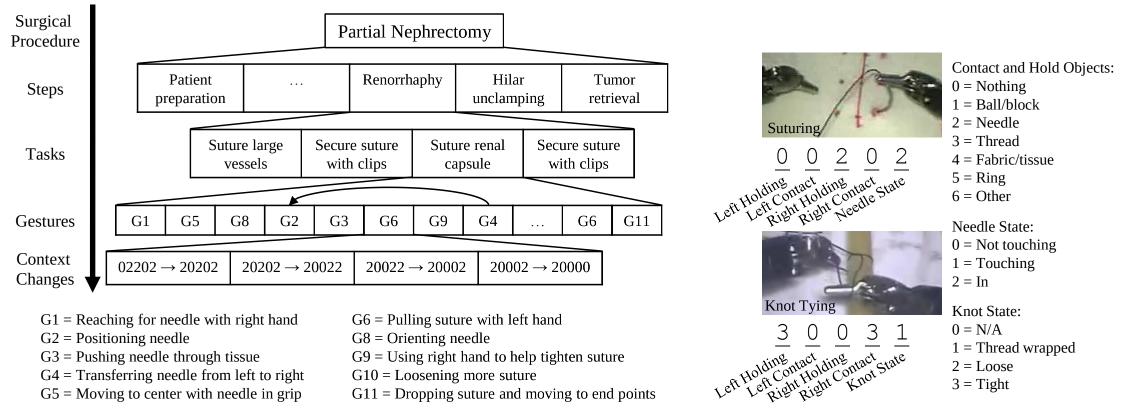

JIGSAWS Dataset Video frames
Suturing
Needle Passing
Knot Tying
Surgical Tool and Object Segmentation
(DeepLab V3 Binary Segmentation Models)
Left Graspers
Right Graspers
Thread
Needle
Rings
Tool and object masks
Automatic Context Inference
Contour Extraction
For each input mask M_i
$Contours(M_i) = C_i$
$RDP(C_i) = I$
LG
RG
N
T
Overlap Detection
Inter(RG,N)
Intersections and Distances Between all Objects v
Context Inference and Resampling State Variables
v 1 2 3 4 5
LG,N D<1 &¬α D<1&α - - D>1
RG,N - - D<1 &¬α D<1&α -
LG,T Inter>0 &¬α Inter>0 & α - - -
RG,T - - Inter>0 &¬α Inter>0 & α D>1
Ts,N - - - - Inter>0 & N.x>Ts.x
Predicted Context 2 0 2 0 2
Context Transcript
Context to Gesture Translation
FSM
or
LSTM
Gesture Transcript

Figure 3.1: Pipeline for automated context inference from video data and translation of context to gestures. D() is the distance between objects, Inter() is the intersection between two objects, α is the gripper open state, and $\neg\alpha$ is the gripper closed state.



(a) (b)

Figure 3.2: (a) Surgical hierarchy and relation between gestures and context in a suturing task. (b) State variables and object encodings that comprise context for the JIGSAWS tasks (see Figure 3.1). In the Suturing and Needle Passing tasks, a needle is used to throw four sutures through the fabric and rings, respectively, while in the Knot Tying task, two knots are tied around a piece of tubing.

we train separate binary classification models to classify background vs. left grasper, background vs. right grasper, background vs. needle, background vs. thread, and background vs. ring. The input to each model is a matrix $A_{H\times W\times 3}$ representing an RGB image of a video frame with height (H) and width (W). The output is a binary matrix $M_{H\times W}$ representing the segmentation mask with 0 for the background class, and 1 for the segmented object class. We then use the intersections between objects to infer context, which cannot be done with existing multi-class segmentation models that classify each pixel to a single object class. Binary segmentation models for each object class enable the analysis of intersections and overlaps among separate object masks to infer interactions between tools and objects.

For each object, we combine the data from all tasks to train a single model to classify that object in all tasks. We leveraged transfer learning by initializing the model with a ResNet-50 [96] backbone pre-trained on the COCO dataset [97]. We obtained tool and object annotations for the JIGSAWS dataset and used a subset of 70 videos to fine-tune the model. However, this significantly limited the test set for the whole pipeline since most of the data from JIGSAWS was needed to train the image segmentation models. We trained our models for up to 20 epochs using Adam optimization [98] with a learning rate of 10^{-5}.

Automated Context Inference

The masks from the segmentation models provide information about the areas and positions of the instruments and objects that can enable state variable estimation for each frame. By calculating intersections and distances between the object masks in a given frame, we can detect interactions such as *contact* and *hold* as shown in Figure 3.2b.

In the mask matrices $M_{H \times W}$ generated by the segmentation models, each element $m_{hw} \in \{0, 1\}$ indicates if the pixel (h, w) belongs to an object mask. We first perform a preprocessing step on M to eliminate noise around the masks such as the needles and threads. As shown in Figure 3.1, these masks have the worst segmentation performance due to their slender shape compared to the rounder and larger graspers. Contour extraction is performed to help eliminate the rough edges of the masks and improve intersection detection. This step uses the OpenCV library [99] to iteratively construct contours around every element $m_{hw} \in M$, thus reducing the input matrix to a list of points $p \in C \subset M$ for each tool or object class where C is the boundary of M. Using simplified polygons instead of binary masks greatly reduces the time required to calculate intersections and distances between objects for each frame. The best instrument masks tend to result in a single clearly defined polygon such as for the left and right graspers, while the needle and thread masks often result in small polygons well outside of the actual instrument position. We experimentally determined that dropping polygons with areas under 15 pixel units squared and smoothing the polygons using the Ramer-Douglas-Peucker (RDP) algorithm [100, 101] resulted in the best performance based on the training set.

Next, we detect overlaps between masks by taking a list of valid polygons and calculating a feature vector, v, of distances, D(), and intersection areas, Inter(), between pairs of input masks. The input polygons Left Grasper (LG), Right Grasper (RG), and Thread (T) are common for all tasks. Task-specific objects are the Needle (N) appearing in the Suturing and Needle Passing tasks, the manually labeled Tissue Points (Ts) representing the markings on the tissue where the needle is inserted in Suturing, and the Rings (R) in Needle Passing.

We define the distance functions $\mathrm{D}(I, J)$ and $\mathrm{d}(i, j)$ and the intersection function $\mathrm{Inter}(I, J)$ to, respectively, calculate the pixel distance between two object masks I and J, the pixel distance between the individual polygons $i_1, j_1, \ldots$ that constitute an object mask, and the area of intersection between two object masks I and J. For any object polygon I comprised of several polygon segments $i_1, i_2, \ldots, i_n$, the distance to any other object J can be calculated as: $\mathrm{D}(I, J) = \mathrm{average}([\mathrm{d}(i, j) \text{ for } i \in I \text{ and } j \in J])$. The intersection function, $\mathrm{Inter}(I, J)$, is implemented using a geometric intersection algorithm from the Shapely [102] library. We also define the components $I.x, I.y$ for an object I as the horizontal and vertical coordinates of the midpoint of polygon I, calculated as the average of every point in I. In order to determine the Boolean function (α) for each grasper, if the distance between the manually labeled pixel coordinates of the grasper jaw ends was less than 18 pixels, then the grasper was closed ($\neg\alpha$), else it was open (α).

The feature vector $v = <\mathrm{D}(LG, N),\ \mathrm{Inter}(LG, T), \cdots>$ (see Figure 3.1) is then used to estimate the values of different state variables using a set of task-specific functions. An example set of functions is shown in Equations 3.1-3.5 for the state variables relating to the left and right robot arms and needle in the Suturing task. As an example, if the distance between the left grasper and needle is less than one pixel ($\mathrm{D}(LG, N) < 1$) and the grasper is closed ($\neg\alpha$), then a value of 2 will be estimated for the Left Hold state variable. Or the Needle State is detected as “touching” (2) when the relative horizontal distance of the needle polygon ($N.x$) is less than the average (midpoint) of the tissue points ($Ts.x$) and these two objects intersect ($\text{Inter} > 0$).

The input sample rate of the context to gesture translation was 3 Hz, so the final estimated variables were downsampled from 30 Hz to 3 Hz using a rolling mode for each state variable with a window of 10 frames.

$$\text{Left Hold}\begin{cases} 2 & \text{if } \mathrm{D}(LG, N) < 1 \wedge \neg\alpha \\ 3 & \text{if } \mathrm{Inter}(LG, T) > 0 \wedge \neg\alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.1}$$

$$\text{Left Contact}\begin{cases} 2 & \text{if } \mathrm{D}(LG, N) < 1 \wedge \alpha \\ 3 & \text{if } \mathrm{Inter}(LG, T) > 0 \wedge \alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

$$\text{Right Hold}\begin{cases} 2 & \text{if } \mathrm{D}(RG, N) < 1 \wedge \neg\alpha \\ 3 & \text{if } \mathrm{Inter}(RG, T) < 1 \wedge \neg\alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

$$\text{Right Contact}\begin{cases} 2 & \text{if } \mathrm{D}(RG, N) < 1 \wedge \alpha \\ 3 & \text{if } \mathrm{Inter}(RG, T) < 1 \wedge \alpha \\ 0 & \text{otherwise} \end{cases} \tag{3.4}$$

$$\text{Needle}\begin{cases} 2 & \text{if } (\mathrm{Inter}(Ts, N) > 0 \wedge N.x < Ts.x) \\ 1 & \text{if } (\mathrm{Inter}(Ts, N) = 0 \vee N.x \geq Ts.x) \wedge \\ & (\mathrm{D}(RG, T) > 1 \vee \mathrm{D}(LG, N) > 1) \\ 0 & \text{otherwise} \end{cases} \tag{3.5}$$

Context to Gesture Translation

The last step in the pipeline translates the inferred context labels into gesture labels. We develop two models for relating our proposed context labels to existing gesture definitions by mapping each time-series context label to a gesture label. The first model is a knowledge-based Finite State Machine (FSM) and the second is a data-driven Long-Short Term Memory (LSTM) model.

The input to these translation models is a 2-dimensional time-series matrix $\chi_{State \times n}$, where $State$ represents the five state variables describing the context (see Figure 2.2) and n represents the total number of samples in the trial. The state at each time step, $State_t$, is mapped to a corresponding gesture, G_i, in the JIGSAWS dataset. The output of the translation is a 1-dimensional time-series array $Y_n \in \{\mathbb{G}\}$ with each time step mapped to a JIGSAWS gesture. We present two approaches based on domain knowledge and data.

Finite State Machine Model Our first approach relies on a finite state machine (FSM) defined based on knowledge of the surgical tasks which directly relates context to gestures and is more explainable than deep learning models. The grammar graphs from [52] for each task were overlaid on top of the ideal context models from Section 2.3.2 so that each gesture could be mapped into the groups of contextual changes that happen as the result of executing the gesture as shown in Figure 3.3 for the Suturing task. For example, G2 (positioning needle) would correspond to a change in the fifth state variable from a "0" to a "1". Or G4 (transferring needle from left to right) is the context sequence 20000 $\rightarrow$ 20020 $\rightarrow$ 20200 $\rightarrow$ 02200 $\rightarrow$ 00200 which means the needle is initially held in the left grasper, and then touched and grasped by the right grasper, and released by the left grasper. In Figure 3.3, the G4 and G8 groupings overlap since G8 (orienting needle) is performed by passing the needle from the right grasper to the left grasper and back to the right grasper while changing its orientation.

Given the context transcript of a trial, the FSM is evaluated for each context and a transition to the next gesture is detected if the input context is part of the next gesture. The FSMs for each task were initialized in the "Start" state since not all of

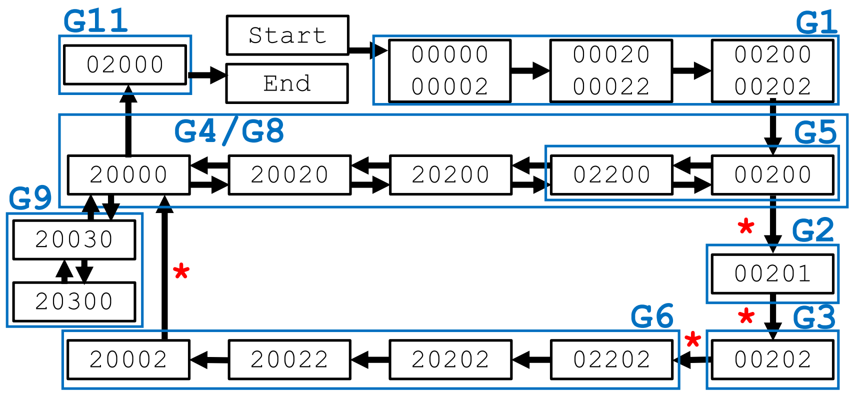


Figure 3.3: Grouping and mapping of context to gestures in the grammar graph of the Suturing task. * denotes transitions due to duration limits as follows: G2 > 6.0 s $\rightarrow$ G3, G3 > 11.1 s $\rightarrow$ G6, G4 > 5.2 s $\rightarrow$ G2, G6 > 6.1 s $\rightarrow$ G4.

the trials started with G1. Also, G11 was assumed to be last and so it was appended to the gestures following the last detected gesture after the cycles through the gesture grammar graph. In addition, in the Suturing and Needle Passing tasks, G9 and G10 had low rates of occurrence and were not included in the final translation. This allowed us to focus only on state changes involving the needle and thus ignore grasps and touches of the thread and rings with the added benefit of simplifying the FSMs and limiting the total number of valid context changes.

We also consider gesture duration as a trigger for transitions between gestures. If the current gesture's duration exceeds a certain threshold based on the average duration of that gesture class, a transition to the next gesture is enforced. This is to address the cases where a transition between gestures does not happen due to inaccuracies in context detection. For example, the segmentation models tend to have lower accuracy in detecting the needle and thread states, leading to undetected transitions that are dependent on those states.

LSTM Model Our second approach for the translation of context to gesture transcripts relies on sequential deep learning methods to learn relationships in the data that are not captured by the FSM models. We chose to use an LSTM model because it is a relatively simple network with the ability to learn temporal features.

We trained an LSTM based on the model from [70] to perform automated context to gesture translation for each task. Specifically, we used a simple double layer LSTM network with 64 hidden units for the Suturing and Needle Passing tasks and 256 hidden units for the Knot Tying task. We used Adam optimization [98] and the cross entropy loss function to train the models. The hidden layers, number of hidden units, and learning rates were determined by hyperparameter tuning. The final models were trained with the best model configurations and used to perform inference on the automatically generated context labels using the segmentation masks in the test set. Note that the LSTM model is a black box model and does not provide transparency like the FSM model in the previous section.

3.2.2 Evaluation

Experimental Setup

We use an 80/20 train/test split of the JIGSAWS dataset for evaluating our pipeline. The original videos are 30 Hz and we obtained binary masks for the tools and objects at 2 Hz which we then used to train and test the segmentation models. The LSTM networks are trained with the 3 Hz context labels from Chapter 2. We evaluate both the FSM and LSTM for context to gesture translation with the test set context labels.

The experiments were conducted on a PC with an Intel Core i7 CPU@3.60 GHz, 32 GB RAM, and an NVIDIA GeForce RTX 2080 Ti GPU running Ubuntu 18.04.2 LTS.

Metrics

The following metrics were used to evaluate the pipeline.

Accuracy Accuracy is the ratio of samples with correct labels divided by the total number of samples in a trial.

Edit Score Edit score is calculated using Equation 3.6 from [70] by normalizing the Levenshtein edit distance, edit(G, P), which quantifies the number of insertions, deletions, and replacements needed to transform the sequence of predicted labels, P, to match the ground truth sequence of labels, G. This is normalized by the maximum length of the sequences so that a higher edit score is better.

$$\text{Edit Score} = \left(1 - \frac{\text{edit}(G, P)}{\max(\text{len}(G), \text{len}(P))}\right) \times 100 \tag{3.6}$$

Intersection Over Union (IOU) Mean Intersection Over Union (IOU) as calculated in Equation 3.7 is a standard metric for assessing segmentation and translation models [97] where TP is the number of true positives, FP is the number of false positives, and FN is the number of false negatives.

$$\text{IOU} = \frac{TP}{TP + FP + FN} \tag{3.7}$$

When comparing labels, each predicted segment is matched to a corresponding segment in the ground truth labels. Then, the average IOU for each class is calculated and the mean of these class IOUs is returned.

Tool and Object Segmentation

Table 3.1 shows the performance of our segmentation models in comparison to related work. Although the MICCAI 18 challenge [19] dataset is from real porcine procedures and differs from the JIGSAWS dataset collected from dry lab experiments, it has similar objects including the clasper (similar to the graspers in JIGSAWS), needle, and thread. The Deeplab V3+ model achieved the best performance on the thread class. The top models from MICCAI 18 do not perform as well as our binary models on the needle and thread classes in the Suturing task. However, the Mobile-U-Net [103] achieved the highest performance for grasper and needle segmentation in the JIGSAWS Suturing task. [104] reported tool segmentation IOUs for all the JIGSAWS tasks with up to 0.8 for KT using a Trained LinkNet34, but did not do object segmentation. However, they do examine each task separately and report the best IOUs for different models for each task as: S: 69.48, NP: 66.10, and KT: 80.01. They also suggest that KT generally does better than S or NP because the contrast between the background and foreground is higher. Among the JIGSAWS tasks, we achieved the best performance in Suturing for the right grasper, needle, and thread, while the model performance on the Needle Passing task was the worst. This is likely due to the background of the Needle Passing videos having less contrast with the foreground compared to the other two tasks as shown in Figure 3.1. In the same figure, we can also see that the masks of the needle and thread are thinner compared to the masks of the grasper. Therefore, the mask boundary errors could contribute to a lower score for the needle and thread class than the grasper class. The estimated time for segmenting the whole JIGSAWS dataset using this approach is 8.6 hours.

Table 3.1: Tool and object segmentation performance on the test set (mean IOU for each object class) on the MICCAI 18 (M) and JIGSAWS Suturing (S), Needle Passing (NP), and Knot Tying (KT) tasks.

Model	Data	Graspers		Objects		
		Left	Right	Needle	Thread	Ring
Deeplab V3+ [19]	M [19]	0.78		0.014	**0.48**	N/A
U-net [19]		0.72		0.02	0.33	N/A
Mobile-U-Net [103]	S	**0.82**		**0.56**	N/A	N/A
Trained UNet [104]	S	0.69		N/A	N/A	N/A
	NP	0.66		N/A	N/A	N/A
Trained LinkNet34	KT	0.80		N/A	N/A	N/A
Deeplab V3 (ours)	S	0.71	**0.64**	**0.19**	**0.52**	N/A
	NP	0.61	0.49	0.09	0.25	**0.37**
	KT	**0.74**	0.61	N/A	0.44	N/A

Automated Context Inference

Tables 3.2 and 3.3 show the performance of the context inference method in terms of IOU achieved for each state variable with the predicted segmentation masks and the ground truth masks from crowdsourcing.

Table 3.2 shows that Left and Right Contact have higher IOUs compared to Left and Right Hold, and the Needle or Knot State has the lowest IOU. This is because errors in estimating the position of the grasper jaw ends affect the accurate inference of the hold state, while contact is relatively simple to infer by finding if the two masks intersect. Better performance in detecting Left and Right Contact compared to Left and Right Hold is also observed in Table 3.3 where state detection performance is shown with ground truth segmentation masks. Hence, the lower performance for Left Hold and Right Hold could primarily be due to the difficulty in detecting these states.

For the Needle/Knot State, we need to detect if the needle is in the fabric or tissue for the Suturing task, in or out of the ring for the Needle Passing task, and if the knot is loose or tight in the Knot Tying task. Detecting the state of the needle and knot is difficult even with the ground truth segmentation masks as shown in Table 3.3. This is because the needle and thread have the lowest segmentation performance compared to graspers as shown in Table 3.1 which could worsen the performance of the context inference. The total time to perform automated context inference is estimated to be about 30 seconds for the whole JIGSAWS dataset.

Table 3.2: State variable IOUs with consensus context using predicted masks from DeepLab V3.

Task	Left Hold	Left Contact	Right Hold	Right Contact	Needle or Knot	Avg
Suturing	0.48	0.75	**0.60**	0.87	0.30	0.60
Needle Passing	0.40	**0.97**	0.18	**0.95**	**0.39**	0.58
Knot Tying	**0.75**	0.72	0.57	0.78	**0.59**	**0.68**
Avg	0.54	**0.81**	0.45	**0.87**	0.43	

Table 3.3: State variable IOUs with consensus context using ground truth masks.

Task	Left Hold	Left Contact	Right Hold	Right Contact	Needle or Knot	Avg
Suturing	0.52	0.77	**0.61**	0.87	0.39	0.63
Needle Passing	0.42	**0.97**	0.19	**0.94**	**0.41**	0.59
Knot Tying	**0.83**	0.77	**0.61**	0.79	**0.62**	**0.72**
Avg	0.59	**0.84**	0.47	**0.87**	0.47	

Context to Gesture Translation

Table 3.4 shows the performance of the FSM and LSTM methods in translating the ground truth context labels from Chapter 2 to gestures. Between these two methods, the FSM achieves higher accuracies and edit scores than the LSTM. The performance of both models is degraded when automated context labels are used as input. In a follow-up work, [105] trained a more complex, Transformer model for gesture recognition that used consensus context labels as input and achieved higher accuracy, but lower edit score compared to the FSM for the Suturing task.

Table 3.4: Performance of models translating consensus context labels to gestures.

Task	Model	Accuracy (%)	Edit Score	IOU
Suturing	FSM	**66.3**	**84.4**	**0.48**
	LSTM	38.8	34.7	0.26
	Transformer [105]	74	71	
Needle Passing	FSM	**70.1**	**88.7**	**0.54**
	LSTM	17.0	20.0	0.04
Knot Tying	FSM	**54.6**	**91.5**	**0.43**
	LSTM	50.8	49.2	0.28

Results for the overall pipeline are shown in Table 3.5 where we observe that using automated context inferred from predicted masks degrades the overall performance because the tool and object segmentation models perform poorly at generating masks for the needle and for all tools and objects in the Needle Passing task. This effect is propagated through the automated context inference and the context to gesture translation resulting in low accuracies and IOUs. The FSM generally outperforms the LSTM likely due to its knowledge-based structure and limits on gesture durations that prevent the model from becoming stuck in any one gesture even when given degraded context labels. The FSM pipeline achieves accuracies lower than unsupervised models from [93] and [94] for Suturing, but outperforms them in terms of edit score. These observations suggest that there are benefits to incorporating knowledge into the context to gesture translation that can make the model more robust to degraded context labels. However, the FSM is manually developed based on domain knowledge and relies on defined inputs and transitions while the LSTM requires labeled data for training. The time to perform gesture translation from context for the entire JIGSAWS dataset is less than 3 minutes for both models.

Table 3.5: Pipeline performance translating automatically inferred context, given masks from the Deeplab V3 models, to gestures.

Task	Model	Accuracy (%)	Edit Score	IOU
Suturing	ST-CNN [87]	**77.7**	**68.0**	
Suturing (vid)	Zero-shot [93]	56.6	61.7	
Suturing (kin)	TSSC-DL [94]	49.7	32.8	
Suturing	FSM	**40.8**	**67.1**	**0.28**
	LSTM	25.4	24.9	0.17
Needle Passing	FSM	**18.0**	**76.2**	**0.12**
	LSTM	14.8	20.1	0.02
Knot Tying	FSM	**42.9**	**70.7**	**0.43**
	LSTM	36.5	23.8	0.17

3.3 Surgical Activity Recognition

In robot-assisted surgery (RAS), modeling and analysis at the gesture and action levels of the surgical hierarchy [9, 10] is performed to gain a better understanding of surgical activity and improve skill assessment [12, 13], error detection [17, 18], and autonomy [86]. Towards these applications, the automated segmentation and classification of surgical workflow has been an active area of research [50]. [8] and Section 3.4.2 provide comprehensive summaries of recent works at the gesture and action levels. However, previous works and comparisons among them have been restricted by differing gesture definitions [8] and limited diversity in the numbers of subjects, trials, and *tasks* across the existing datasets.

Recent works in gesture recognition have each defined their own sets of gestures for their own datasets [44, 53, 59, 73, 74] with limited overlap between gesture definitions.

On the other hand, works on the recognition of fine-grained surgical actions focus on action triplets (verb, instrument, tissue/object) [49, 55, 78], representing surgical instrument and tissue interactions in endoscopic videos. While gesture recognition has been done with kinematic and/or video data [8], recent work on action triplet recognition has mainly focused on video data of surgical procedures [49,78]. To leverage finer-grained action recognition in safety monitoring and autonomy applications, we examine verb-only predictions based on kinematic data. Kinematic data is particularly important for safety analysis [16–18], error detection [8, 18], and improved recognition accuracy using multimodal data [61, 75], since it is unaffected by common camera issues such as occlusions, lens contamination, and smoke [17, 30, 31]. Plus, using fewer data types can reduce computational costs and enable real-time applications [106].

The evaluation of gesture recognition models has been performed using the Leave-One-Supertrial-Out (LOSO) and Leave-One-User-Out (LOUO) cross validation methods defined in [52] for the JIGSAWS dataset. The LOSO setup tests the generalizability of a model to an unseen trial by a known user while the LOUO setup tests the generalizability of a model to an unseen user. The LOUO cross validation setup is considered the gold standard for evaluating gesture recognition models because it represents the realistic situation where a deployed model will need to recognize gestures performed by a surgeon it was has not seen before. However, an even more realistic case would have a gesture recognition model predict on an unseen, but perhaps similar, task performed by an unknown user. Thus, a method of evaluating the performance of models in this scenario is needed.

Using the COMPASS dataset created in Chapter 2, we train surgical activity recognition models based on the Temporal Convolutional Network (TCN) from [70] for motion primitive (MP) and gesture recognition. Some of the tasks in the COMPASS dataset share similar objects and goals enabling their aggregation and comparison. So, the standardized labels in COMPASS can support the combined analysis of datasets and the aggregation of data from contextually similar tasks for improved activity recognition and error detection [8]. We then analyze the performance of the TCN across users, tasks, and datasets by using the standardized MP labels. Since we are primarily interested in the performance of the models with respect to different types of labels and the augmentation of the training set with data from multiple datasets, we use the existing Leave-One-User-Out (LOUO) cross validation setup and propose the Leave-One-Task-Out (LOTO) cross validation setup which provides insights into the data needed to train surgical activity recognition models.

3.3.1 Methods

This section presents our methods for the construction, training, and evaluation of the gesture and MP recognition models.

Data Preprocessing

We use the COMPASS dataset from Chapter 2 to train our surgical activity recognition models since it has kinematic data from da Vinci surgical robots for different dry lab tasks from multiple datasets. It contains kinematic and video data at 30 Hz for a total of six tasks from three different datasets as described in Table 3.6. Context and MP labels are present for all trials, but gesture labels are only available for trials in the JIGSAWS and DESK datasets. To generate separate left and right label sets, MPs performed by each arm of the robot are split into new transcripts. Also, an Idle MP is defined and used to fill the gaps created by the separation so that every kinematic sample has a label.

The input signal to the activity recognition model is the time-series kinematic data, x_t, and the output is a transcript of class labels, y_t, one for each time-series sample, where each class label is selected from the finite set of gestures or MPs. We experimented with different combinations of kinematic variables as inputs to the activity recognition models (while hyperparameter and cross validation settings were kept constant) and found that using only the position, linear velocity, and gripper angle kinematic variables resulted in the best performance. This is consistent with the best-performing gesture recognition models that relied on kinematic data as reported in [8]. The stride was 1, so there was no downsampling, and the kinematic data and gesture and MP labels were all at 30 Hz.

Surgical Activity Recognition Model

One of the fastest and best performing models that used only kinematic data for gesture recognition in [8] was the Temporal Convolutional Network (TCN). The TCN is also used as a component in more complex state-of-the-art models such as MA-TCN [61] and MRG-Net [107]. Thus, as a case study, we adopt the TCN model from [70] for activity recognition at both the gesture and MP levels. This model has an encoder-decoder structure, each consisting of three convolutional layers with pooling, channel normalization, and upsampling. As in [70], the kernel size is set to the average duration of the shortest activity class (e.g., gesture or MP), and the three layers have 32, 64, and 96 filters respectively. We used the cross-entropy loss function and Adam optimizer [98].

The learning rate and weight decay hyperparameters for all TCN models were selected based on a grid search of values by training on the JIGSAWS dataset with gesture labels for each cross validation setup. For LOUO models, the learning rate was 0.00005 and the weight decay was 0.0005. For LOTO models, the learning rate was 0.0001 and the weight decay was 0.001. These values were fixed for all models

Table 3.6: Number of subjects and trials and types of annotations for each task in the COMPASS dataset: Suturing (S), Needle Passing (NP), Knot Tying (KT), Peg Transfer (PT), Post and Sleeve (PaS), and Pea on a Peg (PoaP).

Dataset	JIGSAWS [43]			DESK [58]	ROSMA [45]	
Tasks	S	NP	KT	PT	PaS	PoaP
Trials	39	28	36	47	65	71
Subjects		8		8	12	
Gesture Labels		✓		✓		
MP Labels		✓		✓	✓	

of their respective cross validation setup to analyze the effect of different training and label sets on model performance.

We compare the performance of the TCN when trained with four different sets of labels: gestures, MPs for only the left side (Left MPs), MPs for only the right side (Right MPs), and MPs for both sides together (MPs).

Model Generalization

We evaluate the generalization of the activity recognition models to unseen users/subjects and surgical tasks using two cross validation setups: Leave-One-User-Out (LOUO) from [52] and our novel Leave-One-Task-Out (LOTO) which are introduced next.

Leave-One-User-Out (LOUO) LOUO is the standard cross validation setup for comparing gesture recognition models and is preferred over the Leave-One-Supertrial-Out (LOSO) method because it measures a model's ability to generalize to an unseen user which is expected of a deployed model [8]. Since tasks from different aggregated datasets in COMPASS do not share the same subjects, we extended the LOUO setup from JIGSAWS [52] to include the new subjects, resulting in a maximum of 28 folds (corresponding to 28 unique users) when the model was trained on data from all tasks and datasets.

Leave-One-Task-Out (LOTO) Existing datasets represent a limited number of trials, subjects, and tasks. This means that machine learning models trained on them will see subjects, trials, and *tasks* that could be very different when they are deployed. In order to assess a model's ability to generalize to an unseen task, we introduce the Leave-One-Task-Out (LOTO) cross validation method.

In the LOTO setup, all of the data for one *task* was held out as the test set while the model was trained on all of the data for a set of other tasks. Thus, the model would be tested on all the trials of all subjects from an unseen task. For an example fold, a model could be trained on NP, KT, PT, PaS, and PoaP and tested on S. This differs from the LOSO setup where a model would be tested on unseen *trials* from a known subject of a known task. Similar to the existing LOSO and LOUO setups, average accuracy and edit score across the folds can be reported and used to compare models. However, examining each fold's performance and considering the relationship and similarity between the tasks in the training and test set yields insights about the generalizability of the model to unseen tasks and the data needed to train a model.

Task Combination for Training

The unified set of finer-grained MP labels enables the aggregation of data from different tasks across datasets which can improve the diversity and size of training data and model generalization. On the other hand, gesture labels are specific to each dataset and only tasks with similar labels within that dataset can be combined. To evaluate the effect of label granularity on task-generalization, we use data from different combinations of tasks in the aggregated datasets for training models in the LOUO and LOTO setups. Using MPs, there were two combinations with similar context: S+NP="SNP" where both tasks have a task-specific needle state, and PT+PaS="PTPaS" where both tasks have a task-specific block state. Tasks could also be grouped together if they come from the same dataset: S+NP+KT="JIGSAWS" and PaS+PoaP="ROSMA". Combining all of the data to train a model was referred to as "All". With gestures, only the SNP and JIGSAWS combinations could be used. For LOTO, we also considered specific combinations of data that tested on one task but removed the contextually similar tasks (defined above) from the training set to assess the importance of augmenting the training set with data from similar tasks.

Evaluation Metrics

We use the following standard metrics to evaluate the gesture and MP recognition models.

Accuracy Given the lists of predicted and ground truth labels, the accuracy is the ratio of correctly classified samples divided by the total number of samples in a trial.

Edit Score We report the edit score as defined in [70]. The Levenshtein edit distance, $\text{edit}(G, P)$, is the number of insertions, deletions, and replacements needed to transform the sequence of predicted labels, P, to match the ground truth sequence of labels, G. Then, the edit score is obtained by normalizing the edit distance by the maximum length of the predicted and ground truth sequences as in Equation 3.8 (with 100 representing the best prediction performance):

$$\text{Edit Score} = \left(1 - \frac{\text{edit}(G, P)}{\max(\text{len}(G), \text{len}(P))}\right) \times 100 \tag{3.8}$$

Mean Average Precision (mAP) Average Precision (AP) for a class is the average precision weighted by the increase in recall between thresholds calculated using Scikit-learn [108]. Then, the mean Average Precision (mAP) is the average AP for all classes. Micro mAP is reported for each verb to account for class imbalance.

3.3.2 Evaluation

The experiments were performed on a computer with an Intel Core i9 CPU @ 3.60 GHz x 16 and 64 GB RAM, running Linux Ubuntu 18.04 LTS, and an NVIDIA GeForce RTX 2070 GPU running CUDA 10.2. The models were built and trained using Torch 1.10.1 [109].

Kinematic Variables

We experimented with different combinations of kinematic variables as inputs to the surgical activity recognition models, specifically for the JIGSAWS tasks under Leave-One-User-Out (LOUO) cross validation as defined in [52] since both gesture and MP labels were available for those tasks. Table 3.7 shows the performance of the MP and gesture recognition models for three combinations of kinematic variables: (i) position, linear velocity, and gripper angle; (ii) position, orientation, and gripper angle; and (iii) position, orientation, linear velocity, and gripper angle. We found that using only the position, linear velocity, and gripper angle resulted in the best performance.

This is consistent with the best-performing gesture recognition models that relied on kinematic data as reported in [8].

Table 3.7: Accuracies and edit scores for different combinations of variables for activity recognition in the JIGSAWS tasks under LOUO cross validation.

Variables	MPs		Gestures	
	Acc	Edit	Acc	Edit
Position, Orientation, Linear Velocity, and Gripper Angle	52.4	54.5	77.8	79.2
Position, Orientation, and Gripper Angle	50.8	54.3	76.3	77.9
Position, Linear Velocity, and Gripper Angle	**56.0**	**55.6**	**81.4**	**82.2**

Gesture vs. Motion Primitive Recognition

In this section we present the performance of TCN models in recognizing gestures and MPs in comparison to state-of-the-art models and with different combinations of data.

Tables 3.8 and 3.9 compare the accuracies and edit scores averaged over the folds of the LOUO setup for the TCN models trained to recognize gestures and MPs, respectively. Accuracies for two state-of-the-art models are also presented in Table 3.8 against which our TCN model performs comparably or better. The TCN performed best on S alone achieving an accuracy of 84.6% and an edit score of 87.7 which is also slightly better than the 79.6% accuracy and 85.8 edit score reported by [70] (not shown in Table 3.8) and comparable to the results of [61] for the TCN using only kinematic data.

Despite KT only sharing two similar gestures and having a different task-specific context state variable than the other two JIGSAWS tasks, the performance of the TCN on KT is comparable to its performance on S (accuracy of 84.4%, edit score of 85.4 vs. accuracy of 84.6%, edit score of 87.7). When data from multiple tasks is combined for the SNP and JIGSAWS models, the accuracies of the TCN models are only about the average of their performances on individual tasks while the edit score for the JIGSAWS model drops to 82.0 which is lower than any single task in that dataset. Thus, there does not appear to be much benefit to combining data

Table 3.8: Gesture recognition performance under the LOUO cross validation setup compared to state-of-the-art models using only kinematic data. Results for the state-of-the-art models were only available for the JIGSAWS tasks.

Tasks	Gestures			Baselines	
	Acc (%)	Edit Score	mAP	Acc (%)	Model
PT	73.5	83.8	80.7		
S	84.6	87.7	86.0	90.2	MS-RNN [110]
NP	78.4	85.2	86.4	75.3	SC-CRF [111]
KT	84.4	85.4	89.8	78.9	SC-CRF [111]
SNP	81.4	85.2	85.1		
JIGSAWS	80.9	82.0	85.7		

Table 3.9: MP recognition performance with different task combinations under the LOUO cross validation setup.

Tasks	MPs		Left MPs		Right MPs	
	Acc	Edit	Acc	Edit	Acc	Edit
S	52.6	58.5	**66.0**	**65.2**	60.3	61.8
NP	52.3	53.1	**64.7**	**60.0**	55.9	54.8
KT	62.9	58.0	**71.2**	**67.2**	64.6	59.9
SNP	55.2	56.2	**66.5**	**62.2**	59.5	61.1
JIGSAWS	55.8	55.3	**66.4**	**63.5**	61.7	60.1
PoaP	67.4	74.6	**79.6**	72.6	79.3	**74.7**
PaS	70.2	76.5	**80.0**	**77.6**	78.5	75.9
ROSMA	67.5	**74.9**	**78.8**	73.1	78.2	73.6
PT	75.3	79.9	81.1	81.8	**82.0**	**82.4**
PTPaS	70.3	76.4	78.5	**77.8**	**78.8**	77.4
All	65.9	69.6	**75.0**	70.3	73.1	**70.7**

from the JIGSAWS tasks at the gesture level. The PoaP and PaS tasks from the ROSMA dataset did not have gesture labels, so no gesture recognition models were trained for them. The PT task of the DESK dataset did have gesture labels although their definitions were much closer in scope to MPs rather than the more complex gestures of the JIGSAWS dataset. The TCN only achieves an accuracy of 73.5% for gesture recognition on the PT task which is comparably lower than the performance of any of the MP recognition models for this task in the LOUO setup shown in Table 3.9. For the JIGSAWS tasks, the gesture recognition models performed much better than MP recognition models (only considering verbs). This suggests that the definitions and granularity of the labels in the surgical hierarchy affect activity recognition performance.

By examining Table 3.9, we note that MP recognition performance is better for the task in the DESK dataset, and to a somewhat lesser extent for tasks in the ROSMA dataset, than for tasks in the JIGSAWS dataset. This could be because the JIGSAWS tasks (S, NP, KT) are more challenging with more complex grammar graphs [52], while the tasks in the ROSMA and DESK datasets are variations of a pick-and-place task with simpler grammar graphs. This is supported by the higher edit scores for the models trained on the ROSMA and DESK datasets compared to the models trained on the JIGSAWS dataset. Combining data at the MP level also resulted in performance metrics that are about the average of the individual tasks that were combined. But, training separate models for each side of the robot resulted in higher accuracies with comparable or better edit scores. So, having separate annotations and models for the left and right arms of the robot can improve MP recognition performance.

Furthermore, Table 3.10 shows the mAPs for each MP and micro average over all verbs for the MP recognition models in the LOUO setup. We note that class imbalance may have caused differences between the macro and micro mAPs for tasks from the DESK and ROSMA datasets where MPs with a greater number of instances sometimes had higher mAPs. None of these MP recognition models perform as well as the gesture recognition models for the JIGSAWS tasks as listed in Table 3.8, which achieve mAPs of up to 89.8. So, additional work is needed to improve fine-grained activity recognition performance. Although the recognition models of [49] have been evaluated for verb recognition performance, a direct comparison to action triplet models is not fair as the data (kinematic vs. video) and tasks (robotic dry lab vs. real laparoscopic surgery) are different.

Table 3.10: Number of examples (#) and mean average precisions (mAPs) of MPs for models trained on different combinations of tasks in the LOUO setup with micro mAP for all verbs (weighted by number of samples in each class).

Tasks	Grasp		Release		Touch		Untouch		Pull		Push		All verbs	
	#	mAP	#	mAP	#	mAP	#	mAP	#	mAP	#	mAP	#	mAP
S	471	57.6	441	48.7	518	58.1	314	27.6	194	72.2	179	55.1	2117	52.5
NP	373	63.0	365	57.0	330	57.0	206	16.2	114	69.1	119	34.2	1507	52.0
KT	283	64.5	247	69.1	135	43.8	111	18.6	235	85.3	0	N/A	1011	62.7
SNP	844	61.3	806	54.8	848	58.0	520	21.2	308	70.0	298	47.3	3624	52.9
JIGSAWS	1127	62.2	1053	58.7	983	53.0	631	20.7	543	72.6	298	41.5	4635	53.7
PoaP	577	52.8	556	55.3	1782	88.0	1261	47.2	525	58.3	2	33.5	4703	65.5
PaS	824	50.2	776	50.3	1598	88.9	1131	45.7	0	N/A	0	N/A	4329	63.3
ROSMA	1401	50.7	1332	53.1	3380	89.2	2392	45.3	525	59.2	2	5.1	9032	64.5
PT	323	48.3	313	61.1	539	90.3	364	68.3	0	N/A	0	N/A	1539	70.3
PTPaS	1147	48.7	1089	54.6	2137	89.8	1495	53.0	0	N/A	0	N/A	5868	65.9
All	2851	54.5	2698	55.4	4902	79.5	3387	43.5	1068	65.7	300	37.7	15206	60.7

Model Generalization

Table 3.11 reports the accuracies and edit scores for models trained with different combinations of data in the LOTO setup and immediately shows the limitations of existing gesture definitions. Note that only the JIGSAWS dataset had gesture labels that could be used in the LOTO setup, so gesture recognition models using tasks from different datasets could not be trained because gesture labels were not present (i.e., ROSMA) or were not compatible (i.e., JIGSAWS vs. DESK).

We observe that splitting the MP labels into separate transcripts and training separate models for the left and right arms of the robot generally resulted in improved accuracies compared to having a single model.

We find that a gesture recognition model trained on S or NP is able to transfer to NP or S, respectively, but when KT is added to the training set, performance is severely decreased. Specifically, a model tested on S drops from an accuracy of 48.5% to 24.4%, and a model tested on NP drops from 37.9% to 28.8% when KT is added to the training set. This is due to the lack of generalizable gesture labels between these tasks since S and NP have an almost completely different set of gestures compared to KT. Thus, gesture recognition for the KT task using a model trained

Table 3.11: MP and gesture recognition performance with different task combinations under the LOTO cross validation setup.

Test Set	Training Set						Gestures		MPs		Left MPs		Right MPs	
	(Task combinations)						Acc	Edit	Acc	Edit	Acc	Edit	Acc	Edit
S		NP	KT	PT	PaS	PoaP			39.0	**49.0**	62.3	**59.4**	42.9	**58.5**
S			KT	PT	PaS	PoaP			25.3	40.2	41.3	50.2	34.8	42.5
S		NP	KT				24.4	33.9	43.2	48.3	56.2	52.7	46.3	48.2
S		NP					**48.5**	**70.5**	**44.0**	47.7	**62.7**	58.7	**50.1**	54.7
NP	S		KT	PT	PaS	PoaP			40.8	48.5	**54.1**	**55.9**	41.6	46.4
NP			KT	PT	PaS	PoaP			35.6	44.5	46.2	51.9	34.4	39.9
NP	S		KT				28.8	38.2	37.2	46.9	49.9	52.8	44.7	48.3
NP	S						**37.9**	**52.7**	**42.2**	**48.6**	52.2	51.9	**46.0**	**52.8**
KT	S	NP		PT	PaS	PoaP			**33.3**	40.2	47.2	**51.9**	**35.2**	39.8
KT				PT	PaS	PoaP			22.6	37.5	37.7	36.5	25.1	36.8
KT	S	NP					**6.8**	**9.3**	29.7	**40.5**	**48.1**	50.4	34.5	**42.8**
PT	S	NP	KT		PaS	PoaP			**53.1**	**48.0**	**55.9**	**42.0**	43.9	38.6
PT	S	NP	KT			PoaP			44.5	44.4	49.0	37.6	**55.3**	**44.8**
PT					PaS				48.0	37.6	51.1	40.3	52.6	43.5
PaS	S	NP	KT	PT		PoaP			58.1	**65.5**	58.8	**60.5**	61.1	**58.0**
PaS	S	NP	KT			PoaP			60.7	65.0	58.5	58.5	61.4	57.7
PaS				PT		PoaP			58.0	64.1	**65.8**	58.3	**63.9**	57.2
PaS				PT					**61.0**	37.5	42.5	54.6	55.0	42.9
PaS						PoaP			58.4	62.9	59.5	57.2	59.8	56.1
PoaP	S	NP	KT	PT	PaS				**56.5**	**64.2**	**59.1**	**50.7**	**58.5**	**49.8**
PoaP	S	NP	KT	PT					53.4	47.8	50.4	45.9	36.0	43.9
PoaP					PaS				54.8	63.1	57.8	44.9	58.0	45.2

on S and NP is particularly poor with an accuracy of only 6.8%. Hence, at the gesture level, combining data from different tasks is not beneficial for a model that must predict on an unseen task.

Comparatively, when MPs are used, the model is able to predict on a new task like KT by leveraging information learned from other tasks that are dissimilar to it such as S and NP. Adding data from a dissimilar task has a much smaller detrimental effect at the MP level than at the gesture level. For example, the model's accuracy drops less than 1% for S and 5% for NP when KT is added to the training set.

When the model must predict MPs on a dissimilar task with a different task-specific context state, then combining data from all tasks results in better performance compared to using only data in the same dataset. KT improves from an accuracy of 29.7% to 33.3% and PoaP improves from 54.8% to 56.5% by including data from other datasets.

For S and NP, we observe that models trained with data from the same dataset and with the same task-specific state variable perform better than models including data from the same dataset but without the same task-specific state variable. However, the opposite is true for PaS where models whose training sets included PoaP (same dataset) but not PT (same task-specific state variable) sometimes performed better.

For KT and PoaP, even though data with the same context was not available, models whose training sets included tasks from the same dataset generally performed better than models whose training sets did not. The poorest performing models for PaS were trained with data that only included PT, even though they had the same task-specific state variables. For PT, some models that included PaS (same task-specific state variable) performed better than those that did not. Since tasks from the same dataset were performed by the same subjects, models whose training sets included tasks from the same dataset are tested on different tasks performed by known subjects. This is somewhat similar to the Leave-One-Supertrial Out (LOSO) cross validation method where models are tested on unseen trials performed by known subjects. Models evaluated using the LOSO method perform better than those using the LOUO method which suggests that including data from the same subjects may improve model performance. However, additional data and tests would be needed to determine if it is this or another feature of the dataset that is responsible for the better performance. Additional evaluations are also needed to verify that MPs enable task-generalization for other types of models such as transformers [106].

3.4 Related Work

3.4.1 Surgical Scene Segmentation

There are many models that focus on surgical instrument segmentation [112–115], as well as also identifying anatomy and objects [116] including the needle and suture [117–119]. Most existing works on robot instrument or surgical scene segmentation have been based on real surgery videos using publicly available datasets such as MICCAI EndoVis 17 [28], MICCAI EndoVis 18 [19], and Cata7 [29]. Popular frameworks include UNet [120], TernausNet [121], and LinkNet [122]. Various models have been proposed in the EndoVis challenges, but segmenting all objects in a surgical scene has been challenging. The DeepLab V3+ model [95] achieved the best overall performance in [19], and other DeepLab models [95, 123] have also shown promise in surgical tool and object segmentation. Surgical scene segmentation in dry lab settings with the JIGSAWS dataset has been done in [103] and [104], but we go further by segmenting additional objects and using tool and object segmentation for context inference. Thus, our method benefits from the availability of large open-source image segmentation datasets and could also improve segmentation performance by fine-tuning on smaller datasets.

3.4.2 Surgical Workflow Segmentation

Surgical workflow segmentation has been examined in different datasets with different tasks and at different levels of granularity as summarized in Table 3.12.

Datasets and Tasks

Some recent works in surgical activity recognition perform comparative evaluations of their models across different datasets. For example, [50] developed the TAPIR model and found that it performed better on the MISAW dataset than their PSI-AVA dataset for phase and step recognition in terms of mAP, but did not examine the reason for this. [18] evaluated an LSTM using LOSO cross validation on the

JIGSAWS dataset and their own dataset of Block Transfer on the Raven II. The LSTM achieved a higher accuracy for the Block Transfer task since it was a simpler task with a larger amount of data compared to the JIGSAWS tasks. However, these works did not combine data from multiple datasets or tasks since the label definitions differed. [16] found that combining data from the Suturing and Needle Passing tasks of the JIGSAWS dataset could improve the performance of executional error detection models because the gestures were kinematically similar. This suggests that there is a benefit to aggregating data for training models. Additionally, [124] found that gesture recognition models trained on the JIGSAWS dataset did not generalize well to other dry lab or clinical data.

Label Granularities

Surgical workflow recognition has been examined at different levels of granularity as listed in the fifth column of Table 3.12. Note that there are inconsistencies in label and granularity definitions across datasets. For example, the *tasks* of Suturing, Knot Tying, and Peg Transfer in JIGSAWS and DESK are considered *phases* in MISAW [51] and PETRAW [125]. [73] trained a GRU for gesture and maneuver recognition on the JIGSAWS and MISTIC-SL datasets, respectively. Although the datasets had different labels, the lower-level gesture recognition model had a higher error rate than the maneuver recognition model. The MISAW challenge [51] and HeiChole benchmark [20] datasets were labeled at multiple levels as well as the PSI-AVA dataset [50]. The best-performing models from these works all showed decreasing performance metrics for finer-grained labels which highlights a significant challenge for fine-grained recognition. Interestingly, [51] found that multi-granularity recognition models performed better than activity recognition models because the models may be learning that certain activities only occur during specific phases and steps. Also, recent works on action triplet recognition in laparoscopic procedures focus on concurrent phase, step, and action recognition [125]. The poor performance of activity recognition models is a barrier to clinical applications, but understanding the relationship between granularity levels can address this challenge and guide model development.

Table 3.12: Surgical workflow segmentation models that considered multiple datasets, data types, and label granularities.

Paper	Dataset	Data Type	Tasks	Label Levels	Best Teams/Models and Performance	
MISAW Challenge 2021 [51]	MISAW	Kinematics and/or Video	Anastomosis	Phases	MedAIR [107]	AD-Accuracy: 96.5%
				Steps	MedAIR [107]	AD-Accuracy: 84.0%
				Activities	NUSControl Lab and UniandesBCV	AD-Accuracy: ∼64%
				Multi-granularity	NUSControl Lab	AD-Accuracy: ∼72%
HeiChole benchmark 2021 [20]	EndoVis 2019	Video	Laparoscopic cholecystectomy	Phases	HIKVision and CUHK	F1 Score: ∼65%
				Actions	Wintegral	F1 Score: 23.3%
Valderrama 2022 [50]	PSI-AVA	Video	Radical prostatectomy	Phases	TAPIR	mAP: 56.6%
				Steps		mAP: 45.6%
				Actions		mAP: 23.6%
DiPietro 2019 [73]	JIGSAWS	Kinematics	Suturing	Gestures	GRU	Error rate: 15.2% Edit distance: 8.4
	MISTIC-SL		Knot Tying	Maneuvers		Error rate: 8.6% Edit distance: 9.3
Multimodal attention [61]	JIGSAWS	Kin+Vid	Suturing	Gestures	MA-TCN (Acausal)	Accuracy: 86.8% Edit: 91.4
	own (dV)					Accuracy: 80.9% Edit: 79.6
Gesture Recognition Survey [8]	JIGSAWS	Kinematics	Suturing	Gestures	MS-RNN [110]	Acc: 90.2% Edit Score*: 89.5
		Video			Symm dilation+attention [126]	Acc: 90.1% Edit Score: 89.9
		Kin+Vid			Fusion-KV [75]	Acc: 86.3% Edit Score: 87.2
PETRAW Challenge [125]	PETRAW	Video	Peg Transfer	Phases, Steps, and Activities	SK	AD-Accuracy: 90.8%
		Kinematics			MedAIR	AD-Accuracy: 90.7%
		Segmentation			SK	AD-Accuracy: 88.5%
		Vid+Kin			NCC NEXT	AD-Accuracy: 93.1%
		Vid+Kin+Seg			NCC NEXT	AD-Accuracy: 93.1%
Sim2Real Gesture Classification [44,58]	DESK	Kinematics	Peg Transfer	Gestures	RF	Simulator Acc: 86%
						Robot Acc: 95%
						Sim2Real (0% Real) Acc: 34%
						Sim2Real (18% Real) Acc: 85%
CholecTriplet2021 Challenge [21]	CholecT50	Video	Laparoscopic cholecystectomy	Action Triplets	Trequartista	AP_V: 52.9 AP_{IVT}: 38.1

* Normalized by maximum number of segments in any ground-truth sequence.

Robotic Systems

[44] and [58] examined gesture classification using transfer learning for the Peg Transfer task. Their random forest (RF) model was more accurate for trials performed on the Taurus robot compared to the dVRK, but the Taurus simulator had the lowest accuracy. Although the performance of the model for complete transfer from the Taurus simulator to the Taurus robot (0% Taurus robot data in the training set) resulted in poor performance, including a percentage of Taurus robot data in the training set allowed the model to achieve comparable performance to the Taurus simulator model. They found that models can transfer from simulated to real robots, or between real

robots, although the latter required a greater percentage of data from the target domain to achieve comparable performance. Thus, future work may be able to leverage transfer learning across robotic systems and/or tasks to improve model performance.

3.5 Conclusion

In this chapter, we presented methods for the automated inference of surgical context and translation to gestures based on video data, and the recognition of motion primitives and gestures from kinematic data. Models such as these are critical components in understanding the surgical scene and workflow, and support the development and implementation of features to improve the safety, security, and efficiency of robotic surgery.

First, the pipeline for the inference of surgical context and translation to gesture labels can perform automated and explainable gesture inference given video segmentation masks and key points. It integrated data-driven segmentation with knowledge-based context inference and context to gesture translation to perform gesture recognition. Compared with deep learning approaches to gesture recognition, our approach enables improvements by integrating human input. Furthermore, it can be used as an efficient and fast inference method by significantly shortening the manual gesture labeling time (~9 hours vs. ~26 hours for the case study of the JIGSAWS dataset). We rely on models pre-trained on general images and publicly available datasets which lowers the cost of manually labeling video data and makes our model generalizable to other datasets and tasks.

For the case study of the JIGSAWS dataset, our binary segmentation models achieve comparable performance to state-of-the-art models on the grasper and thread classes, and better performance on the needle class. However, they still do not perform well enough for the needle and thread classes which are important for accurate context inference, specifically for the task-specific state. Context inference also does not perform equally well for all of the states. Given the ground truth segmentation masks, it achieves ~85% IOU for states such as Left and Right Contact, but only

$\sim$45% IOU for the Needle/Knot State. The FSM and LSTM models for context to gesture translation have better performance given ground truth context labels compared to predicted context which may be due to imperfect model performance at each stage of the pipeline and error propagation. A more complex Transformer model is found to be comparably effective at translating ground truth context labels to gestures compared to the FSM for the Suturing task [105].

Limitations of this pipeline included the need for manual annotations of the grasper jaw end points and tissue entry/exit points. These gaps could be closed by implementing models such as those in [127] and [128] to obtain these points automatically from video data. Work such as [129] could also be used to improve the thread masks. In addition, this method relied on 2D images to infer context about a 3D environment. This can particularly complicate the detection of contact states since tools and objects that overlap in the 2D images may not necessarily be in contact. Future work will focus on addressing these limitations and improving the performance and robustness of the overall pipeline in order to apply it to runtime error detection [16, 18]. Additional data modalities and input features, as well as model architectures are also avenues of future research for generating gesture labels.

Second, we compared the performance of surgical activity recognition in a case study of TCN models at different levels of the surgical hierarchy, evaluated their generalizability to unseen users and tasks, and drew insights from the combination of tasks used to train these models. We found that gesture-level recognition models performed better than motion primitive-level recognition models under the LOUO cross validation method which is consistent with the observations of [51]. Our models achieved comparable or better accuracies than state-of-the-art models in recognizing gestures from JIGSAWS. Using motion primitives, we combined data from different datasets, tasks, and subjects and found that having separate models for the left and right sides improved performance.

We also introduced the Leave-One-Task-Out (LOTO) cross validation setup, and performed the first evaluation of a surgical activity recognition model in terms of its ability to generalize to an unseen task. When tested on a task from a specific dataset, the model performed better if data from other tasks in that dataset were included in the training set. Also, models for tasks with different task-specific state variables performed best when data from all other tasks were aggregated in their training sets. Similarly, [130] evaluated the performance of surgeme classification models in sim2real domain transfer using different data percentages in the target domain and found that this improved the accuracies of their models. Thus, improved performance may be achieved by including a small percentage of data from the target test task in the training data. Future work will focus on evaluating the task-generalization of other state-of-the-art recognition models (e.g., recurrent neural networks and transformers) using both kinematic and vision data as well as using other tasks and datasets.

The models and methods developed in this chapter can be utilized in providing context and activity-awareness in autonomous systems. An example of an autonomous camera arm for a surgical robot is described in Appendix B which would benefit from the integration of these models in further refining its movements and behaviors during surgical tasks.

Chapter 4

Surgical Error Analysis

4.1 Background

Safety during robotic surgery is entirely dependent on the surgeon since surgical robots are fully teleoperated and have no autonomy or haptic feedback. This leaves patients vulnerable to human errors [133–135] or faults in the robot-assisted surgery (RAS) system which can lead to cuts, bleeding, or collisions that may propagate into adverse events and cause complications during or after the procedure [136]. A review of adverse event reports in the FDA's MAUDE (Manufacturer and User Facility Device Experience) database for the da Vinci Surgical System found that 81% of events related to pre-existing health conditions or surgeon errors resulted in injury or death [33].

However, the sources of errors in RAS are diverse and domain-specific, including faults in the robotic system software and hardware, and human errors [136]. Error detection is an emerging subfield [8], but there has been limited attention to identifying executional errors made by surgeons during training and skill assessment. This is because errors are rare, leading to a lack of datasets that are labeled for errors. Previous works that do consider errors have done so in the context of activity-aware skill assessment where the errors are associated with a particular fine-grained unit

This chapter contains material from the works [131], [16], and [132], coauthored with Z. Li, L. A. Cantrell, N. Schenkman, K. Chen, and H. Alemzadeh; [16] copyrighted by IEEE.

of surgical activity such as gestures or motions. However, the surgical process and workflow are affected by suboptimal performance and safety-critical events [33]. Thus, error detection and analysis are important applications of the surgical workflow and activity recognition models developed in the previous chapters.

In addition, automated scoring and feedback on training exercises have focused on efficiency, safety, and task/procedure-specific metrics. But, they assess overall performance during a demonstration and are not descriptive enough to pinpoint inefficiencies or errors. In order to improve explainability, previous works have performed skill assessment at different levels of the surgical hierarchy. At the gesture level, [37] found that experts used fewer gestures, made fewer errors, and had more predictable transitions than novices. So, methods for automated skill assessment that leverage gestures, such as those listed in Table 4.14, use gestures as inputs to their models or by training models for specific gestures. Efforts towards increasing interpretability have thus focused on identifying problematic gestures that contribute to the low skill scores [137, 138], using data-driven models. But, there is a large variability in the correct performances of gestures due to style or expertise and it is impossible to define all errors or incorrect performances of gestures, thus limiting the effectiveness of supervised machine learning models trained on small datasets. At the motion level, skill assessment has focused on statistical measures such as time and path length [34, 38], and data-driven models such as HMMs [12, 13] which lack specificity. Motions learned from data in an unsupervised manner are not easily human interpretable and do not correspond with labeled units of surgical activity [35]. Providing gesture-specific feedback based on identifying patterns in finer-grained units of surgical activity has not been explored yet.

In this chapter, we use the surgical task and activity models developed in the previous chapters to create activity-aware methods for the identification and analysis of errors. We use these novel methods to identify interpretable patterns in fine-grained surgical activities that strongly correlate with measures of surgical expertise and can be used to provide specific and interpretable feedback to surgeons during training and skill assessment.

This chapter addresses the above challenges by making the following contributions:

- We propose a task- and gesture-specific rubric for the identification of executional and procedural errors using data collected from real or simulated surgical demonstrations.
- We apply this rubric to manually annotate video data of the Suturing and Needle Passing tasks to augment the JIGSAWS dataset with executional error labels.
- We develop quantitative analytical methods to characterize gesture-specific executional and procedural errors using pre-collected kinematic data and gesture labels.
- We provide insights on the types, frequencies, and durations of executional and procedural errors across tasks and gestures and their correlations with skill levels which can provide a basis for the design of automated error detection mechanisms.
- We analyze the relationship between surgical gestures and fine-grained motion primitives which shows that the sequence of motion primitives can help detect labeling errors in surgical gestures.
- We define and detect inverse motion primitives in the execution of surgical gestures and tasks. Inverse motion primitives are often used as recovery actions to correct the position or orientation of objects, or may be indicators of other issues such as poor depth perception.
- We analyze the types, frequencies, and durations of inverse motion primitives across gesture types and skill levels which shows that they strongly correlate with lower skill levels, and could be used to improve the interpretability of skill assessment by identifying the specific motions within a trial that contribute to lower scores.

This chapter is organized as follows. Section 4.2 describes our methods for identifying and analyzing gesture-specific executional and procedural errors. Section 4.3 identifies and examines inverse motion primitives and their categorization. Section 4.4 reviews related work on error detection, activity-aware skill assessment, and interpretable feedback. Section 4.5 gives the conclusion for this chapter.

4.2 Gesture-Level Identification and Analysis of Errors

Sources of errors in robot-assisted surgery are diverse and domain-specific, including faults in the robotic system software and hardware, or human errors [136]. We focus on errors in the execution of procedures that can occur at any level of the surgical hierarchy [10] and may propagate and cause errors at other levels as shown in Figure 4.1. In a surgical **operation** there may be multiple **procedures** which are divided into **steps**. Each **step** is subdivided into **tasks** comprised of **gestures** (also called sub-tasks or surgemes) which are made of **motions** such as moving an instrument or closing the graspers. We specifically focus on studying the quality of the task demonstrations at the **gesture** level.

To inform the development of error detection models, we present methods for analyzing executional and procedural errors and provide insights about key features that are indicative of errors. Figure 4.2 shows our overall method for the analysis of executional and procedural errors in the Suturing and Needle Passing tasks of the JIGSAWS dataset. We focus specifically on these two tasks because they share similar gestures and contain a sufficient number of demonstrations for comparisons between the two tasks. We analyze the kinematic data for erroneous and normal performances of each gesture since many gesture recognition models in [8] are based on kinematic data and this modality is robust to camera occlusions, lens contamination, smoke and other negative effects on video data [17, 30, 31].

We focus on answering the following research questions:

RQ 1: Which tasks and gestures are most prone to errors?

RQ 2: Are there common error modes or patterns across gestures and tasks?

RQ 3: Are erroneous gestures distinguishable from normal gestures?

RQ 4: What kinematic parameters can be used to distinguish between normal and erroneous gestures?

RQ 5: Do errors impact the duration of the trajectory?

RQ 6: Are there any correlations between errors and surgical skill levels?

4.2.1 Rubric for the Objective Assessment of Errors

Two types of human errors in laparoscopic surgery were defined by [133]: procedural errors and executional errors. **Procedural errors** involve "the omission or re-arrangement of correctly undertaken steps within the procedure" while **executional**

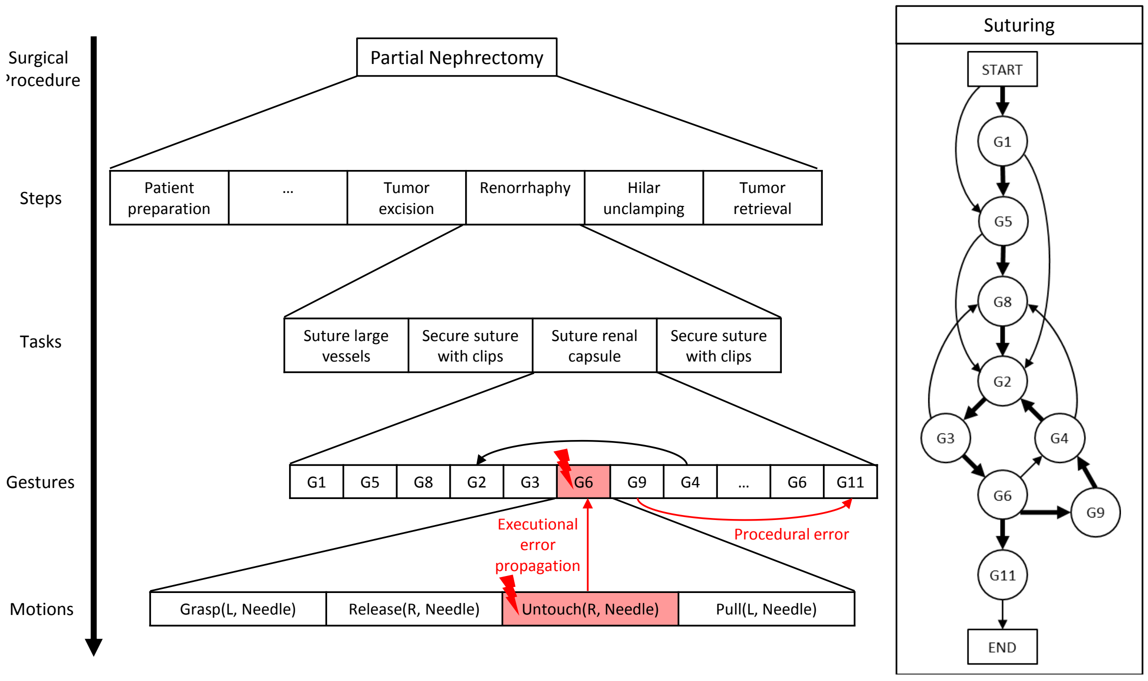


Figure 4.1: Surgical hierarchy (adopted from [10]) for an example urological procedure of partial nephrectomy (based on [139] and [140]) with example gesture-specific executional and procedural errors.

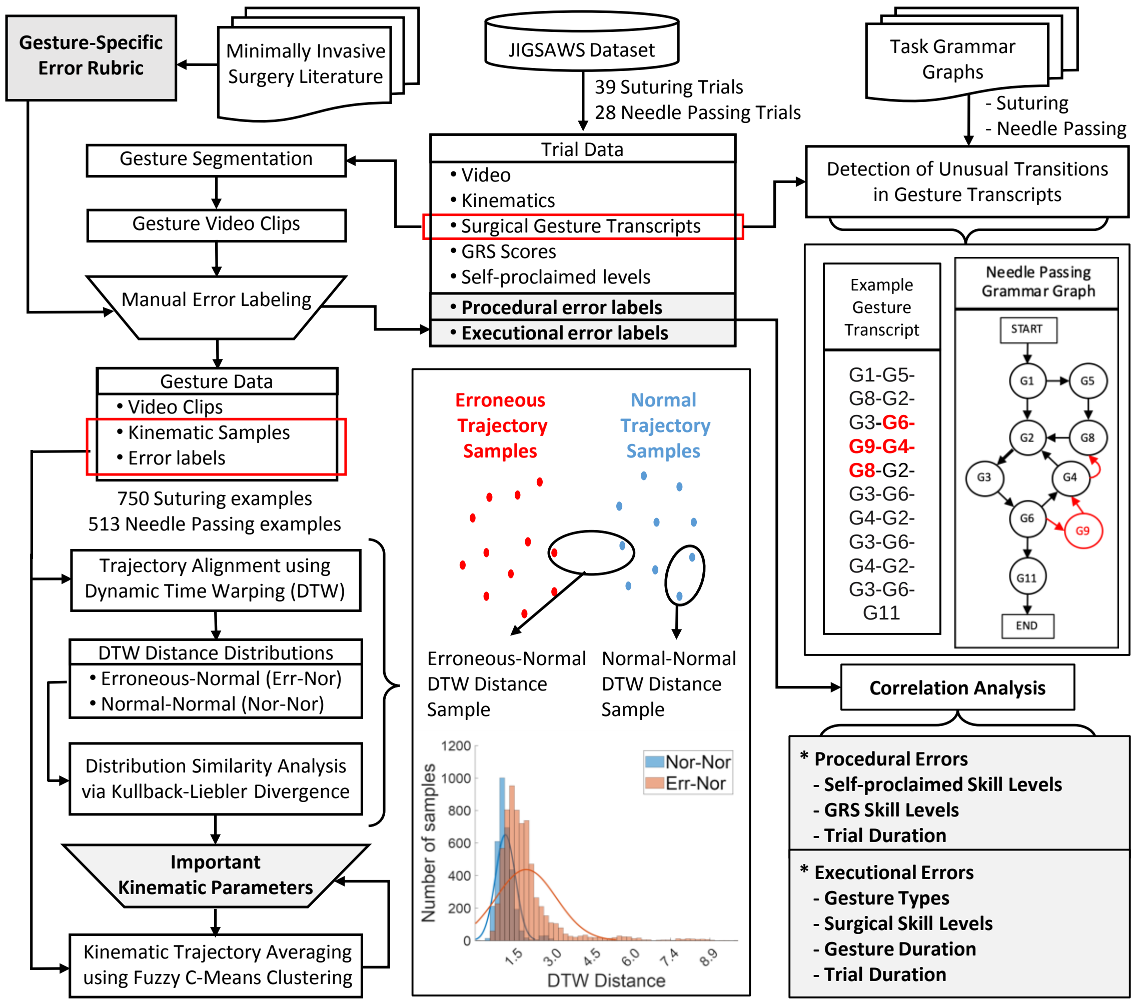

Gesture-Specific Error Rubric
Minimally Invasive Surgery Literature
JIGSAWS Dataset
39 Suturing Trials
28 Needle Passing Trials
Task Grammar Graphs
- Suturing
- Needle Passing
Gesture Segmentation
Gesture Video Clips
Manual Error Labeling
Trial Data
• Video
• Kinematics
• Surgical Gesture Transcripts
• GRS Scores
• Self-proclaimed levels
• Procedural error labels
• Executional error labels
Detection of Unusual Transitions in Gesture Transcripts
Example Gesture Transcript
G1-G5-G8-G2-G3-G6-G9-G4-G8-G2-G3-G6-G4-G2-G3-G6-G4-G2-G3-G6-G11
Needle Passing Grammar Graph
START
G1
G5
G2
G8
G3
G4
G6
G9
G11
END
Gesture Data
• Video Clips
• Kinematic Samples
• Error labels
750 Suturing examples
513 Needle Passing examples
Erroneous Trajectory Samples
Normal Trajectory Samples
Erroneous-Normal DTW Distance Sample
Normal-Normal DTW Distance Sample
Trajectory Alignment using Dynamic Time Warping (DTW)
DTW Distance Distributions
• Erroneous-Normal (Err-Nor)
• Normal-Normal (Nor-Nor)
Distribution Similarity Analysis via Kullback-Liebler Divergence
Important Kinematic Parameters
Kinematic Trajectory Averaging using Fuzzy C-Means Clustering
Number of samples
DTW Distance
Nor-Nor
Err-Nor
1200
1000
800
600
400
200
0
1.5
3.0
4.5
6.0
7.4
8.9
Correlation Analysis
* Procedural Errors
- Self-proclaimed Skill Levels
- GRS Skill Levels
- Trial Duration
* Executional Errors
- Gesture Types
- Surgical Skill Levels
- Gesture Duration
- Trial Duration

Figure 4.2: Overall methodology for the analysis of executional and procedural errors.

errors are the "failure of a specific motor task within the procedure." **Technical errors** are a subtype of executional errors that can be quantified with thresholds and are defined as the "failure of a planned action to achieve a goal," including inadequate (too much or too little) use of force or distance, inadequate visualization, and incorrect orientation of instruments or dissection plane [141]. This is similar to the definition used by [17] which we now extend. In this work, we define executional and procedural errors at the gesture level in order to generalize these definitions to different procedures and tasks and because gestures are the lowest-level activities with semantic meaning. Our goal is to define a rubric for the identification of errors based on video and/or kinematic data which can also be used for automated error detection using quantitative measures such as instrument position, amount of force, traveling distance, and system events.

We define a set of executional error modes for each gesture as listed in the rubric in Table 4.1. Some of the errors are gesture-specific such as "Needle orientation" which is only defined for G4 (transferring needle from left to right) and G8 (orienting needle) since those gestures specifically manipulate the needle in preparation for G2 (positioning the needle) and G3 (pushing needle through tissue) as shown in the grammar graphs of Figures 2.15a and 2.15b. The standard acceptable practice for those gestures is to hold the needle in the grasper $\frac{1}{2}$ to $\frac{2}{3}$ of the way from the tip of the needle and with the needle perpendicular to the jaws of the grasper [142]. Other gestures that do not purposely alter the orientation of the needle in the grasper cannot have this error mode. For G3, the definition of a "Multiple attempts" error also includes "Not moving along the curve" since these two errors are very difficult to distinguish and often occur simultaneously. Other error modes including "Multiple attempts", "Needle drop", and "Out of view", could occur at any time during a task and are thus considered for every gesture.

We define procedural errors as any deviation in the sequence of gestures performed in a demonstration from the standard accepted gesture sequences defined for that task and shown in the grammar graphs in Figures 2.15a and 2.15b from [52]. [133] defined several subcategories for procedural errors, including adding an unexpected step, skipping a step, out of order transitions, and repeating steps. These subcategories are included in our analysis of procedural errors as discussed in Section 4.2.4.

4.2.2 Dataset

The JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS) [43] is a publicly available dataset, collected using the Research API for the da Vinci Surgical System (dVSS), containing trials from eight surgeons of varying skill levels performing three dry lab surgical tasks: Suturing, Needle Passing, and Knot Tying. These tasks are among the standard modules in most surgical skill training curricula.

Table 4.1: Gesture-specific executional errors for Suturing and Needle Passing in the JIGSAWS dataset.

			Suturing		Needle Passing	
Gesture Description		Error Mode	Total No. Errors	Erroneous Gestures (%)	Total No. Errors	Erroneous Gestures (%)
G1	Reaching for needle with right hand	Multiple attempts	7	8/29 (28%)	N/A	11/30 (37%)
		Needle drop	0		2	
		Out of view	1		10	
G2	Positioning needle	Multiple attempts	21	22/166 (13%)	51	55/117 (47%)
		Needle drop	0		0	
		Out of view	1		6	
G3	Pushing needle through tissue	Multiple attempts/ Not moving along the curve	80	82/164 (51%)	17	17/111 (15%)
		Needle drop	0		0	
		Out of view	2		0	
G4	Transferring needle from left to right	Multiple attempts	19	71/119 (60%)	15	23/83 (28%)
		Needle orientation	53		9	
		Needle drop	0		0	
		Out of view	14		3	
G5	Moving to center with needle in grip	Needle drop	1	2/37 (5%)	0	3/31 (10%)
		Out of view	1		3	
G6	Pulling suture with left hand	Multiple attempts	8	121/163 (74%)	14	46/112 (41%)
		Needle drop	2		0	
		Out of view	120		37	
G8	Orienting needle	Multiple attempts	18	28/48 (58%)	1	3/28 (11%)
		Needle orientation	22		1	
		Needle drop	0		0	
		Out of view	4		2	
G9	Using right hand to help tighten suture	Multiple attempts	3	11/24 (46%)	1	1/1 (100%)
		Needle drop	0		1	
		Out of view	11		0	
Total number of errors across all gestures		Multiple attempts	156	345/750 (46%)	99	159/513 (31%)
		Needle drop	3		3	
		Needle orientation	75		10	
		Out of view	154		61	

Example videos for each error mode can be found at `https://www.youtube.com/watch?v=I7jQ6U9jaoc`, `https://www.youtube.com/watch?v=V-NJjgRu2OI`, `https://www.youtube.com/watch?v=-UNNWQ3jOyU`, and `https://www.youtube.com/watch?v=LhNg8uLRQzI`.

The JIGSAWS dataset includes kinematic and video data from 39 trials of Suturing and 28 trials of Needle Passing, along with manually annotated gesture transcripts (indicating the sequence of gestures, with the beginning and end of each gesture and its type) and surgical skill levels for each demonstration. The vocabulary of surgical gestures used for labeling is shown in Table 4.1 along with the number of examples of each gesture, and the kinematic variables and abbreviations are shown in Table 4.2.

Table 4.2: Kinematic variables in the JIGSAWS dataset (adopted from [43]).

Index	Description of variables	Parameter name
39-41	Right PSM1 tool tip position (xyz)	R Pos
42-50	Right PSM1 tool tip rotation matrix (R)	R Rot Mat
51-53	Right PSM1 tool tip linear velocity (x' y' z')	R Lin Vel
54-56	Right PSM1 tool tip rotational velocity (α' β' γ')	R Rot Vel
57	Right PSM1 gripper angle (Θ)	R Grip Ang
58-60	Left PSM2 tool tip position (xyz)	L Pos
61-69	Left PSM2 tool tip rotation matrix (R)	L Rot Mat
70-72	Left PSM2 tool tip linear velocity (x' y' z')	L Lin Vel
73-75	Left PSM2 tool tip rotational velocity (α' β' γ')	L Rot Vel
76	Left PSM2 gripper angle (Θ)	L Grip Ang

Surgical skills were characterized using both self-proclaimed expertise levels and the Global Rating Scale (GRS) score for each demonstration. Self-proclaimed (SP) expertise levels were based on the number of hours of robotic surgical experience, divided into three groups: “SP-Expert” (>100 hrs), “SP-Intermediate” (10-100 hrs), and “SP-Novice” (<10 hrs). GRS scores were given using a modified Objective Structured Assessments of Technical Skills (OSATS) approach based on six elements (on a rating scale of 1-5 per element): Respect for Tissue, Suture & Needle Handling, Time & Motion, Flow of Operation, Overall Performance, and Quality of Final Product [43]. We also categorized the demonstrations into three groups based on these GRS scores: “GRS-Novice” ($0 \leq GRS \leq 9$), “GRS-Intermediate” ($10 \leq GRS \leq 19$), and “GRS-Expert” ($20 \leq GRS \leq 30$).

4.2.3 Executional Error Analysis

Method

Kinematic and video data for each trial were first segmented into gestures based on the gesture transcript annotations. The video clip for each gesture was then reviewed and labeled by two to three independent annotators (with experience in robotic surgery and/or suturing) as normal or erroneous for each error mode. Final labels for each error mode were obtained by taking the consensus among annotators. A gesture example that exhibited one or more errors was marked as erroneous, otherwise, it was labeled as normal. We then proceeded with the analysis of the patient-side manipulator (PSM) kinematic data corresponding to each gesture for all the normal and erroneous trials of each task.

Our analytical method for executional errors compares normal and erroneous instances of each gesture and performs pairwise trajectory alignment using Dynamic Time Warping (DTW) on the kinematic data to obtain distributions of distances between pairs of normal gestures (Nor-Nor) and between normal and erroneous gestures (Err-Nor). We then perform distribution similarity analysis using Kullback-Liebler (KL) divergence to determine which kinematic parameters are most different between normal and erroneous gestures. After that, we examine those kinematic parameters by performing trajectory averaging using C-means clustering to visualize the differences in kinematics and gain insight into why these parameters with high KL divergence are important.

Dynamic Time Warping We used Dynamic Time Warping (DTW) to measure the similarity between normal and erroneous trajectories for each gesture. DTW is an effective method for aligning two temporal sequences, independent of the non-linear variations in time, by minimizing the Euclidean distance between the two signals. In our analysis, we performed independent DTW on each variable before summing the returned distances for each parameter listed in Table 4.2. We found no significant difference between this method and dependent DTW where all variables in each parameter group were warped together yielding a single distance instead of a sum of distances. Similar observations were made in [143]. DTW was performed on every combination of two example trajectories for each gesture. From this, we obtained comparisons of normal examples to other normal examples (Nor-Nor) and comparisons of erroneous examples to normal examples (Err-Nor). The DTW distance samples represented a distribution of distances for the Nor-Nor and Err-Nor subsets as shown in the histogram of Figure 4.2. This resulted in two sets of distance samples for each parameter, each representing a DTW distribution for a comparison subset.

Kullback-Liebler Divergence Kullback-Liebler (KL) divergence, also called relative entropy, is a non-symmetric measure of the difference between two probability distributions. The KL divergence between two identical distributions is zero. As shown in Equation 4.1, KL divergence was used to compare the Err-Nor and Nor-Nor DTW distance distributions for each gesture to determine which parameters had a significant difference between the two distributions.

$$D_{\text{KL}}(DTW_{\text{Err-Nor}}||DTW_{\text{Nor-Nor}}) = -\sum DTW_{\text{Err-Nor}} log\left(\frac{DTW_{\text{Nor-Nor}}}{DTW_{\text{Err-Nor}}}\right) \tag{4.1}$$

Trajectory Averaging We examined the kinematic data for important parameters to verify differences between normal and erroneous gestures using a method based on [144]. Each signal was time-normalized by downsampling the signal by 3 (keeping only every third sample) and then linearly interpolated to stretch it to the average duration of the normal or erroneous gesture examples of that task (supported by our analysis of gesture durations in Section 4.2.3). Then, fuzzy C-means clustering was performed on each variable and its normalized time index to obtain the average normal and erroneous trajectories (represented by 15 cluster centers), shown with blue (normal) and red (erroneous) dots in Figure 4.4b.

Results and Insights

Table 4.1 lists the number of examples of each error mode as well as the total number of erroneous examples for each gesture. Note that a gesture example could exhibit multiple error modes, so the sum of the total number of errors does not necessarily equal the number of erroneous gestures.

Distribution of Executional Errors Among Gestures Figure 4.3 shows the distribution of errors of each type for each gesture from Table 4.1. If a gesture example had more than one error label, it was counted under the "Multiple errors" category. We made the following observations:

- G5 for both tasks and G1, G8, and G9 for Needle Passing did not have enough examples of executional errors, so further analysis was not performed on these gestures. G8 from Needle Passing and G5 from both tasks had the lowest percentage of errors because they may be less challenging than other gestures.
- G2 and G3 have the most "Multiple attempts" errors in both tasks because they require a high level of accuracy in positioning and driving the needle though the tissue, respectively. G2 has more errors in Needle Passing because the eye of the ring is a smaller target than the dot on the fabric. G3 has more errors in Suturing because surgeons often tried multiple times to align the tip of the needle with the exit point while the needle was not visible beneath the fabric. Comparatively in Needle Passing, the needle only had to pass through one point and was always visible.
- G4 and G6 from both tasks, and G8 from Suturing have the most gestures with "Multiple errors". G4 and G8 both involve manipulating the needle between the graspers and the predominant error modes were "Needle orientation" and "Multiple attempts" likely due to issues with hand coordination. For G6, the main error modes were "Out of view" and "Multiple attempts" due to multiple attempts at grasping the needle and pulling it through the ring or tissue and then moving off-camera to pull the suture through.

- G6 has a large number of "Out of view" errors especially in Suturing possibly because surgeons could not move the camera for the trials in the JIGSAWS dataset. However, a different technique to pull the suture could have been used, such as the hand-over-hand or pulley methods, that would have kept the tools within view.

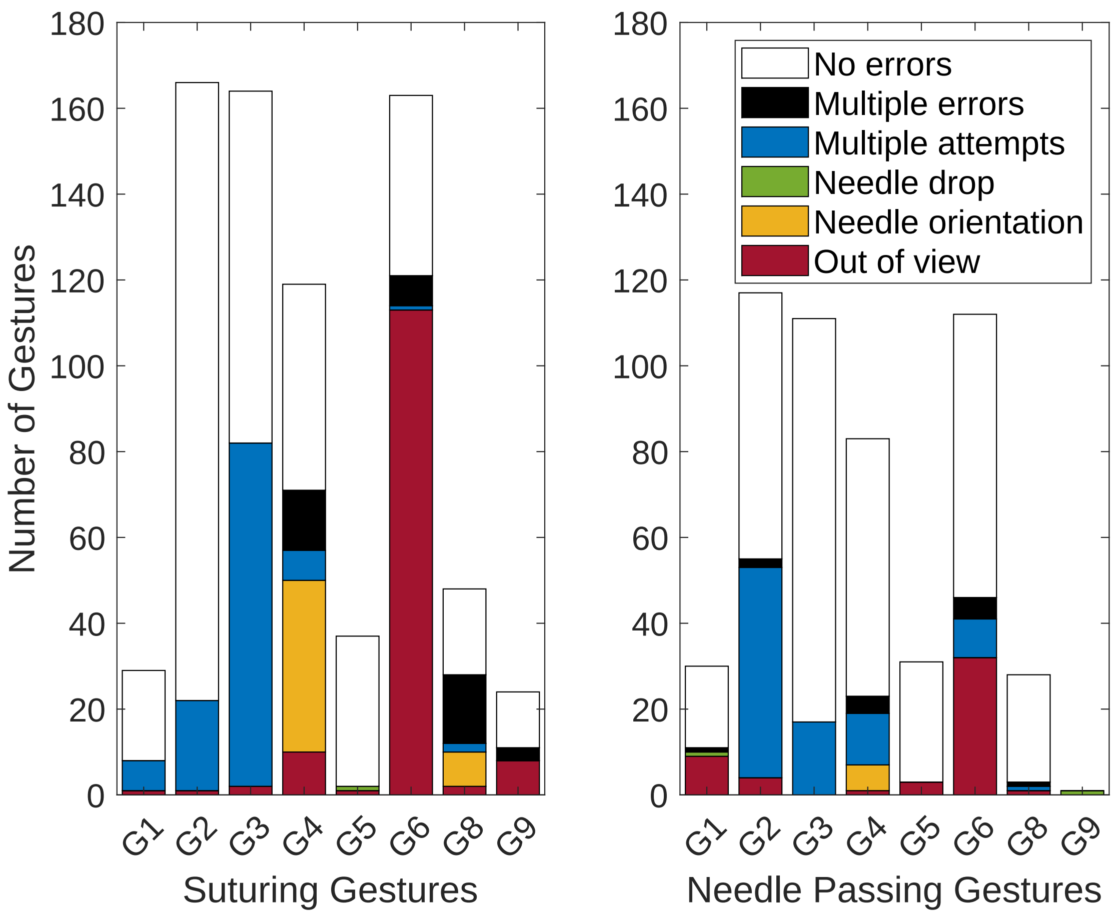


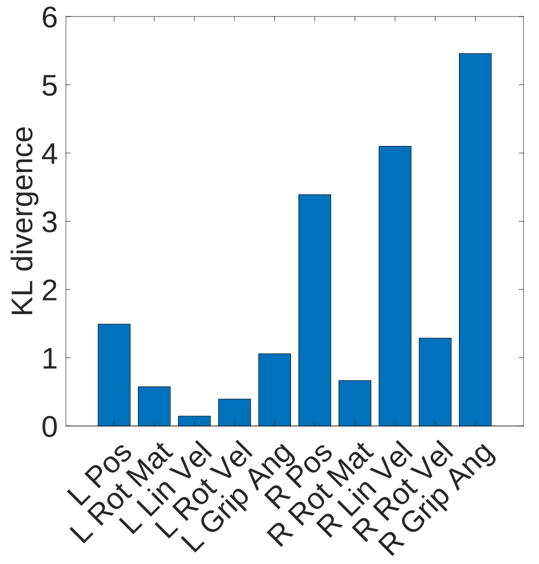
Figure 4.3: Distribution of executional errors for each gesture.

Kinematic Parameter Analysis We performed a comparative analysis of KL divergence values for parameters in each gesture and identified the kinematic parameters that are associated with error occurrences as listed in Table 4.3. We also examined the KL divergence of the kinematic data between normal gestures of each task to quantify their similarity. The following are key observations from this analysis:

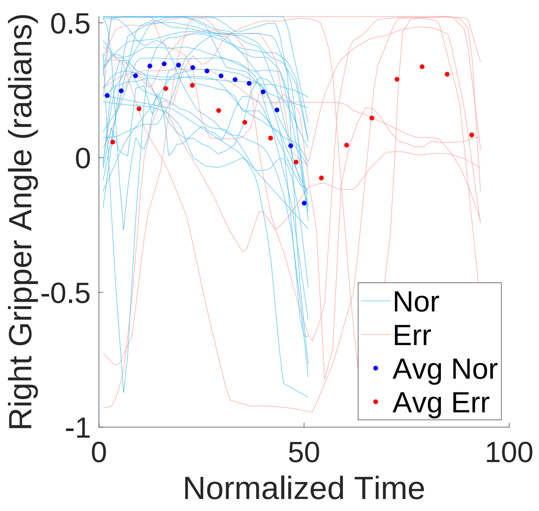
Table 4.3: Kinematic parameters with the greatest KL divergence distinguishing errors in different gestures.

Task	Gesture	Parameters
Suturing	G1	Right Gripper Angle Right Linear Velocity Right Position
	G3	Right Linear Velocity Right Rotational Velocity Right Gripper Angle
	G6	Left Position
	G8	Right Position Left Gripper Angle Left Linear Velocity Right Gripper Angle
	G9	Left Gripper Angle
Needle Passing	G2	Left Rotational Velocity Left Linear Velocity
	G3	Left Rotational Velocity Right Rotation Matrix Right Gripper Angle

- For G1 in Suturing, the predominant error mode was "Multiple attempts" at picking up the needle. Figures 4.4b and 4.5 show that erroneous gestures exhibited a second opening and closing of the grasper and a large difference in Y Position trajectories. This explains the large KL divergences for those right hand parameters in Figure 4.4a.



(a)



(b)

Figure 4.4: G1 in Suturing: (a) KL divergences of kinematic parameters, (b) right gripper angle trajectories for normal and erroneous gestures.

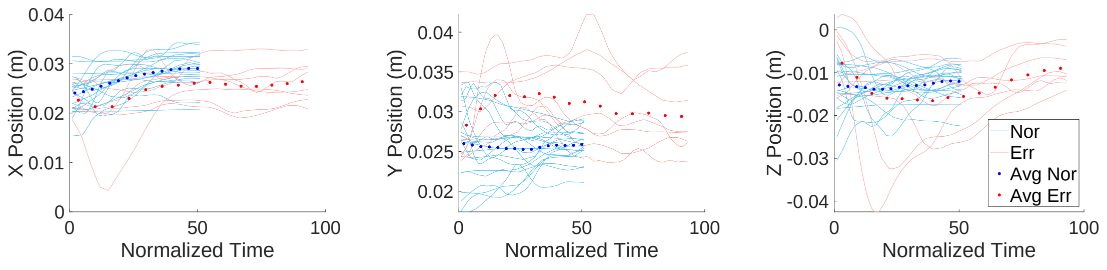
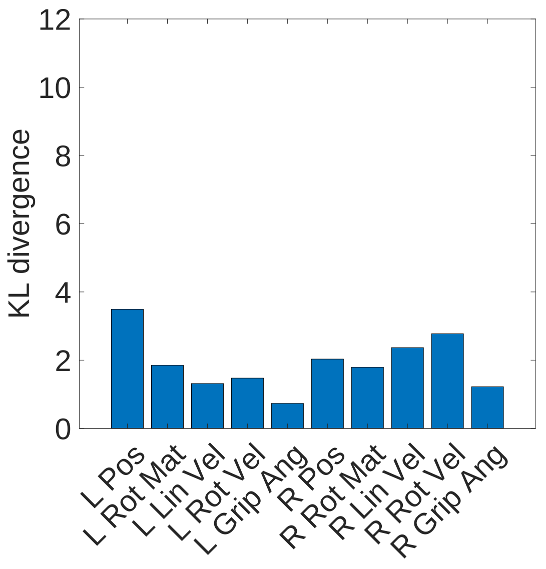


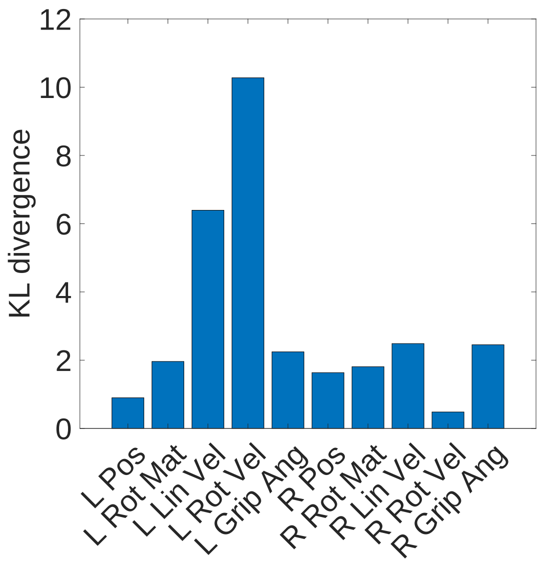
Figure 4.5: Right tooltip xyz position for normal and erroneous G1 in Suturing.

- For G2 in Needle Passing, Figure 4.6 shows a large difference in KL divergence for Left Rotational and Linear Velocities which may be due to the active role the left hand plays in stabilizing the ring unlike in Suturing. This is an important contextual difference between tasks.

- The main error mode for G3 was “Not moving along the curve/Multiple attempts”. Erroneous gestures in Suturing were caused by lateral, instead of characteristically rotational, movements of the needle while in the fabric. In surgery, lateral movements may tear tissue and contribute to a safety-critical



(a) Suturing



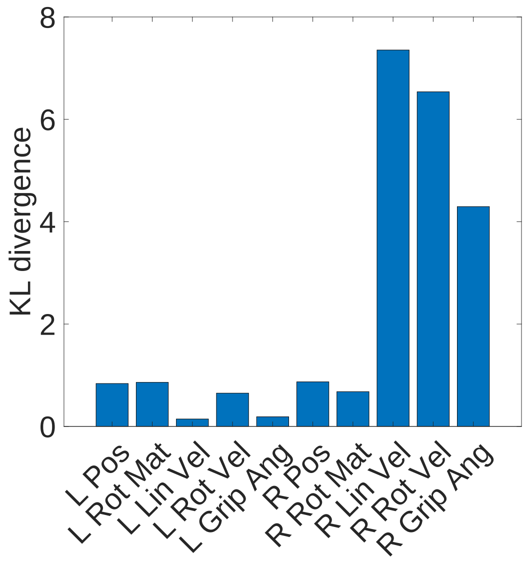
(b) Needle Passing

Figure 4.6: KL divergences of kinematic parameters for G2.

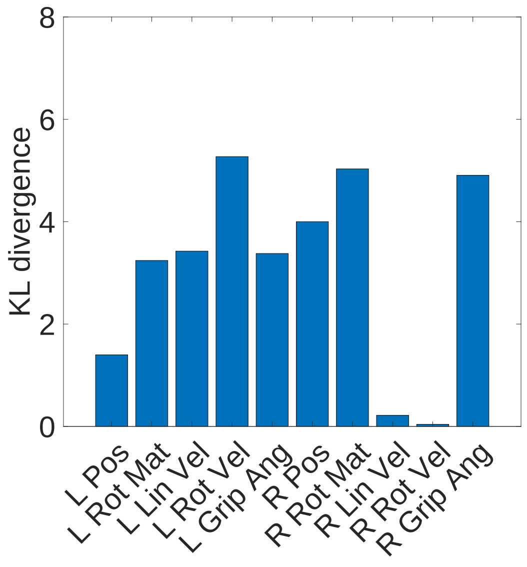
event. This explains the high KL divergences for the parameters listed in Table 4.3 and shown in Figure 4.7a. This is also consistent with [145] who found that the rate of orientation change during needle insertion (i.e. Rotational Velocity during G3) was higher for experienced surgeons.

However, Needle Passing shows nearly the opposite result in Figure 4.7b. Upon reviewing the gesture clips for both tasks, we noticed that clips for Suturing showed the right grasper driving the needle through the fabric and the left grasper pulling it through, but clips for Needle Passing began with the needle halfway through the ring and only showed the left grasper pulling the needle through. Due to the large difference in KL divergences between the two tasks, we see that the part of G3 that involves driving the needle with the right grasper is important to the correct execution of this gesture.

- In both tasks, G4 had KL divergences below 0.6 for all parameters meaning normal and erroneous examples had very similar kinematics.



(a) Suturing



(b) Needle Passing

Figure 4.7: KL divergences of kinematic parameters for G3.

- G6 in Suturing had the most errors with primarily "Out of view" errors. Figure 4.9 shows that final Y and Z Positions for the left grasper were much larger for erroneous gestures since the left grasper exceeded the threshold for visibility while pulling the suture. This explains the large KL divergence for the Left Position parameter in Figure 4.8.

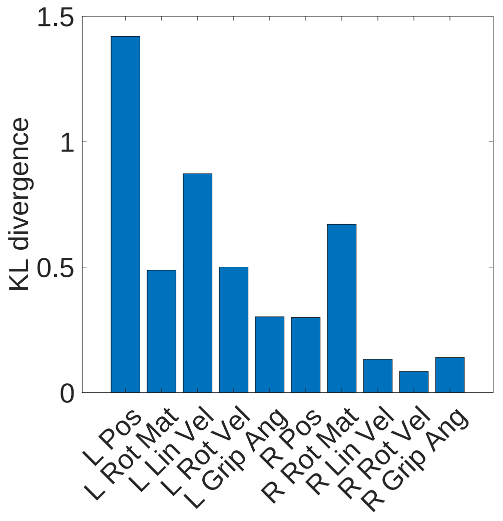


Figure 4.8: KL divergences of kinematic parameters for G6 in Suturing.

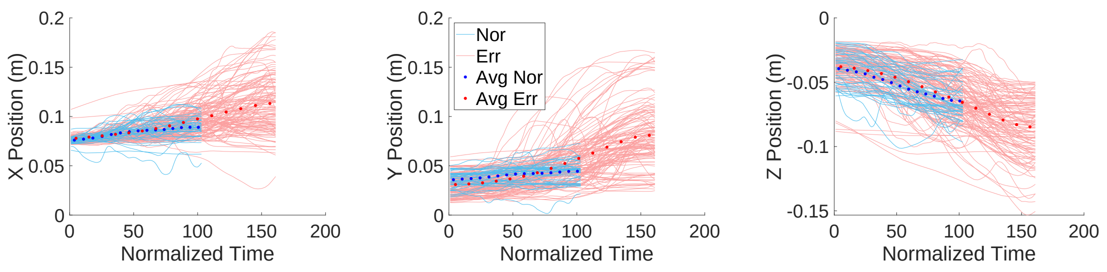
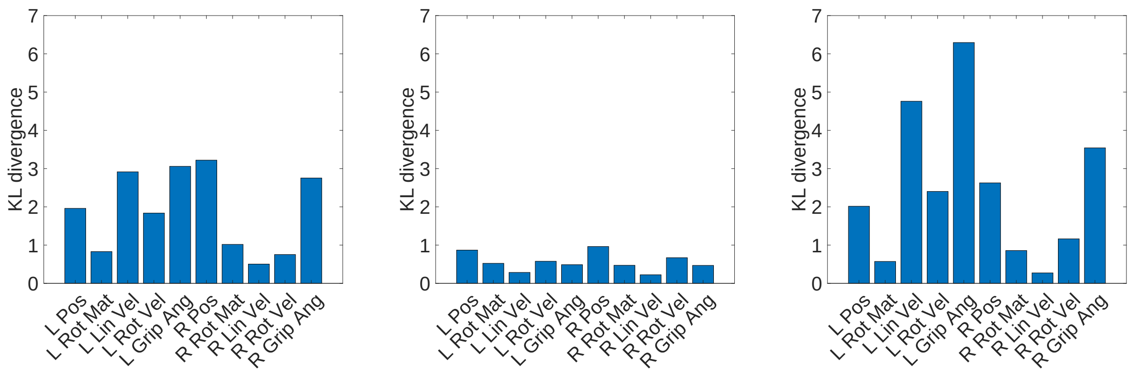


Figure 4.9: Left tooltip xyz position for normal and erroneous G6 in Suturing.

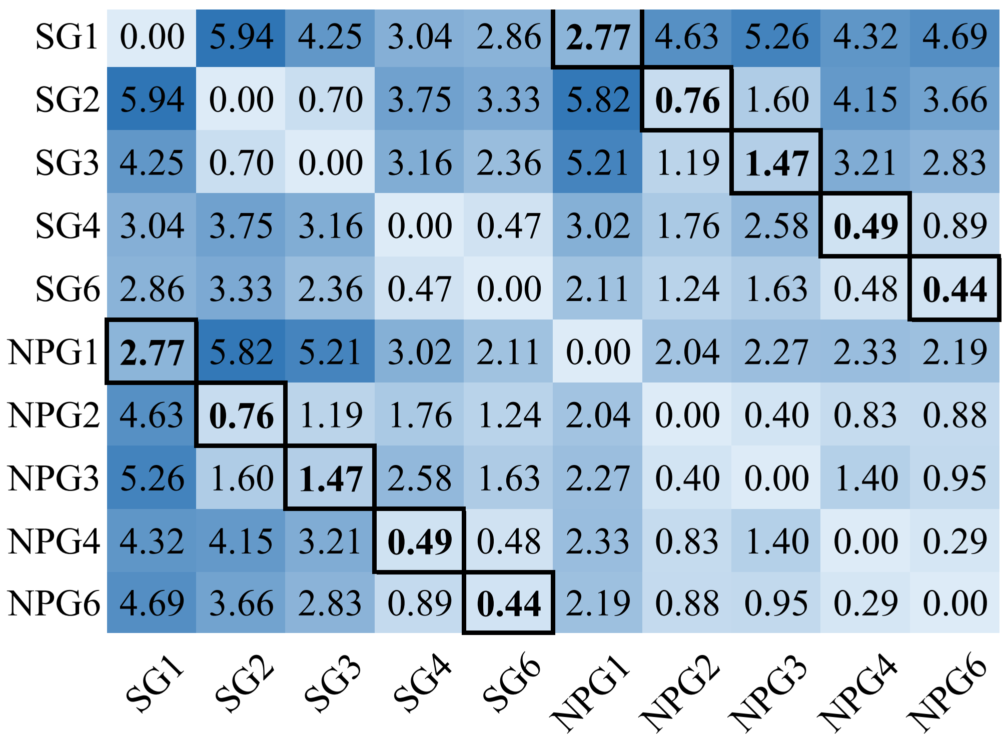
- There were two main error modes for G8 in Suturing: "Multiple attempts" and "Needle orientation". Figure 4.10 shows a comparison of DTW and KL divergence analysis for G8 from Suturing for all errors, for "Multiple attempts" versus all other examples, and for "Needle orientation" versus all other examples. The "Needle orientation" error alone had the greatest KL divergences and contributed the most to the results for all errors. For the "Multiple attempts" error, both the Left and Right Position parameters had the highest KL divergences which suggests that hand coordination is important in this gesture. Since this gesture includes the right gripper moving to grasp the needle, we see that Right Position is an important parameter in the "Multiple attempts" error in both G1 and G8.



(a) All errors (b) Multiple attempts errors (c) Needle orientation errors

Figure 4.10: KL divergences of kinematic parameters for G8 in Suturing.

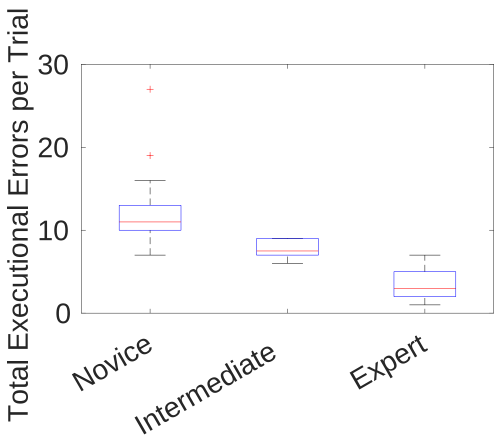
Since the Suturing and Needle Passing tasks contained many of the same gestures, we also investigated the similarity of normal gestures between these two tasks. Similarities between these tasks could support data aggregation such as for training models for skill assessment and error detection. We used KL divergence to quantify the differences in the distributions of the values of the kinematic variables. The average KL divergences between two gestures are shown in Figure 4.11. The squares outlined in black show the KL divergences between distributions of kinematic data for the same gestures from the two different tasks. These values are the smallest in the rows for G1, G4, and G6. For G2 and G3, 0.76 and 1.47, respectively, are smaller than most other elements in their respective rows in the quadrants comparing gestures from Suturing and Needle Passing. This means that normal gestures are similar between tasks. However, the KL divergence values between the gestures used in Needle Passing are generally lower than between other tasks and gestures. Thus, Needle Passing gestures are more similar to other Needle Passing gestures which could contribute to errors in gesture recognition.



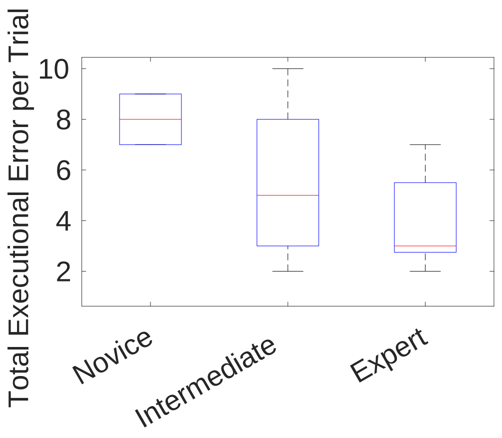
	SG1	SG2	SG3	SG4	SG6	NPG1	NPG2	NPG3	NPG4	NPG6
SG1	0.00	5.94	4.25	3.04	2.86	**2.77**	4.63	5.26	4.32	4.69
SG2	5.94	0.00	0.70	3.75	3.33	5.82	**0.76**	1.60	4.15	3.66
SG3	4.25	0.70	0.00	3.16	2.36	5.21	1.19	**1.47**	3.21	2.83
SG4	3.04	3.75	3.16	0.00	0.47	3.02	1.76	2.58	**0.49**	0.89
SG6	2.86	3.33	2.36	0.47	0.00	2.11	1.24	1.63	0.48	**0.44**
NPG1	**2.77**	5.82	5.21	3.02	2.11	0.00	2.04	2.27	2.33	2.19
NPG2	4.63	**0.76**	1.19	1.76	1.24	2.04	0.00	0.40	0.83	0.88
NPG3	5.26	1.60	**1.47**	2.58	1.63	2.27	0.40	0.00	1.40	0.95
NPG4	4.32	4.15	3.21	**0.49**	0.48	2.33	0.83	1.40	0.00	0.29
NPG6	4.69	3.66	2.83	0.89	**0.44**	2.19	0.88	0.95	0.29	0.00

Figure 4.11: Average KL divergences among distributions of normal gestures.

Executional Errors and Skill Levels We analyzed the relationship between executional errors and surgical skill levels. Based on self-proclaimed expertise levels, Figure 4.12a shows a clear difference in the number of errors across different self-proclaimed expertise groups for Suturing. However, no similar pattern was seen in Needle Passing. This might be because Suturing is a more difficult task so the number of executional errors is more reflective of self-proclaimed skill levels in Suturing. For GRS-defined skill levels, the total number of executional errors per trial was larger for GRS-Novices than for GRS-Experts in Needle Passing (Figure 4.12b), which is consistent with our expectation that experts with high GRS scores make fewer executional errors than novices. However, since there was only one GRS-Novice trial for Suturing, we did not observe clear differences.



(a) Self-proclaimed skill levels

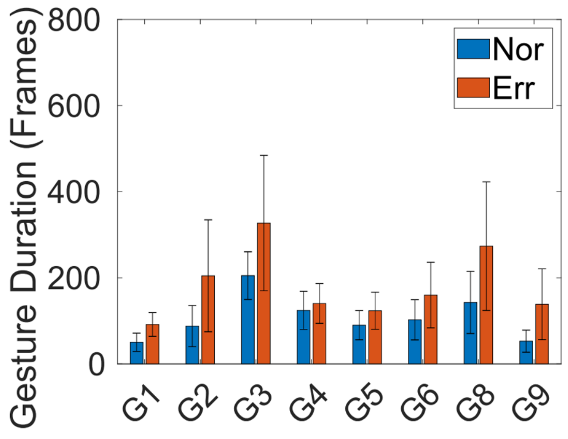


(b) GRS skill levels

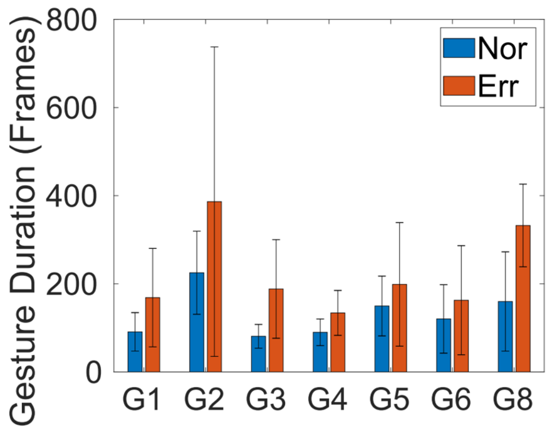
Figure 4.12: Total number of executional errors across surgical skill levels for (a) self-proclaimed skill levels in Suturing, and (b) GRS skill levels in Needle Passing.

Executional Errors and Gesture Duration We compared erroneous and normal gesture durations using a one-tailed t-test. The null hypothesis is that normal and erroneous gestures have similar durations. The alternative hypothesis is that erroneous gestures are longer than normal gestures. Figures 4.13 and 4.14, respectively, show the average durations and several examples of differences in durations (along with the p-values from the hypothesis test) for normal and erroneous gestures in both tasks.

We observed that some error types increase the gesture duration, e.g., "Multiple attempts" for G1, G2, G3, and G8 in Suturing, and G2 and G3 in Needle Passing; and "Out of view" for G6 and G9 for Suturing, and G4 in Needle Passing. Erroneous gestures with "Out of view" errors are longer because the distance traveled by the tool is larger, while the speed is similar. We rejected the null hypothesis and found that erroneous gestures are longer than normal gestures for all gestures of both tasks. There is a relatively large p-value (p=0.308) for G4 compared to other p-values. This could be because "Needle orientation" is the primary error mode in G4 and orienting the needle erroneously takes about the same amount of time as orienting it correctly.

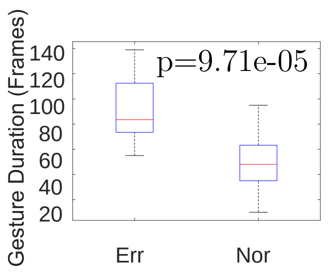


(a) Suturing

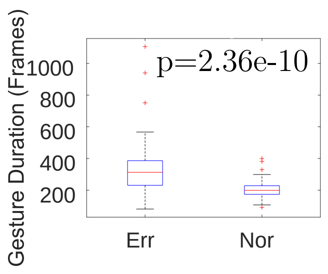


(b) Needle Passing

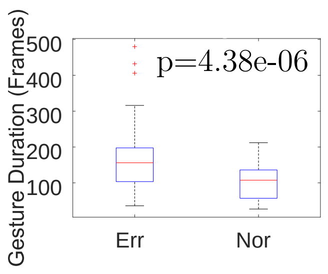
Figure 4.13: Average normal and erroneous gesture durations.



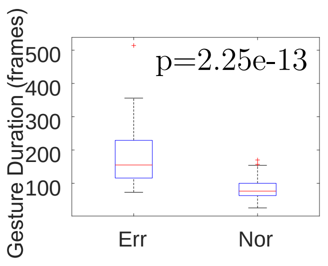
(a) Suturing G1



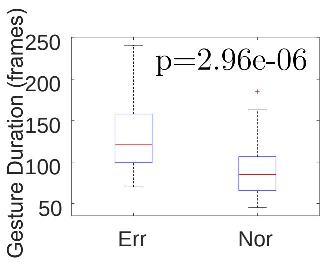
(b) Suturing G3



(c) Suturing G6



(d) Needle Passing G3



(e) Needle Passing G4

Figure 4.14: Erroneous vs. normal gesture durations for Suturing and Needle Passing.

Executional Errors and Trial Duration For each trial, we summed the number of executional errors of all gestures in the trial. Then, we analyzed the correlation between the total number of executional errors per trial and the duration of the trial (in number of frames). Figure 4.15 shows that there is a significant positive correlation for Suturing (r=0.837, p=6.18e-12), but no significant correlation for Needle Passing. This is likely due to the limited number of trials and fewer errors for Needle Passing in the JIGSAWS dataset (see Table 4.1).

4.2.4 Procedural Error Analysis

Method

Previous works proposed modeling the standard acceptable gesture sequences for a task using a grammar graph that shows the relationship, order, and flow of gestures [12, 13, 52]. The grammar graph of a task is a digraph with the vertices representing the set of gestures for the task and the edges representing common transitions between two gestures. We adopted the grammar graphs for Suturing and Needle Pass-

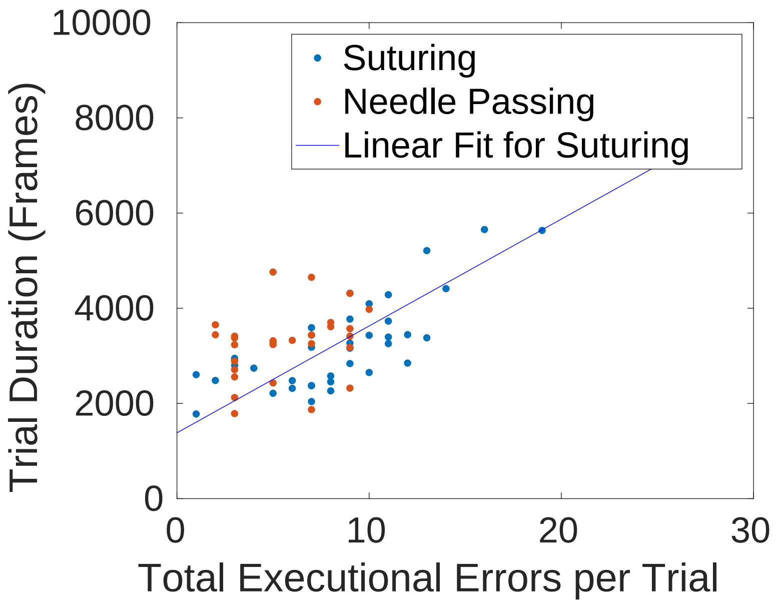


Figure 4.15: Correlation between executional errors and durations of trials.

ing from [52] and included an additional directed link from G1 to G2 in Suturing (see Figures 4.1 and 4.2).

We acquired the gesture sequences performed for Suturing and Needle Passing from the JIGSAWS transcripts. Then, we developed a method for checking if each gesture sequence follows the standard acceptable sequence of gestures in the grammar graph. As shown in Algorithm 1, for each gesture we check if it is in the grammar graph for the task and if it is a valid successor of the previous gesture, otherwise it is marked as a procedural error. Each transcript can have multiple, possibly sequential, procedural errors. This algorithm, combined with a gesture segmentation algorithm, can be used for the automated detection of procedural errors in real-time. Deviations from the grammar graph might also happen because of variations in surgical style and expertise, as discussed in Section 4.2.4.

Algorithm 1 Procedural Error Detection Algorithm

Input:

- A grammar graph $G(V, E)$ for a surgical task which is a digraph with each vertex in V representing the entry point START or one of the common gesture types G_i in the task, and each $edge(G_i, G_j) \in E$ representing a common transition between gestures G_i and G_j.

- A set of m task transcripts $T = \{T_1, T_2, ..., T_m\}$, where $T_k \in T$ is an ordered sequence of n gestures $T_k = [G_1, G_2, ..., G_n]$

Output:

- A list of erroneous gesture transitions *error_seq* for each transcript

```
for $T_k \in T$
    $error\_seq = \emptyset$
    $val \leftarrow G.successors(START)$
    for $G_i \in T_k$
        if $G_i \in V$
            if $G_i \notin val$
                $error\_seq$.append($[G_{i-1}, G_i]$)
            $val \leftarrow G.successors(G_i)$
        else
            $error\_seq$.append($[G_i]$)
            $val \leftarrow [G_{i+1}]$
```

Results and Insights

We analyzed the numbers and patterns of procedural errors by task, skill level, and subject. We hypothesized that the number of procedural errors would be inversely proportional to surgical experience and negatively correlated with the demonstration duration.

Procedural Errors and Self-Proclaimed Skill Levels We compared the percentage of erroneous trials for the SP-Novice, SP-Intermediate and SP-Expert groups. As shown in Table 4.4, we observed that for both tasks, SP-Expert surgeons on average had more procedural errors compared to SP-Intermediate surgeons. For Needle Passing, SP-Intermediate surgeons made more errors than SP-Novice surgeons. This could be due to variations in surgical style especially in more experienced surgeon groups. For example, our analysis of error patterns by subject showed that one of the

Table 4.4: Procedural errors and self-proclaimed skill levels.

Task	Skill Level	Total Number of Procedural Errors	Percentage of Erroneous Trials	Longest Erroneous Gesture Sequences
Suturing	SP-Expert	11	6/10	G9-G6-G2
	SP-Intermediate	2	2/10	G3-G11
	SP-Novice	23	10/19	G4-G5-G6-G2
Needle Passing	SP-Expert	11	6/9	G2-G6-G10
	SP-Intermediate	9	5/8	G6-G8-G6
	SP-Novice	7	4/11	G6-G5-G6
Total		63	33/67	

SP-Expert subjects consistently made G9-G11 transitions in different trials of Suturing. This is a unique non-safety-critical pattern that was not observed in the trials by other subjects. However, procedural errors by SP-Novice subjects were more random and did not follow specific patterns.

Of the two tasks, the longest erroneous gesture sequence is G4-G5-G6-G2 in Suturing performed by an SP-Novice surgeon. Upon review of the video, G5 may be a typo in the transcript.

Table 4.5: Correlation between the number of procedural errors and GRS subscores for Suturing and Needle Passing.

GRS Subscore	Suturing		Needle Passing	
	Correlation Coefficient	p-value	Correlation Coefficient	p-value
Respect for Tissue	-0.41	0.009	-0.12	0.528
Suture & Needle Handling	-0.50	0.001	-0.26	0.184
Time & Motion	-0.55	<0.001	-0.11	0.594
Flow of Operation	-0.43	0.006	-0.22	0.268
Overall Performance	-0.62	<0.001	-0.16	0.412
Quality of Final Product	-0.26	0.115	-0.02	0.920
GRS Score	-0.51	<0.001	-0.15	0.434

Procedural Errors and GRS Skill Levels We analyzed the correlation between the number of procedural errors and GRS score (Table 4.5). The strongest negative correlation between the number of procedural errors, GRS score, and GRS subscores is in Suturing. Among the subscores of Suturing, Overall Performance has the strongest negative correlation with procedural errors. This could happen because an inefficient procedure has the greatest impact on Overall Performance in Suturing. Needle Passing has a weaker negative correlation between procedural errors and GRS score. The Suture & Needle Handling subscore has the highest negative correlation with the number of procedural errors. This is expected since needle handling is the main component of the Needle Passing task and poor performance due to procedural errors would lead to a lower score.

Procedural Errors and Trial Duration In Suturing, there is a significant positive correlation between procedural errors and the durations of the trials, so more procedural errors lead to longer trials. However, there is no significant correlation in Needle Passing possibly because Needle Passing is an easier task (Table 4.6).

Table 4.6: Correlation between procedural errors and trial durations.

Task	r	p-value
Suturing	0.71	<0.001
Needle Passing	0.17	0.399

4.2.5 Research Questions

We used our insights from the analysis of executional and procedural errors in the JIGSAWS dataset to answer the following research questions:

RQ 1: Which tasks and gestures are most prone to errors? The more challenging gestures in each task that required a high level of accuracy and hand coordination were more prone to executional errors. As shown in Table 4.1, Suturing is more difficult than Needle Passing and had a greater number of executional errors. G6, G3, and G4 had the greatest number of executional errors in Suturing while G2 and G6 had the greatest number of executional errors in Needle Passing.

However, procedural errors were almost equally likely in both tasks. 18/39 Suturing trials and 15/28 Needle Passing trials contained procedural errors (Table 4.4).

RQ 2: Are there common error modes or patterns across gestures and tasks? Within each task, each gesture had a different predominant error mode that correlated with the challenging aspects of performing the gesture. For both tasks, G2 and G3 had a large number of "Multiple attempts" errors, G5 had the fewest errors, G6 had the largest number of "Out of view" errors, and G4 and G6 had the greatest number of gestures with "Multiple errors". Thus, the types and frequencies of executional errors are both task- and gesture-specific.

RQ 3: Are erroneous gestures distinguishable from normal gestures? KL divergence magnitudes provide insight into which gestures have the greatest difference between normal and erroneous examples. We found that G9 from Suturing, G2 from Needle Passing, and G3 from Suturing had the three greatest KL divergences for any parameter. However, upon examination of the kinematic data for the Left Gripper Angle of G9 from Suturing, the large KL divergence for this gesture could be due to the effect of three outlying gestures on an already relatively small set of only 24 examples.

RQ 4: What kinematic parameters can be used to distinguish between normal and erroneous gestures? Table 4.3 lists the parameters with the greatest KL divergences for each gesture and task which can be used to develop automated error detectors. Focusing on a subset of variables for a given task and gesture may enable and improve real-time error detection and skill assessment by reducing processing time and providing context. Our KL divergence analysis approximated the DTW distance distributions as Gaussian, which might not always be accurate. Future work will focus on further refining our analytical method to address this limitation.

RQ 5: Do errors impact the duration of the trajectory? Executional and procedural errors often lead to lengthier trials, especially during more complicated tasks such as Suturing. Timely detection and correction during training or surgery will enable more efficient and safer patient care, and aid in reducing learning curves and time to certification.

RQ 6: Are there any correlations between errors and surgical skill levels? The total number of executional errors made per trial could help differentiate between skill levels. We found this to be true for self-proclaimed skill levels in Suturing and GRS skill levels in Needle Passing.

There was a significant negative correlation between overall GRS scores and subscores and the total number of procedural errors made per trial in Suturing meaning a greater number of procedural errors contributes to lower GRS scores. After examining procedural error patterns, we noticed that self-proclaimed novice surgeons tend to closely follow the grammar graph, but experts have unique signatures that deviate from the graph. This motivates developing automated gesture identification and procedural error detection techniques based on grammar graphs for training novice surgeons in simulation experiments. Further verification of the correlation between errors and skill levels requires access to larger datasets representing more tasks and surgeons. Additionally, the grammar graphs cannot completely capture all possible valid gesture sequences and surgeon-specific signatures, and manual labeling may introduce errors in the gesture transcripts (e.g., incorrectly adding or missing some gestures) that might lead to the incorrect detection of errors.

4.3 Motion Primitive-Level Analysis

Previous work found that gesture-level HMMs for skill assessment were better than task-level models, and that building the task-level models using gestures as the states was better than learning the states from unlabeled kinematic data [146]. In addition, [13] and [12] used HMMs for motion-level skill assessment and found that the sequence of states was indicative of skill level. This motives a lower, motion-level analysis of surgical activity enabled by the fine-grained motion primitive labels in the COMPASS dataset.

4.3.1 Methods

This section presents an overview of the surgical process model and datasets used in this study and our methods for the analysis of motion primitive sequences and their relationship to gestures and surgical skill.

Surgical Process Model

Surgical procedures are often modeled by decomposing them into finer levels of granularity, including phases, steps, tasks, gestures, and actions [9, 10]. Surgical **tasks** such as suturing and knot tying are modeled as a sequence of **gestures** (shown as grammar graphs [52]) that represent units of surgical activity with semantic meaning and specific intents (Section 3.2). Gestures (also called sub-tasks or surgemes) can be decomposed into finer-grained atomic motions, referred to as **motion primitives** (MPs), dexemes [8, 13], or action triplets [49]. In the literature, MPs are often modeled as triplets [49, 147] that encode the type of action, the tool that is used, and the object with which the tool interacts. In this study, we use the model and notation from Chapter 2 (e.g., Grasp(L, Needle)) where the left and right tools are abbreviated as "L" and "R".

Datasets

We use the publicly available JIGSAWS dataset [43] that contains kinematic and video data from dry lab simulation training experiments, including 39 trials of Suturing, 28 trials of Needle Passing, and 36 trials of Knot Tying. These trials were performed by two expert (E) surgeons with more than 100 hours of robotic surgical experience, two intermediate (I) surgeons with between 10 and 100 hours of experience, and four novice (N) surgeons with less than 10 hours of experience. Each trial was annotated with gesture labels and received a Global Rating Scale (GRS) score comprised of subscores for the areas of Respect for Tissue, Suture & Needle Handling, Time & Motion, Flow of Operation, Overall Performance, and Quality of Final Product which was based on a modified OSATS approach [148]. We also use the COMPASS dataset from Chapter 2 that provides MP labels for the tasks in the JIGSAWS dataset. In this analysis, we focus on gestures that are repeated multiple times during a trial as listed in Table 4.7.

Table 4.7: JIGSAWS gesture definitions from [43] and surgeon-defined motion primitive sequences based on Chapter 2 where N = Needle.

Gesture and Description		Task	Surgeon-defined Motion Primitive Sequence
G2	Positioning needle	S	Touch(Needle, Fabric)
		NP	Release(L, N), Touch(Needle, Ring), Push(Needle, Ring)
G3	Pushing needle through tissue	S	Touch(Needle, Fabric), Push(R, N), Grasp(L, N)
		NP	Grasp(L, N)
G4	Transferring needle from left to right	S	Grasp(R, N), Release(L, N)
		NP	Grasp(R, N)
G6	Pulling suture with left hand	S	Grasp(L, N), Release(R, N), Pull(L, N)
		NP	Release(R, N), Pull(L, N)
G8	Orienting needle	S	Grasp(L, N), Release(R, N), Grasp(R, N), Release(L, N)
		NP	Release(R, N), Grasp(R, N)
G12	Reaching for needle with left hand	KT	Grasp(L, Thread), Release(R, Thread)
G13	Making C loop around right hand	KT	Pull(L, Thread), Touch(R, Thread)
G14	Reaching for suture with right hand	KT	Grasp(R, Thread)
G15	Pulling suture with both hands	KT	Pull(L, Thread) Pull(R, Thread)

MP Sequence Analysis

As the first step, two expert surgeons in urology and gynecology with experience in robotic surgery reviewed the gesture definitions and videos of gesture trials and defined an ideal sequence of MPs for each gesture as listed in Table 4.7. Since these surgeon-defined MP sequences omitted Touch and Untouch MPs immediately preceding or following Grasp and Release MPs, respectively, of the same object with the same tool, we combine MPs in the transcripts meeting those requirements prior to our MP sequence extraction and subsequent analysis.

We extracted the corresponding MP sequence for each gesture trial from the COMPASS dataset based on the start and end frames of the gestures in JIGSAWS. If an MP overlapped at the beginning or end of a gesture, it was *included in the MP sequence for the gesture*. For example, in Figure 4.16, Grasp(L, Needle) is included in the MP sequences for G2, G3, and G6 since it overlaps all of them. But this also means that in the video clip created for the gesture, *parts of those MPs get cut off in the video* (because the start and end frames of the gesture are used to create the video clips).

Then, we examine the varieties of MP sequences by plotting the frequencies of the MP sequences for each gesture in each task and creating state graphs to visualize their transitions. This provides insight into how surgeons perform gestures, reveals specific patterns in the MPs, and identifies inconsistencies in the gesture annotations.

Inverse MP Definition and Analysis

We define **inverse MPs** as two or more sequential MPs performed by the same tool on the same object whose verbs effectively negate or undo each other (e.g., Touch(Needle, Fabric), Untouch(Needle, Fabric), Touch(Needle, Fabric) of G2 in Figure 4.16). Inverse MPs are easily identifiable patterns in the MP sequences of the gestures or tasks. They may be symptoms of issues with depth perception or used as recovery actions to correct the position or orientation of objects. However, they may be necessary in some cases like the Release(R, Needle), Grasp(R, Needle) in G8, so we do not count those instances in G8 where inverse MPs were part of the surgeon-defined MP sequences

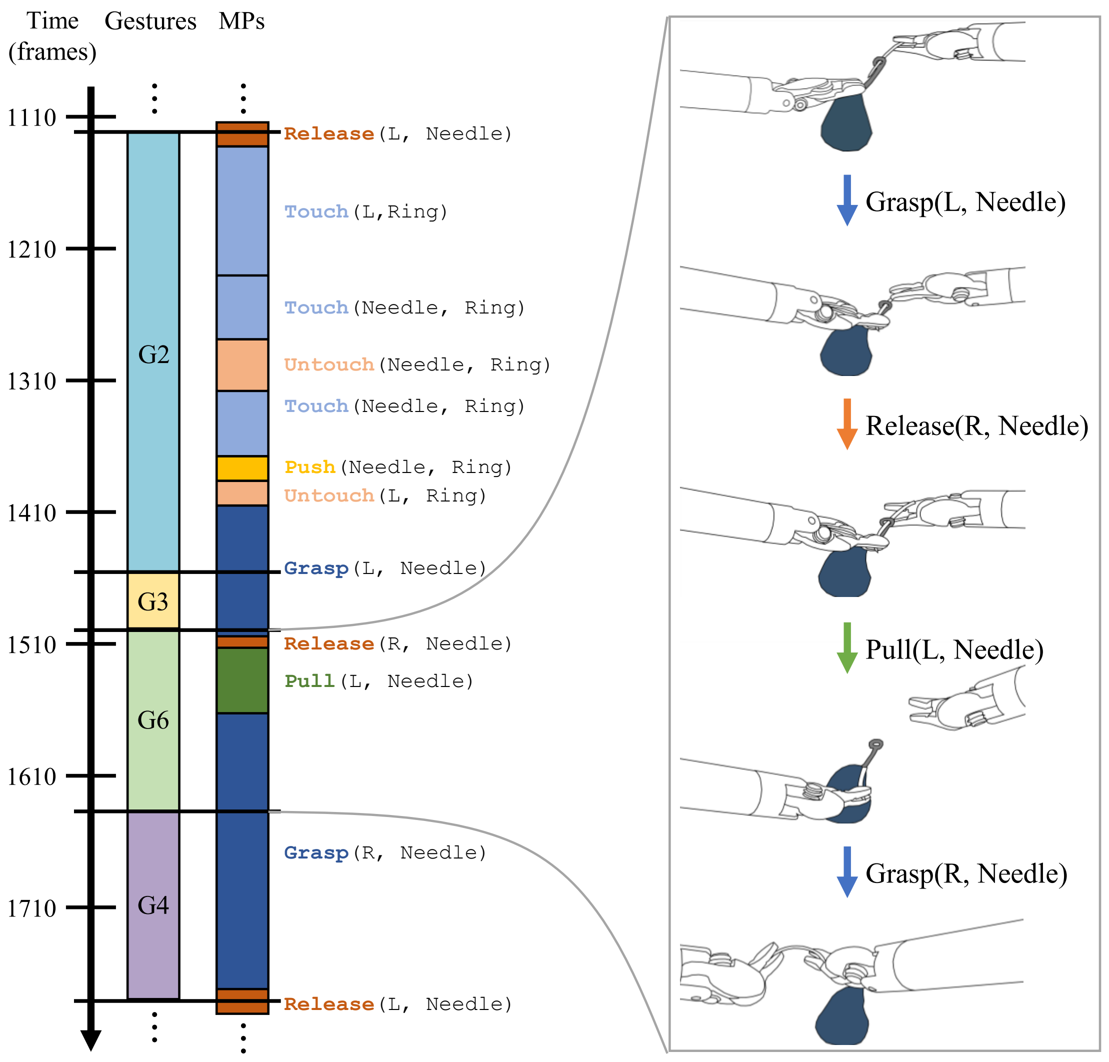


Figure 4.16: Example of gesture and motion primitive labels for a trial of Needle Passing and the motion primitive sequence extracted for G6.

as shown in Table 4.7. We then manually review the video clips associated with each gesture containing one or more instances of inverse MPs to confirm their presence and gain insight about the relationship between their types and the gestures they occur in.

Correlation between Inverse MPs and GRS Scores

Since inverse MPs may be indicative of inefficiencies in the performance of a surgical gesture or task, we examine the correlation between their occurrence and the GRS score and subscores. For each trial, we count the number of inverse MPs and their total durations. This summation includes inverse MPs in all gestures in a trial even if it was not considered in the analysis above (e.g., G1), and was not subject to manual review to remove instances when the inverse MP may not have been seen. We then calculate the Spearman correlation coefficient (ρ) between the total number of inverse MPs or their total durations, and the GRS scores or subscores.

4.3.2 Results

For each gesture, we examine the variety of MP sequences in comparison to the surgeon-defined MP sequences shown in Table 4.7. As an example, the vertical axis of Figure 4.17 lists all of the different MP sequences that comprise G2 in the Suturing task. From these graphs, we observed that many of the MP sequences contained strings of MPs that effectively negated each other, as highlighted in the red boxes in Figure 4.17, so we identify them as inverse MPs and conduct further analysis on them in the following sections.

MPs at the Boundaries of Gestures

We also note that several gestures contain MPs from the surgeon-defined MP sequence of gestures that are before or after them according to the gesture grammar graphs in [52]. Table 4.8 lists the common MPs from preceding or following gestures. For the Needle Passing task, the surgeon-defined MP sequence of G2 includes Push(Needle, Fabric), so there were fewer boundary issues with MPs from gestures before and after G2 compared to Suturing. Some of the MP sequences contained partial or full repetitions of the MPs in the surgeon-defined sequences in almost all of the gestures except G6 in Suturing and Needle Passing, and G13 in Knot Tying, indicating the gesture may have been attempted twice. These MPs are important because machine learning models usually struggle at the transitions between gestures and future work should explore if the above boundary inconsistencies in the training data is contributing to this poor performance.

Examining the sequence of MPs in gestures can also help identify gesture annotation errors. [25] listed a dozen amendments they made to the gesture labels for Suturing and we find that four of them contain MPs from the gesture immediately after the original gesture and two of them do not contain all of the MPs in the labeled gesture. For example, in the second trial of Suturing performed by subject C, the MP sequence for the G6 at frames 1506-1685 is Release(R, Needle), Pull(L, Needle), Grasp(R, Needle), Release(L, Needle). According to Table 4.7, the MP sequence for G6 includes Release(R, Needle), Pull(L, Needle), but the following Grasp(R, Needle) and Release(L, Needle) actually belong to G4 which is the correction given in [25].

Additionally, gesture annotation errors could also be found using inverse MPs since all gesture clips from Needle Passing with the Touch/Untouch(Needle, Ring) inverse MPs that were labeled G4, G6, and G8 were found to be labeling errors and should have been labeled as G2. Thus, using the surgeon-defined MP sequences for each gesture from Table 4.7 could help review gesture labels since Touch(Needle, Ring) is an important part of G2.

Table 4.8: Common MPs from preceding or following gestures found in the MP sequences of each gesture.

Task	Gesture	MPs from the Preceding or Following Gesture
Suturing	G2	Release(L, Needle) from the G8 before and/or the Push(Needle, Fabric) from the G3 after
	G3	Release(R, Needle) of the G6 after
	G6	Grasp(R, Needle) of the G4 after
Needle Passing	G3	Release(R, Needle) in the G6 after
	G4	Release(L, Needle) in the G2 after
	G6	Grasp(L, Needle) from the G3 before and/or the Grasp(R, Needle) from the G4 after
	G8	Release(L, Needle) from the G2 after
Knot Tying	G12	Pull(L, Thread) from the G13 after
	G14	Pull(L, Thread) and Pull(R, Thread) from the G15 after
	G15	Release(L, Thread) and Release(R, Thread) from the G11 after

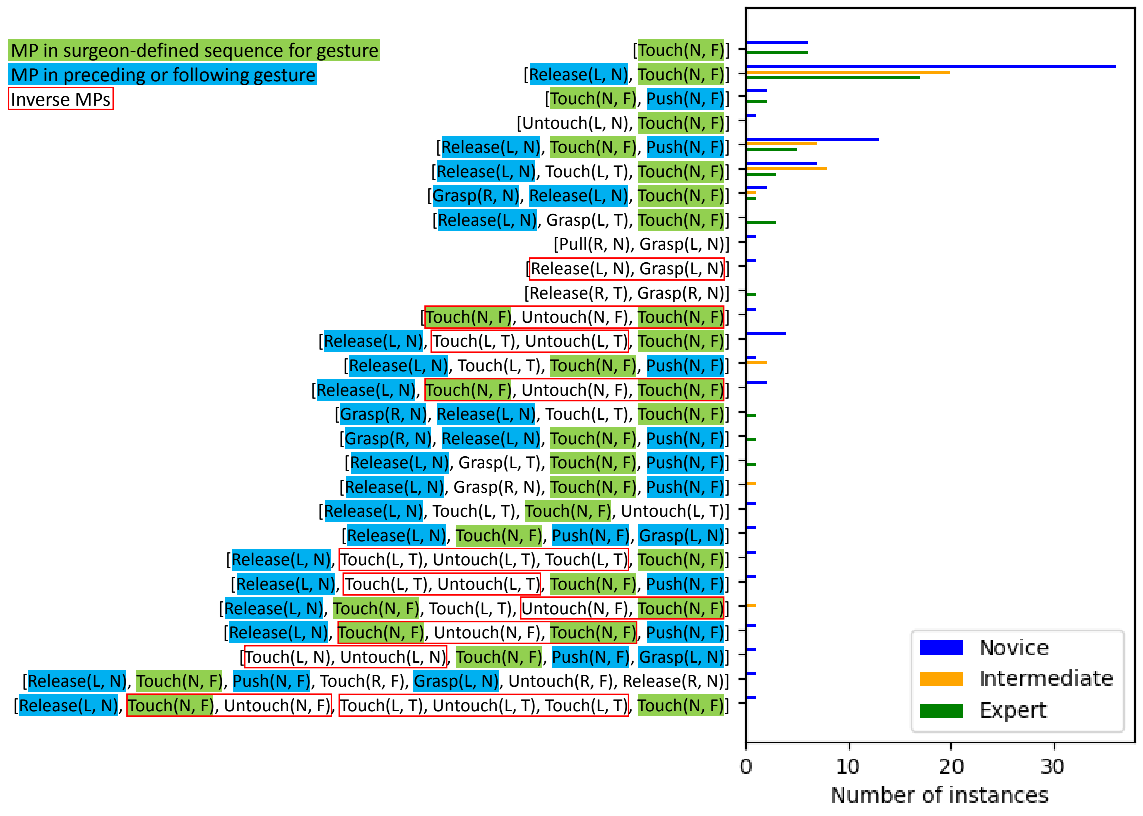


Figure 4.17: Number of instances of each MP sequence in G2 of Suturing where F = Fabric, N = Needle, and T = Thread.

MP State Graphs of Gestures

The MP sequences in each gesture can also be visualized in state transition diagrams such as those in Figure 4.18 which show how the different experience levels used MPs to perform G3 in Suturing. In these graphs, we see that the surgeon-defined sequences are part of the dominant path in each graph, but they show variation with task and experience level. Additional MP state graphs for all tasks, gestures, and expertise levels are shown in Appendix C.

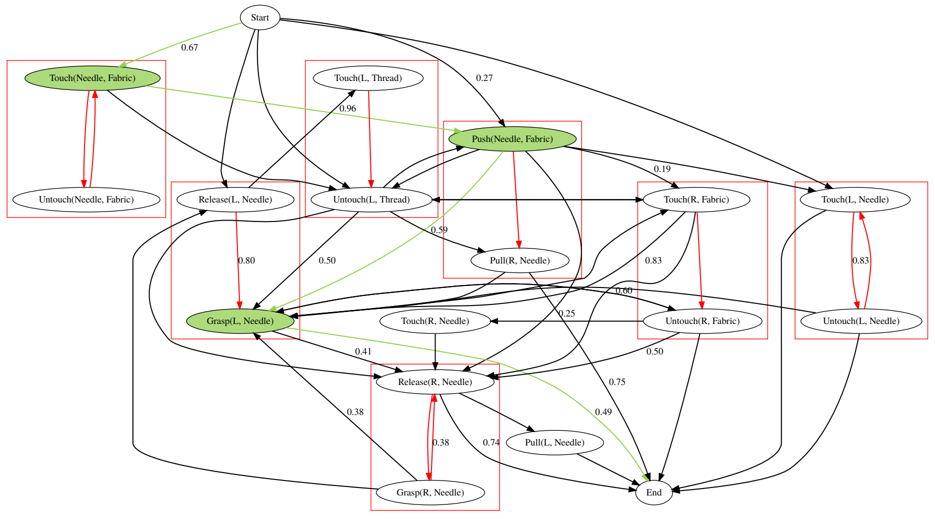
Furthermore, G3 corresponds to surgeme #3 examined in [13] which they found was modeled with five dexemes a-e. They note that dexemes a, b, and c show the right tool driving the needle which best aligns with the Push(Needle, Fabric) MP. Dexemes d and e show the left tool reaching and grasping the needle, respectively, and would thus align with the Touch(L, Needle) and Grasp(L, Needle) MPs, which agrees with the surgeon-defined MP sequence for G3 in Suturing listed in Table 4.7.

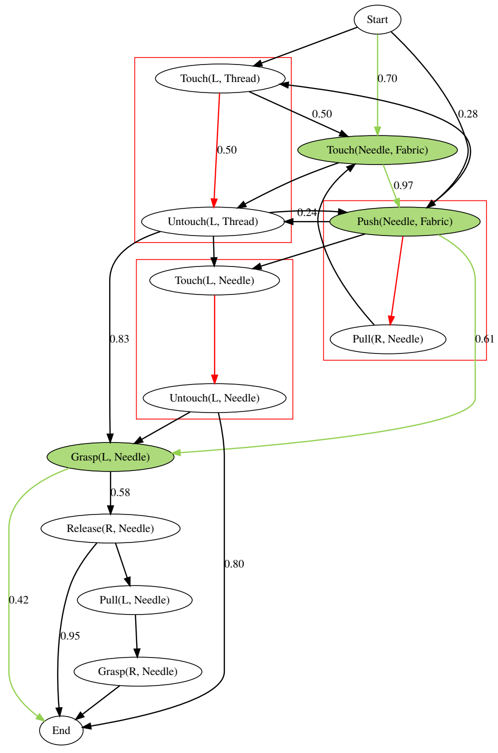
Inverse MPs

The number of inverse MPs by type is shown in Table 4.9, and the number of gestures containing one or more inverse MP is shown in Tables 4.10 and 4.11 by gesture and expertise level, respectively, after subtracting the number of instances not seen in the manual review. These instances could have been due to annotation errors in the MP labels or the boundaries of the gestures because the gesture clips that were reviewed were generated using the start and end frames from the JIGSAWS labels. This is because all MPs that overlapped with a gesture were included in the extracted MP sequence and could be detected as part of inverse MPs, but since the end of the MP wasn't within the boundaries of the gesture, it was not seen in the reviewed gesture clip.

Table 4.9: Number of inverse MPs in each gesture where F/R = Fabric or Ring.

Inverse MP	G2	G3	G4	G6	G8	G12	G13	G14	G15	Total
Touch(L, Needle) Untouch(L, Needle)	1	10	0	2	3	-	-	-	-	16
Touch(R, Needle) Untouch(R, Needle)	1	0	7	2	4	-	-	-	-	14
Grasp(L, Needle) Release(L, Needle)	2	8	2	6	12	-	-	-	-	30
Grasp(R, Needle) Release(R, Needle)	3	6	3	1	2	-	-	-	-	15
Touch(L, Thread) Untouch(L, Thread)	7	3	0	2	0	5	0	0	2	19
Touch(R, Thread) Untouch(R, Thread)	0	0	1	8	0	3	0	15	8	35
Grasp(L, Thread) Release(L, Thread)	0	0	0	1	0	17	1	3	9	31
Grasp(R, Thread) Release(R, Thread)	0	0	0	0	0	6	0	2	13	21
Touch(L, F/R) Untouch(L, F/R)	3	2	0	0	0	-	-	-	-	5
Touch(R, F/R) Untouch(R, F/R)	0	0	1	7	0	-	-	-	-	8
Grasp(L, F/R) Release(L, F/R)	3	0	0	0	0	-	-	-	-	3
Grasp(R, F/R) Release(R, F/R)	0	0	0	0	0	-	-	-	-	0
Touch(Needle, F/R) Untouch(Needle, F/R)	49	2	2	2	1	-	-	-	-	56
Push(Needle, F/R) Pull(R, Needle)	0	4	0	0	0	-	-	-	-	4
Total	69	35	16	31	22	31	1	20	32	257


(a) Novice


(b) Intermediate

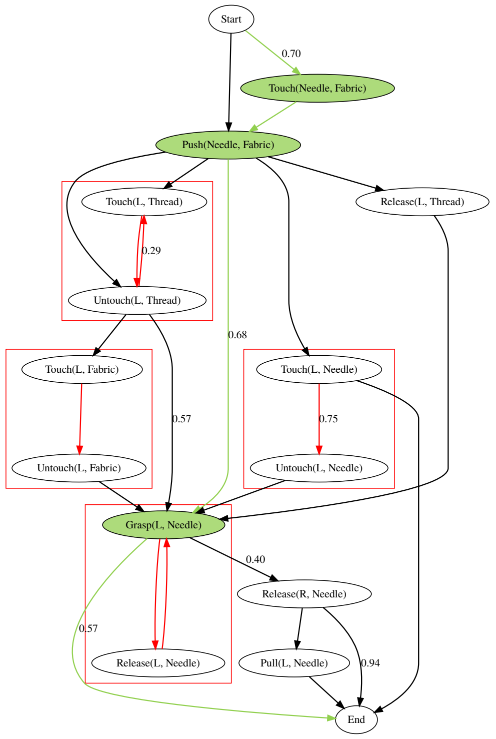

(c) Expert

Figure 4.18: State transition diagrams for Suturing G3 as performed by (a) novices, (b) intermediates, and (c) experts with the surgeon-defined MP sequence in green and inverse MPs in red.

Categories of Inverse MPs

During our manual review, we observed that the inverse MPs could be broadly categorized as relating to depth perception issues, poor positioning or orientation of the needle or suture in the graspers, or incidental actions to adjust the environment.

Depth perception issues were characterized by overshoot in the movement towards an object, running into an object when reaching for another object, or multiple attempts to grab it. This appears as Touch/Untouch(L, Needle) inverse MPs in G3 while reaching for the needle with the left tool after it emerged from the fabric or ring, or in G14 which had the most instances of Touch/Untouch(R, Thread) in Table 4.9 where surgeons struggled to grasp the suture on the left side of the knot due to poor depth perception or occlusion by one of the tools.

Poor positioning or orientation of the needle or suture was a significant contributor to inverse MPs where they are used as recovery actions to fix the position or orientation of the needle or suture from previous gestures. When suturing, the needle should be held with the plane of its curve perpendicular to the tool and about two-thirds from the tip of the needle. The needle tip must also be inserted into the fabric at about 90 degrees. Holding the needle parallel to its curve or inserting it at other angles can result in difficulty driving the needle through the fabric leading to inverse MPs to correct or re-attempt the gesture. Deviations from these proper positions and orientations were responsible for many of the Grasp/Release(L, Needle) inverse MPs in G3 where the left tool grasped the needle trying to adjust how the needle was held while the right tool was trying to drive it through the fabric. This is clinically significant because it can cause excessive strain on the tissue leading to tearing or bleeding. In Section 4.2.3, it was noted that G3 in Suturing had the most "Multiple attempts" errors for similar reasons. In addition, all four occurrences of Push(Needle, Fabric) Pull(R, Needle) were in G3 of the Suturing task where the needle had to be removed from the fabric since poor orientation of the needle made it too difficult to complete the throw as shown in Figure 4.19.

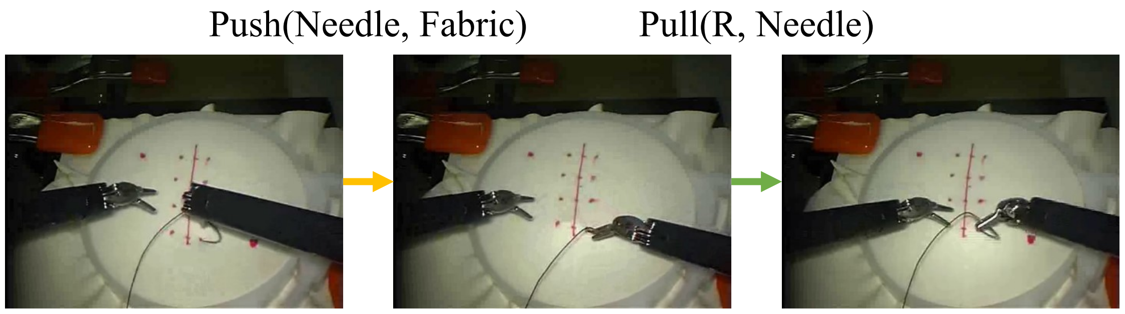


Figure 4.19: Example of a Push(Needle, Fabric) Pull(R, Needle) inverse MP in Suturing.

In Needle Passing, all three expertise levels struggled with accurately positioning the needle because the eye of the ring was a small target leading to a high occurrence of Touch/Untouch(Needle, Fabric/Ring) inverse MPs in G2 as shown in Table 4.9. These were a large contributor to the 40.2% of G2 instances in Needle Passing that contained inverse MPs from Table 4.10 and is also consistent with the results from Section 4.2.3 where G2 had the most "Multiple attempts" errors in Needle Passing.

When tying knots, if the left tool grasps the suture too close or too far from the knot, there will be too little or too much slack to make a properly sized wrap, thus making it difficult for the right tool to grasp and pull the end of the left suture through the wrap made in G14. The right tool should grasp the other end of the suture near the end, otherwise it might require more than one re-grasp and pull to get all of it through the knot like in G15 of Knot Tying. If the suture is grasped too far away from the knot, one or both tools will have to release it and grasp it closer to the knot and perform a second pull to fully tighten the knot. This was responsible for many of the Grasp/Release(R, Thread) and Grasp/Release(L, Thread) inverse MPs which were present in most of the 28.8% of G15 instances that had inverse MPs in Table 4.10.

Similarly, inverse MPs also occurred in additional exchanges of the needle or suture that were used to manipulate the object's position in the tools that could have been accomplished with fewer MPs. For example, correctly orienting the needle in G8 of Suturing could be accomplished with a single exchange, but Table 4.10 shows that this gesture has the highest percentage of gestures with inverse MPs. Many of these were

Grasp/Release(L, Needle) inverse MPs with most from novices. Also, G12 can be efficiently performed by grasping the suture in the right tool and passing it to the left as in the surgeon-defined MP sequence shown in Table 4.7. But there was a significant number of occurrences of Grasp/Release(L, Thread) that were mainly from novices and accounted for most of the 31.4% of G12 instances with inverse MPs in Table 4.10.

Adjustment of the environment was another reason for the occurrence of inverse MPs. For example, inverse MPs were used to move or hold the suture out of the way, as shown in Figure 4.20, or to hold the ring in place when positioning the needle in G2 of Suturing and Needle Passing. Specifically, all seven occurrences of Touch/Untouch(L, Thread) were performed by novices in the Suturing task while trying to move the suture out of the way when positioning the tip of the needle at the dot on the fabric. But, they either moved the tool away from the suture or the suture slipped out from beneath the tool resulting in the inverse MPs.

It is also interesting to note that there was only one inverse MP found in G13. This is because G13 involved wrapping the suture around the right grasper and was a difficult motion represented by Pull(L, Thread). However, its inverse where the suture became unwrapped from around the tool was represented by the same MP, and thus was not detected as an inverse MP. In the manual review, several clips of G14 also showed instances where the wrap slipped off of the right tool, but these contained different types of inverse MPs that either contributed to the suture coming unwrapped or tried to re-wrap the suture.

Table 4.10: Number and percentage of gesture clips with one or more inverse MPs in each gesture over the total number of instances of that gesture.

Task	G2	G3	G4	G6	G8	G12	G13	G14	G15
Suturing	12/166 (7.2%)	21/164 (12.8%)	7/119 (5.9%)	17/163 (10.4%)	13/48 (27.1%)				
Needle Passing	47/117 (40.2%)	10/111 (9.0%)	7/83 (8.4%)	12/112 (10.7%)	4/28 (14.3%)				
Knot Tying						22/70 (31.4%)	1/75 (1.3%)	17/98 (17.3%)	21/73 (28.8%)
Total	59/283 (20.8%)	31/275 (11.3%)	14/202 (6.9%)	29/275 (10.5%)	17/76 (22.4%)	22/70 (31.4%)	1/75 (1.3%)	17/98 (17.3%)	21/73 (28.8%)

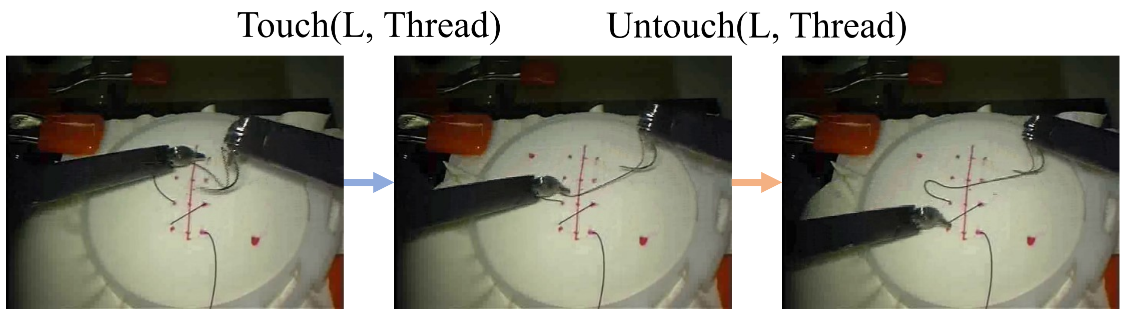


Figure 4.20: Example of a Touch(L, Thread) Untouch(L, Thread) inverse MP in Suturing.

Table 4.11: Number and percentage of gesture clips with one or more inverse MPs in each gesture over the total number of instances of that gesture by experience level.

Task	N	I	E	Total
Suturing	45/342 (13.2%)	13/161 (8.1%)	12/157 (7.6%)	70/660 (10.6%)
Needle Passing	27/160 (16.9%)	22/138 (15.9%)	31/153 (20.3%)	80/451 (17.7%)
Knot Tying	39/139 (28.1%)	14/87 (16.1%)	7/90 (7.8%)	61/316 (19.3%)
Total	111/641 (17.3%)	50/386 (13.0%)	50/400 (12.5%)	211/1427 (14.8%)

Relationship between Inverse MPs and Skill Level

Occurrences of Inverse MPs by Expertise Level Table 4.11 shows the number and percentage of gesture clips with one or more inverse MPs over the total number of clips by expertise level. When considering expertise level, we use percentages due to the data imbalance of having four novices, two intermediates, and two experts. Overall, and for the Suturing and Knot Tying tasks, the percentage of gestures with inverse MPs decreased with increasing expertise level. However, this trend is not seen in the Needle Passing task where experts had the highest percentage of gestures with inverse MPs. This could be because Needle Passing is a more difficult task than Suturing since greater accuracy is required to thread the needle through the eye of the ring. This also makes it less surgically realistic and we found that more than half of the inverse MPs in the Needle Passing task were Touch/Untouch(Needle, Ring) inverse MPs from experts which was even greater than the number from novices.

MP State Graphs for Expertise Levels By examining the MP state graphs of each task and gesture for the different experience levels such as the ones shown in Figure 4.18, we observed that in Suturing and Knot Tying, novices tended to have more states and more complicated graphs than experts and intermediates similar to [35], but in Needle Passing, experts usually had more complicated graphs than novices. Additional MP state graphs for all tasks, gestures, and expertise levels are shown in Appendix C.

Table 4.12: Spearman's correlation coefficient (ρ) between the total number of inverse MPs in a trial and GRS scores and subscores.

	Suturing		Needle Passing		Knot Tying	
GRS Subscore	ρ	p-value	ρ	p-value	ρ	p-value
Respect for Tissue	-0.63	<0.001	-0.13	0.517	-0.63	<0.001
Suture & Needle Handling	-0.52	<0.001	-0.15	0.432	-0.46	0.005
Time & Motion	-0.64	<0.001	**-0.20**	0.296	-0.64	<0.001
Flow of Operation	-0.58	<0.001	-0.06	0.775	**-0.65**	<0.001
Overall Performance	**-0.65**	<0.001	-0.14	0.471	-0.61	<0.001
Quality of Final Product	-0.50	0.001	-0.03	0.890	-0.59	<0.001
GRS Score	**-0.65**	<0.001	-0.19	0.338	-0.63	<0.001

Correlation between Inverse MPs and GRS Scores Tables 4.12 and 4.13 show the Spearman's correlation coefficient (ρ) between the number and total duration of inverse MPs in a trial with the GRS scores and subscores. We find that there are significant and strong negative correlations for all GRS scores and subscores in the Suturing and Knot Tying tasks with $\rho < -0.5$ except for the Suture & Needle Handling subscore in Knot Tying which has correlation coefficients of -0.46 and -0.49. This means that a greater number of occurrences and longer durations of inverse MPs can be indicative of lower skill. However, in the Needle Passing task, the correlation coefficients are only between -0.06 and -0.33 with larger p-values indicating a weaker and not significant correlation.

Table 4.13: Spearman's correlation coefficient (ρ) between the total duration of inverse MPs in a trial and GRS scores and subscores.

	Suturing		Needle Passing		Knot Tying	
GRS Subscore	ρ	p-value	ρ	p-value	ρ	p-value
Respect for Tissue	-0.59	<0.001	-0.26	0.188	-0.58	<0.001
Suture & Needle Handling	-0.61	<0.001	-0.28	0.151	-0.49	0.002
Time & Motion	**-0.71**	<0.001	-0.31	0.113	**-0.71**	<0.001
Flow of Operation	-0.68	<0.001	-0.20	0.305	-0.66	<0.001
Overall Performance	-0.66	<0.001	-0.32	0.100	-0.63	<0.001
Quality of Final Product	-0.57	<0.001	-0.22	0.258	-0.61	<0.001
GRS Score	**-0.71**	<0.001	**-0.33**	0.087	-0.65	<0.001

4.4 Related Work

4.4.1 Error Detection

State-of-the-art RAS systems and simulators are designed with data logging mechanisms to collect system logs, kinematics, and video data from surgical procedures. The recorded data has been mostly used for offline surgical skill evaluation [146, 149, 150], with the aim of improving surgeons' performance and making evaluations objective and scalable.

Current work focuses on using surgical data for skill assessment and distinguishing between expert and novice surgeons. Methods for the objective assessment of robotic technical skills can be classified into two general categories: manual assessment and automated assessment. Manual skill evaluation is usually performed globally, assessing performance over an entire demonstration using frameworks such as OSATS (Objective Structured Assessment of Technical Skills) [148], R-OSATS [151], GOALS (Global Operative Assessment of Laparoscopic Skills) [152], and GEARS [153]. However, manual assessment methods are subjective, time consuming, cognitively demanding, and prone to errors [150]. In response, automated assessment methods utilizing kinematic, video, and system event data [75] are being developed to provide objective and quantitative metrics [150,154], and explainable feedback [41]. The metrics used for surgical skill assessment can be classified into three broad categories of: (i) efficiency (e.g., path length, completion time), (ii) safety (e.g., instrument collisions [155], instruments out of view, excessive force, needle drops, tissue damage [156]), and (iii) task/procedure-specific metrics (e.g., task outcome metrics, camera movement, energy activation [157]). While most previous works focused on skill evaluation for distinguishing between expertise levels, less attention has been paid to identifying specific erroneous surgical activities that contribute to suboptimal performance and potential safety-critical events.

Towards error detection, [18] was an early work that trained models to detect errors. Then, [137] trained a contrastive regression transformer for skill assessment that was also able to detect those errors. The most closely related works have proposed objective gesture-based checklists for laparoscopic and robot-assisted suturing [142] and [158], as well as general and custom rubrics for the evaluation of human errors [133] and technical errors [159] in laparoscopic surgery. Another closely related work is [15] which defines ineffective, effective, and erroneous labels for dissection gestures and found that novices performed the greatest proportion of ineffective gestures, intermediates performed the greatest proportion of erroneous gestures, and experts performed the greatest proportion of effective retraction gestures. They also found differences between experience levels in terms of gesture selection and efficacy, but this study was limited to only videos of a dissection task in real surgery and for the purpose of classifying experience level. Other related works on errors in RAS mainly focused on analyzing adverse events and system malfunctions as reported by the surgical teams and institutions [136].

Table 4.14: Related work on gesture-aware skill assessment.

Method	Data Modalities	Skill Metrics [1]	Tasks	Method Performance [3]	Evaluation Method [4]
Transformers [160]	Clinical features	1-year EF recovery	RARP	AUC=0.69	Monte Carlo random folds
	Gesture sequences			AUC=0.77	
	Combined			AUC=0.75	
Logistic regression [160]	Clinical features			AUC=0.65	
	Gesture sequences			AUC=0.68	
	Combined			AUC=0.67	
Contra-Sformer [137]	Video	GRS score	Suturing Needle Passing Knot Tying	ρ=0.65, MAE=2.58 ρ=0.71, MAE=3.17 ρ=0.69, MAE=1.39	LOUO
			Across tasks	ρ=0.68, MAE=2.38	
C3D-MTL-VF [138]	Video	GRS score	Suturing Needle Passing Knot Tying	ρ=0.0.69 ρ=0.86 ρ=0.83	LOUO
		N/I/E	Suturing Needle Passing Knot Tying	Acc=100.0% Acc=100.0% Acc=97.5%	LOSO
IMTL-AGF [138]	Video and gestures	Intermediate scores	Suturing Needle Passing Knot Tying	ρ=0.98 ρ=0.98 ρ=0.96	LOUO
Discriminative interpretable patterns [42]	Kinematics	N/I/E	Suturing Needle Passing Knot Tying	Acc=89.74% Acc=96.30% Acc=61.11%	LOSO
			Peg Transfer	Acc=83.33%	LOO
		N/E	Microsuturing	Acc=85.19%	
SD-Net [161]	Video	N/I/E	Suturing	Acc=90.5%, Edit=90.6, F1@10=93.5	LOUO
DCC-CSM [162]	Kinematics and gestures	Binned GRS score	Suturing [2]	Average Acc=84.51%	LOUO
HMM [35]	Kinematics and gestures	N/I/E	Suturing [2]	Average Acc=70%	LOO

[1] N/I/E = Novice/Intermediate/Expert.
[2] Not from the JIGSAWS [43] dataset.
[3] ρ = Spearman's correlation.
[4] LOO = Leave-One-Out, LOUO = Leave-One-User-Out, LOSO = Leave-One-Supertrial-Out.

4.4.2 Activity-Aware Skill Assessment

Skill assessment has been performed at different levels of the surgical hierarchy. Automated methods enable the subdivision of demonstrations into subtasks or gestures, and to base performance assessment and technical skill evaluation on the quality and/or sequence of these components as proposed in [52], [12], and [149]. In addition, [163] trained a model for step, task, and action recognition that learned inter- and intra-relations between units of activity at different levels.

Many works have focused on specific activity levels. At the gesture level, [37] found that experts used fewer gestures, made fewer errors, and had more predictable transitions than novices. Gesture-level error detection is an emerging subfield [8] that can augment skill assessment since several works have shown a correlation between skill level and errors [15, 164, 165]. Many works have found significant differences between gestures performed by expert and novice surgeons, so some gestures are more indicative of skill than others [13, 15, 34–36, 80]. Table 4.14 lists related works that leverage knowledge of gestures in performing skill assessment by using gestures as inputs to their models or by training models for specific gestures. Efforts towards increasing the interpretability of these models have used data-driven methods to identify problematic gestures that contribute to low skill scores [137, 138]. However, there is a large variability in the correct performances of gestures due to style or expertise and it is impossible to define all errors or incorrect performances of gestures, thus limiting the effectiveness of supervised machine learning models trained on small datasets.

Skill assessment has also been performed at the motion level using both statistics and data-driven models. [34] and [38] found differences between experts and novices in the numbers and durations of actions performed during laparoscopic simulation and dry lab tasks. [36] examined efficiency in microsurgical suturing based on time and the cosine similarity of movement patterns. But, these works were limited to statistical analyses of the motions without considering patterns in the motion sequences. Models such as Hidden Markov Models (HMMs) have also been used for motion-level skill assessment where the motions were modeled with hidden states and learned from data [12, 13]. [13] showed that the sequence of states visited was indicative of skill level and confirmed this by measuring the average edit distance between sequences for gestures performed by surgeons of different skill levels. Interestingly, [35] found that gesture-level HMMs for skill assessment were better than task-level models, and that building the task-level models using gestures as the states was better than learning the states from unlabeled kinematic data. This motivates a lower, motion-level analysis of surgical activity. Gestures have also been modeled as strings comprised of finer-grained activities mapped to an alphabet in [42] and [162], while [90] and [91] use bottom-up approaches that learn finer-grained motions as an intermediate step in recognizing surgical gestures. However, motions learned from data in an unsupervised manner are not easily human interpretable and do not correspond with labeled units of surgical activity [35].

4.4.3 Interpretable Feedback

Recent efforts have been directed at increasing the interpretability of surgical skill assessment. Several methods use metrics such as energy usage, time, path length, and economy of motion, as inputs to their models so that the outputs are explainable and objective [39, 150, 166–168], while others identify or localize the parts of the trial that contribute to the predicted score by visualizing trajectories or features [40–42, 48]. [138] proposed a model for gesture recognition and skill assessment with running intermediate skill scores to identify problematic gestures. But, providing gesture-specific feedback based on identifying patterns in finer-grained units of surgical activity has not been explored yet.

4.5 Conclusion

In conclusion, we presented a new rubric and method for the objective evaluation of RAS procedures with a focus on gesture- and task-specific executional and procedural errors. We used the proposed rubric to evaluate dry lab demonstrations of Suturing and Needle Passing tasks. Our analysis identified the most common error modes and their correlations with skill levels and demonstration times as well as important error-specific kinematic parameters that distinguish erroneous gestures. This study is a step towards developing methods for automated error detection and providing real-time context-dependent feedback for performance improvement. Future work will extend the error rubric and analytical methods to larger datasets and other surgical tasks.

We also analyzed surgical activity at the gesture and motion levels and found that the sequence of MPs could help detect labeling errors in surgical gestures. We defined and identified inverse MPs that are motion-level patterns often used as recovery actions to correct the position or orientation of objects. Although these inverse MPs are sometimes required to perform gestures, we found that the number and duration of inverse MPs strongly correlates with lower GRS scores for two dry lab surgical tasks. Recent work [169] found only minimal correlation between self-proclaimed expertise and GRS scores in the JIGSAWS dataset, but subjects with higher GRS scores were more similar in their performance of certain tasks. Thus, future work could investigate how the MPs used to perform gestures may vary with GRS scores. Future work also includes developing a pipeline for interpretable surgical skill assessment that can show inverse MPs identified in the MP sequence output from an action recognition model. Showing video clips of these inefficient or problematic motions will provide specific explanations of the causes of lower performance scores. However, better performing action recognition models must first be developed to obtain reliable and precise motion sequences and support this method of skill assessment.

In addition, the methods and insights from our analysis can be leveraged in the design of simulators and training tasks to improve training and skill assessment with more specific and interpretable feedback. Specifically, the safety of robotic surgery could be improved with the development of safety monitoring systems that are aware of the tools, objects, and tissues in the surgical scene, the surgical process, and the actions the surgeon is performing. Mechanisms for context-specific monitoring [17] and virtual coaching will be enabled by the advanced sensing and computing technology, artificial intelligence, and data science driving the development of the next generation of RAS systems. These will provide early and context-specific feedback to surgeons on potentially suboptimal or unsafe motions that could help improve performance scores in training, prevent safety-critical events in actual surgery [18], and improve the safety, efficiency, and quality of care [170].

Chapter 5

Conclusions and Future Work

5.1 Conclusions

Surgical robots are complex cyber-physical systems that are driving the development of new features and technologies to improve efficiency and patient safety. But, the development of these features is limited by small datasets with low diversity, the poor performance of black box activity recognition models in fine-grained segmentation, and limited attention to identifying errors at the fine-grained gesture and motion primitive levels.

In this dissertation, we proposed a formal framework for the fine-grained modeling of surgical tasks using context and motion primitives which enabled the modeling of tasks as finite state machines and aggregation of data from different tasks and datasets. Our method for labeling context achieved high agreement between non-expert annotators and expert surgeons and was used to create objective and consistent labels for the COntext and Motion Primitive Aggregate Surgical Set (COMPASS) which nearly triples the amount of kinematic and video data for comparative analysis. Using video data, we leveraged image segmentation models in inferring surgical context, and using kinematic data, we utilized the larger variety of tasks in the COMPASS dataset to evaluate the task-generalizability of surgical activity recognition models using our novel Leave-One-Task-Out (LOTO) cross validation method. We found that aggregating data across datasets supports the task-generalization of motion primitive recognition models. Finally, we identified and analyzed activity-specific errors and patterns in surgical gestures and motion primitives. We used our gesture-specific rubric for executional errors to label and analyze errors, and identified inverse motion primitives in gestures and tasks. Our analysis found that different gestures have different predominant error modes. Inverse motion primitives were often used as recovery actions to correct the position or orientation of objects and may be indicative of other issues such as with depth perception. We found that the occurrence of errors and inverse motion primitives was correlated with objective measures of skill level and thus can be used as part of interpretable feedback in surgical skill assessment.

5.2 Future Work

The data, methods, and models presented in this dissertation can support many areas of future work within and beyond the field of robotic surgery.

- **Generalization to real surgery:** We presented a framework for modeling surgical tasks with context and motion primitives in order to aggregate data from different tasks and datasets and create the COMPASS dataset. Although the tasks in COMPASS are dry lab exercises, the framework can be extended to model surgical activities in real operations by increasing the number of objects encoded in the context, adding task-specific context state variables, and defining additional verbs that change those states. Whereas previous tasks were all performed with graspers, specific tools could also be incorporated into the framework by defining tool-specific verbs such as Cut for scissors or Monopolar Cautery for tools with monopolar electrocautery. Extending the framework to include data from real surgery can enable additional analyses of how well models trained on dry lab data may be able to transfer to surgical operations as a parallel to how surgeons advance from dry lab exercises to real patients.

- **Application to human activity modeling and recognition:** Surgical gestures and motion primitives are only very specific examples of human activity. Recognizing human activities can be important in other scenarios like everyday tasks such as shopping [171] and preparing food [172, 173], interacting with robots in a cooperative task [53], or treatment actions and interventions taken by health professionals such as nurses, physicians, and first responders. In these scenarios, context could represent the objects in contact with or held by the left and right hands of the human and/or robot, and task- or tool-specific verbs could be defined to describe progress. Applying the framework proposed in this dissertation could enable dataset aggregation in the more general field of human activity recognition since there are many similarities between basic activities in different tasks such as reaching for or grasping a task-specific object.

- **Activity-aware safety monitoring:** The error rubric and analytical methods proposed in this dissertation can be extended to larger datasets and other surgical tasks by defining additional gesture-specific error modes and generating ideal models of tasks using context and motion primitives. Furthermore, each motion primitive may be subject to constraints such as virtual fixtures and no-go zones [63, 64] that can be used as properties for runtime context-aware monitoring to improve the safety of robotic surgery. Combined with the insights from our activity-aware analysis of surgical tasks, this can inform the development of safety monitoring systems and support interpretable feedback. However, there are several challenges in implementing an activity-aware safety monitoring system such as the following:

    1. Surgical scene segmentation and workflow recognition models need to be reliable and robust so that the safety monitor is provided with accurate context.

    2. Error detection models should have high true positive rates but low false positive rates because it is important to catch errors before they result in

adverse events, but too many alerts may lead to fatigue and cause surgeons to ignore the system.

3. The surgical scene segmentation, workflow recognition, and error detection models must be implemented in real-time to provide timely alerts about potential errors to surgeons and enable corrective actions on the robot.

Appendix A

Data Collection System

A.1 Introduction

Although the da Vinci Surgical Systems from Intuitive Surgical are currently the only FDA-approved surgical robot used in the operating room, other companies are developing systems to compete with it. For example, the Hugo robot-assisted surgery (RAS) system by Medtronic [174] received approval for urologic and gynaecologic procedures in Europe in 2021. There are also many robots and systems used for research such as the Raven II [175] developed by Applied Dexterity, the Taurus II, and the YuMi robot that was adapted for surgical tasks in [58]. Simulated robots and environments are also used for training and research such as SimNow by Intuitive Surgical, and virtual models of the Raven II [176] and Taurus II [58]. [174] details several more commercial and prototype RAS systems that are in use or under development. Thus, there is a growing variety of RAS systems. Efforts towards facilitating collaboration in the research community among users of different RAS systems have included the development of a Collaborative Robotics Toolkit (CRTK) as a common API for the Raven II and the da Vinci Research Kit (dVRK) [177].

As cyber-physical systems (CPS), these surgical robots still have much in common in terms of how they are controlled. A human surgeon or operator interacts with the surgical robot using a master console where they are provided a 3D view of the

This system was designed and developed in collaboration with S. Yapalparvi, J. Park, and S. Liu.

surgical environment and directly control the movements of the surgical robot using manipulators or controllers. The surgical robot then uses various control systems and methods to execute these commands, which, at a low level, involves forward and inverse kinematic calculations to solve for the signals that should be sent to the actuators of the surgical robot both real and in simulation. These variables comprise the kinematic data recorded from surgical robots that complements the stereoscopic video data of the environment. Having multimodal datasets is important for the improvement of activity recognition [125] and the development of safety and error detection applications since kinematic data is robust to camera occlusions and lens contamination which are common in robotic surgery [17, 30, 31].

However, kinematic data is less often recorded and included in multimodal datasets for research in surgical robotics. This could be due to several factors including accessibility where kinematic data is easily obtainable from open source robots such as the Raven II, but not from proprietary systems such as the da Vinci Surgical Systems. In addition, different robotic systems record different kinematic variables at different frequencies, and in different units, and the video and kinematic data must be synchronized during or after recording. There is much preprocessing that must be performed to make kinematic data consistent and compatible before it can be analyzed. Thus, obtaining kinematic data from surgical robots remains a challenge in creating multimodal datasets for research.

This project addresses the challenge of acquiring multimodal data from a variety of robots by developing a platform-independent Data Collection System (DCS) that records and synchronizes video and kinematic data from the master console. This enables the standardized collection of consistent and compatible data from different RAS systems in both simulated and dry lab settings. The following sections describe the design, operation, and evaluation of the DCS.

A.2 Methods

This section describes the design of the Data Collection System (DCS) including its overall architecture, details of the subsystems for video and kinematic data collection, and data synchronization.

A.2.1 System Architecture

The Data Collection System (DCS) deployed on a da Vinci Surgical System is shown in Figure A.1. It consists of an application running on a laptop that records video data from a video capture card and kinematic data from an NDI trakSTAR system. The DCS application has a front end coded in Python and a back end coded in C++. The front end handles the GUI, initiates and terminates the collection of data from the video capture card and trakSTAR, and automatically synchronizes the video and kinematic data in post-processing. The back end interfaces with the trakSTAR via its API which implements settings from the front end such as sampling rate and optional offsets.

The GUI gathers information about the trial including the type of task that is being performed, an identifier for the subject, the trial number, and the subject's experience level. This information is used to identify individual trials and connect them to specific experience levels such as the number of years in the residency program. The sampling frequency is set to a default of 30 Hz. The front end records this information and combines it with a time stamp to create a unique set of folder and file names for each trial.

Then, the front end begins recording data from the video capture card using the OpenCV library in Python the kinematic data from the NDI trakSTAR. A video capture card was selected for the collection of video data because connections and ports for the display of video data are easily accessible on the master console. For example, the da Vinci Surgical Systems have DVI ports that enable the projection of the endoscopic video on external monitors and displays. Thus, only a DVI to HDMI cable is needed to input the video to the video capture card. On the other hand,

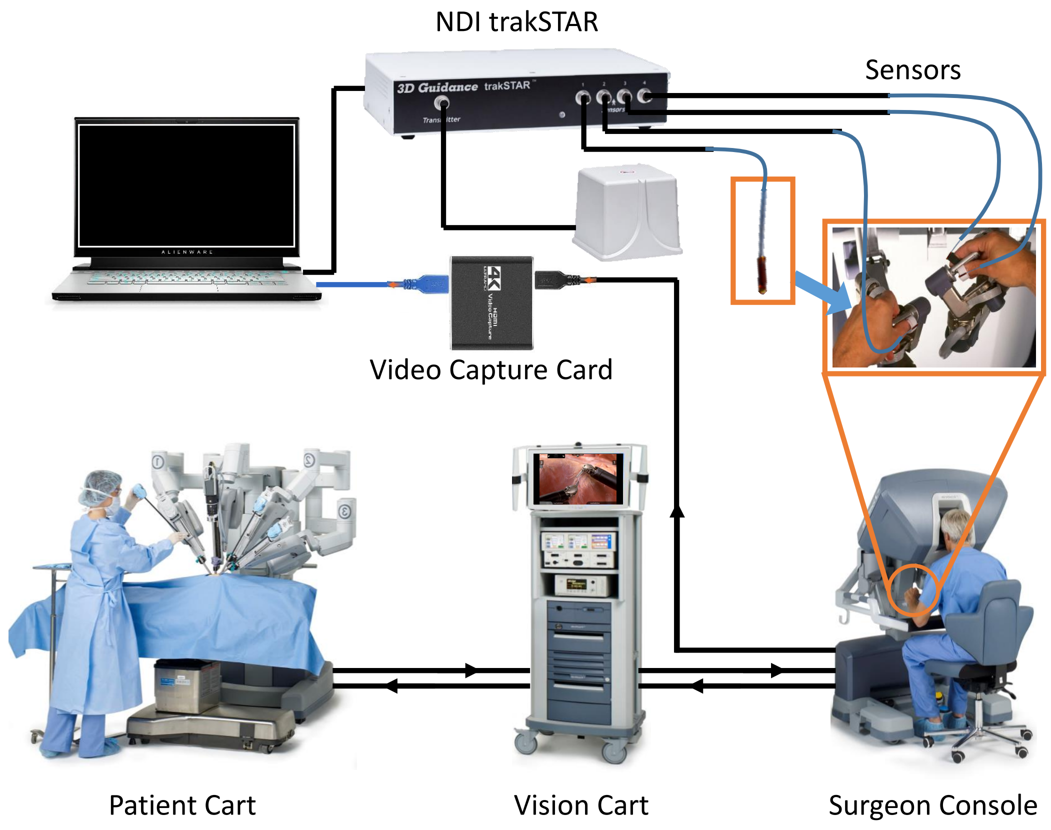


Figure A.1: The Data Collection System deployed on a da Vinci Surgical System. Image sources [178] and [179].

the Raven II has VGA ports, so a separate VGA to HDMI converter is needed to input the video to the video capture card.

The trakSTAR system is an electromagnetic tracking system comprised of an electronics unit, a transmitter, and multiple sensors. Compared to the alternative method of using cameras to capture the movements of the surgeon's hands, electromagnetic tracking is more accurate, does not require the use of markers for visual identification in the images of the video, and is not affected by occlusions. Four Model 180 sensors that are only 2.0 mm wide and 9.9 mm long are temporarily attached to the manipulators of the master console to avoid interference with the surgeon's hands. This configuration enables the collection of kinematic data representative of the surgical robot's internal kinematics due to the direct control of the surgical robot by the surgeon, up to a scaling factor or any constant offsets induced by tool or camera clutching. In analysis, the average of the two sensors on the left and right manipulators respectively are taken to represent the position of the end effector, and the distance between them is normalized to represent the angle of the jaws of the tool.

When the trial is complete, data collection is terminated by the front end, and the video and kinematic data files are automatically synchronized. During data collection, epoch timestamps are recorded for each kinematic sample and video frame. Since the video recordings start later than the kinematic data recordings, the kinematic data files are cropped so that they start with the sample with the timestamp closest to the timestamp of the first video frame.

A.3 Results

A.3.1 Deployment Details

The DCS application runs on an Alienware m15 R4 laptop computer so that it is portable and can be used to record data from different robots and systems in different locations. When surgeons operate, they hold the manipulator with their thumb and middle finger, while their index finger controls the finger clutch for the tool. Thus, the four sensors of the trakSTAR are attached to the manipulators in the following order: 1 is the left middle finger, 2 is the left thumb, 3 is the right thumb, and 4 is the right middle finger. Figure A.2 shows the DCS set up for data collection on a da Vinci simulator.

A.3.2 System Evaluation

We evaluated the DCS by comparing the kinematic data obtained from it to the kinematic data logged by the Raven II surgical robot during trials of Peg Transfer in dry lab. Specifically, we compared the Cartesian x, y, and z coordinates of the left and right end effectors measured by the Raven II to the positions of the left and right manipulators measured by the DCS. The Peg Transfer task is a standard training task in robotic surgery that develops skills in object manipulation and hand coordination.

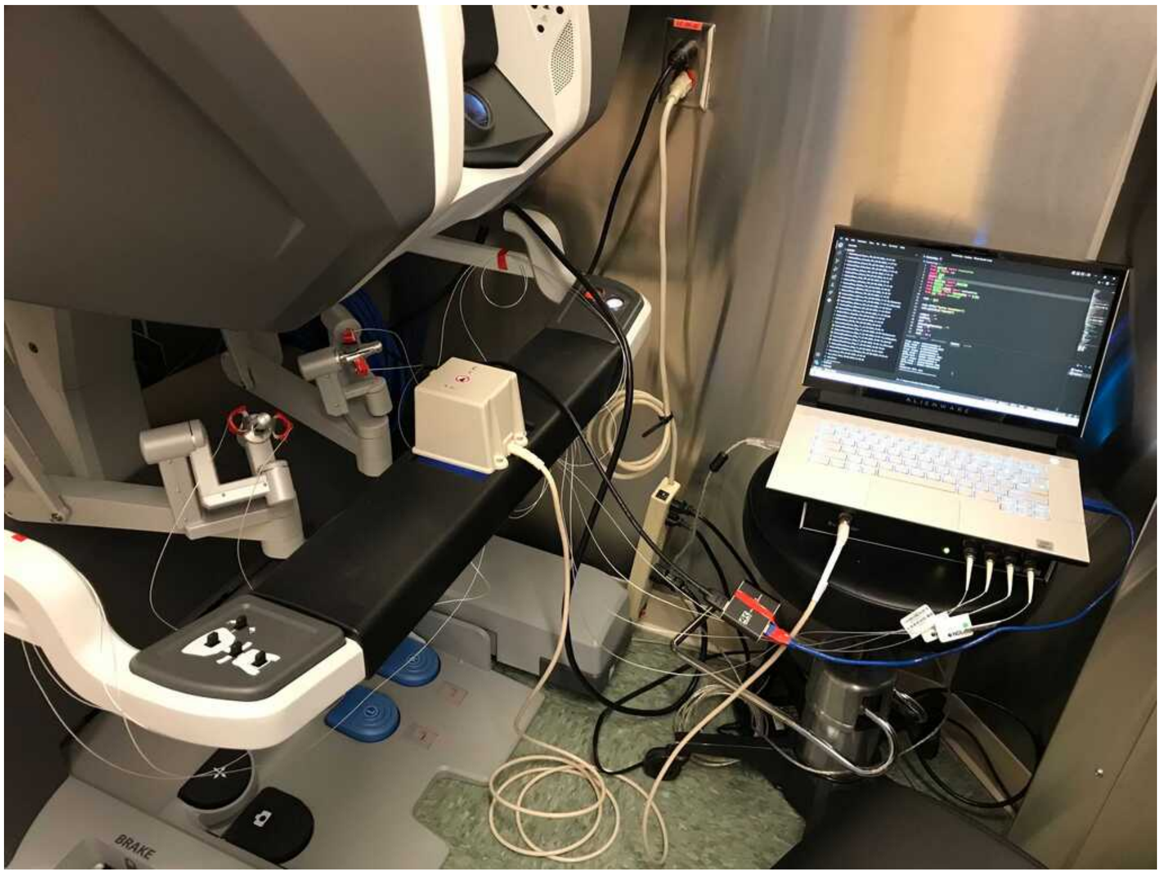


Figure A.2: Data Collection System set up on a da Vinci simulator.

During this task, the surgeon picks up a block, passes it to the other tool, places it on another peg, and repeats this several times per trial as shown in Figure A.3.

We collected data for ten trials of the Peg Transfer task performed on the Raven II surgical robot. However, the DCS did not record kinematic data for one of the trials, so it was removed from the dataset. In addition, the Unix timestamps for the first two trials differed between the Raven II computer and the DCS, so offsets of 77s and 76s were applied to trials 1 and 2 respectively. Since the Raven II recorded the x, y, and z positions of the left and right end effectors while the DCS recorded the x, y, and z positions of four sensors, we took the average position of the sensors on the left manipulator (denoted L_x, L_y, and L_z) and the average position of the sensors on the right manipulator (denoted R_x, R_y, and R_z). This provides a representation of the master console kinematics which should be the the same as or very similar to the kinematics of the surgical robot except for scaling and offset constants.

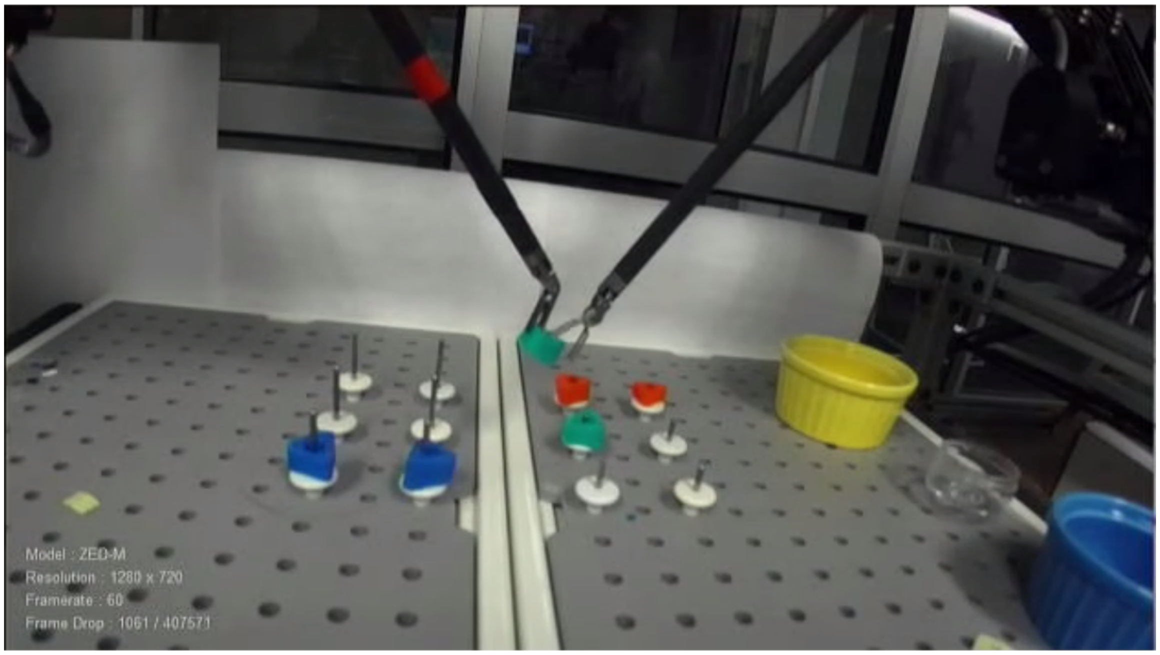


Figure A.3: Peg Transfer performed with the Raven II surgical robot.

Previous testing and calibration of the system mapped the x, y, and z coordinates of the DCS to those of the Raven II as shown in Table A.1. The Raven II kinematics were scaled by a factor of 1/1000 so that all measurements had units of millimeters, and downsampled to a rate of 30 Hz with missing values filled in using linear interpolation.

Table A.1: Mapping between kinematic variables of the DCS and Raven II.

DCS	Raven II
L_x	$field.pos1$
L_y	$-field.pos2$
L_z	$-field.pos0$
R_x	$-field.pos4$
R_y	$field.pos5$
R_z	$-field.pos3$

Both the DCS and Raven II kinematics were cropped to the start and end of the trial based on the first and last clutch commands in the $field.runlevel$ variable in the Raven II kinematics. Specifically, samples at the beginning and end of the trials corresponding with $field.runlevel = 2$ where dropped because the clutch was engaged and the robot was not being operated.

Since use of the clutch pedal was allowed during the trials, this meant that when the clutch pedal of the Raven II was pressed, the robot would be disengaged so that its end effectors would not move, but the manipulators at the master console could still move freely. Thus, the kinematics from the DCS would show movement during this time but no corresponding motion would be seen in the Raven II kinematics. In order to remove the effect of clutching, we first find the parts of the trial during which the clutch pedal is pressed. That is, we detect when $field.runlevel$ transitions from 3 to 2 indicating that the clutch pedal is pressed, and when $field.runlevel$ transitions from 2 to 3 indicating that the clutch pedal is released. When the clutch pedal is pressed, we note the current kinematic values of the DCS and subtract it from the kinematic values of the DCS when the clutch pedal is released to obtain an offset that represents how much the manipulators moved while the robot was disengaged. This offset is added to all previous samples in the DCS data up to the time when the clutch pedal was engaged, and the samples corresponding to the time during which the clutch pedal was pressed are filled in with the kinematic values of the DCS when the clutch pedal is released. These calculations are repeated for each period of clutching during the trial. Thus, the kinematic files for both the DCS and Raven II will now show no change in the variables for periods when the clutch pedal is pressed.

Then, we remove the effect of constant offsets by subtracting the average value from each variable. A scaling factor of 0.4 was used in operating the Raven II, so the values of the DCS kinematics were multiplied by 0.4. Figure A.4 shows the resulting kinematic traces for Trial 9 from the Raven II and DCS after the above transformations.

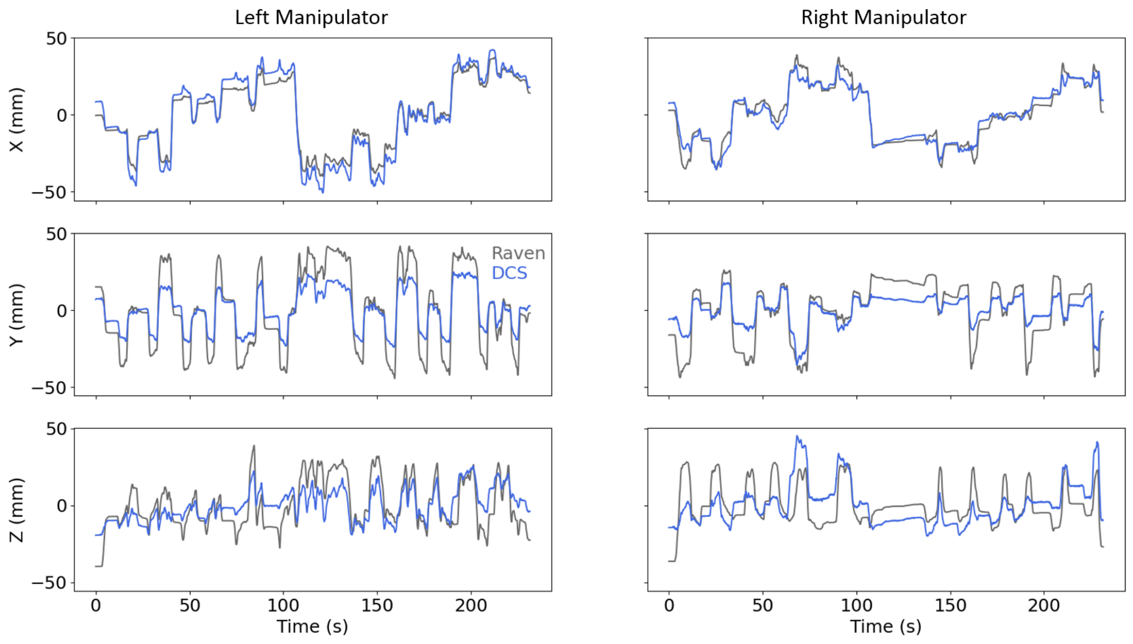


Figure A.4: Cartesian positions of the left and right end effectors on the Raven II compared to the positions of the left and right manipulators derived from the DCS for the ninth trial of Peg Transfer.

Evaluation Metrics

We quantified the performance of the DCS by calculating the root mean squared error (RMSE), mean absolute error (MAE), and mean absolute percentage error (MAPE) between the kinematics from the DCS and Raven II averaged across all nine trials. These are standard metrics for comparing predicted and true values from models and are implemented in the Scikit-learn library [180] for Python.

Root Mean Squared Error RMSE measures the Euclidean distance between predicted and measured values using Equation A.1 where n is the number of data points, y_i is the true value of the i-th sample given by the Raven II kinematics, and $\hat{y}_i$ is the predicted valued of the i-th sample derived from the DCS.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{A.1}$$

Mean Absolute Error MAE is the arithmetic average of the absolute errors between samples. Equation A.2 shows how it is calculated as the sum of the absolute value of the differences between the true and predicted values divided by the number of samples.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{A.2}$$

Mean Absolute Percentage Error MAPE measures the relative percentage error and is not sensitive to scaling like the previous metrics. It is calculated using Equation A.3 where ϵ is a small, strictly positive number to avoid undefined results if y_i is 0.

$$\text{MAPE} = \frac{1}{n}\sum_{i=1}^{n}\Big(\frac{|y_i - \hat{y}_i|}{\max(\epsilon, |y_i|)}\Big) \tag{A.3}$$

Table A.2 shows the performance of the DCS in representing the Raven II kinematics in terms of RMSE, MAE, and MAPE for each of the Cartesian positions for the left and right sides of the robot. We find that the RMSE is about 10-15 mm and the MAE is about 6-13 mm across all of the variables with MAPEs of about 2% except for L_z and R_z. This is because our method for processing and comparing the kinematic data assumed that the coordinate systems for the Raven II and DCS were aligned. However, there is likely to be some misalignment between the systems, meaning a linear movement along the positive x-axis for the Raven II may actually have additional components along the y- and z-axes of the DCS system. These misalignments should be accounted for by performing a calibration sequence to generate a more exact transformation matrix that maps the kinematic variables between the two systems. Such a transformation matrix would replace Table A.1.

Despite this, the kinematic data from the DCS is a good representation of kinematic data from a surgical robot due to the generally low MAPEs. As shown in Figure A.4, the DCS captures the overall movements of the surgeon's hands holding the manipulators.

Table A.2: Comparison between kinematic positions from the Raven II and DCS.

Metric	L_x	L_y	L_z	R_x	R_y	R_z
RMSE (mm)	14.29	12.37	15.36	7.20	10.88	15.87
MAE (mm)	11.83	9.86	12.74	6.08	8.56	13.06
MAPE (%)	2.61	1.82	7.23	1.96	1.68	10.34

A.4 Conclusion

In conclusion, we develop and evaluate a platform-independent system for collecting video and kinematic data from surgical robots. The DCS has been used to collect data of simulated tasks from a da Vinci Surgical System and the Raven II surgical robot. We evaluated the DCS by comparing the kinematic data collected from it to the kinematic data from the Raven II surgical robot and found that it had low MAPEs for most of the kinematic variables. This demonstrates that the DCS can be used to collect video and kinematic data from different surgical robotic systems.

Future work on the DCS includes further evaluation of the system using data collected from different tasks and surgical robots, and integrating additional sensors to collect multimodal datasets. In addition, the software architecture of the DCS could be restructured so that data logging and synchronization occurs at runtime, instead of in post-processing.

Appendix B

A Reactive Autonomous Camera System for the Raven II Surgical Robot

The endoscopic camera of a surgical robot provides surgeons with a magnified 3D view of the surgical field, but repositioning it increases mental workload and operation time. Poor camera placement contributes to safety-critical events when surgical tools move out of the view of the camera. This paper presents a proof of concept of an autonomous camera system for the Raven II surgical robot that aims to reduce surgeon workload and improve safety by providing an optimal view of the workspace showing all objects of interest. This system uses transfer learning to localize and classify objects of interest within the view of a stereoscopic camera. The positions and centroid of the objects are estimated and a set of control rules determines the movement of the camera towards a more desired view. Our perception module had an accuracy of 61.21% overall for identifying objects of interest and was able to localize both graspers and multiple blocks in the environment. Comparison of the commands proposed by our system with the desired commands from a survey of 13 participants

This appendix contains material from [181] coauthored with M. S. Yasar, H. Bhatia, and H. Alemzadeh, with help from E. Tian, A. Maran, and R. Wang in the collection and analysis of data, copyrighted by IEEE.

indicates that the autonomous camera system proposes appropriate movements for the tilt and pan of the camera.

B.1 Introduction

Surgical robots such as Intuitive Surgical's da Vinci Surgical Systems [3] are advancing medical specialties such as urology, gynecology, and general surgery by providing surgeons with increased flexibility and precision, while reducing incision size, recovery time, and scarring. Their adoption is linked to an increase in volume of minimally invasive surgery (MIS) cases [2], and is driving the development of new surgical procedures and technologies [1].

However, the current generation of robots is not autonomous yet. They are in level 0 of autonomy [182] and the surgeon must position and control all four arms manually which increases their mental workload [183]. One of these arms holds the endoscopic camera which provides surgeons with a magnified 3D view of the surgical field, but requires both hands and a foot to switch control of the arms and reposition the camera. During an operation, surgeons often adjust their camera position or settle for a suboptimal viewpoint which increases procedure time and risk to the patient since poor camera placement contributes to safety-critical events such as arm collisions, use of excessive force, dropping an object, or movement of the instruments out of camera's view [136, 184].

Existing methods for automating the surgical robot's camera focus on using *reactive* simple sets of rules, *proactive* machine learning algorithms to learn movement behavior, or *combined control* strategies that integrate these two techniques [183]. Most reactive autonomous camera systems rely on kinematic data from the surgical robot [185], use a camera to track the surgical tools [186], or track the surgeon's eyes using visual servoing [187], and change the camera position in direct relation to changes in these measurements. But, these methods overemphasize tracking tools and do not account for other objects of interest in the environment such as important tissues or needles. Our goal is to create an autonomous camera system that reduces surgeon workload and improves safety by providing an optimal viewpoint of the surgical environment that keeps *objects of interest and surgical tools* in the field of view thus reducing the likelihood of off-camera injuries.

In related work on autonomous camera systems, Yu et al. [188] introduced a region of interest (ROI) around the robot end effectors. Yang et al. [189] defined an intuitive virtual plane (IVP) as the plane normal to the surgeon's line of sight and containing the intersection of the surgeon's line of sight with the ROI. The IVP was a constraint to reduce misorientation that occurs when the optical and physical axes of the laparoscope are not parallel. However, their work used a 2D laparoscope and followed only one end effector.

While previous work focused on laparoscopic surgery, [190–192] were the first works that created an autonomous camera system for a surgical robot using the da Vinci Research Kit (dVRK) [193]. Their system used several rules and kinematic data from the dVRK to keep the tools centered in the camera's view. They conducted a trial with 20 participants, including four surgeons, that compared their automated camera system to the traditional clutched camera control. The results showed that the automated camera was able to keep the tools within the camera's field of view, and improved metrics of workload, efficiency, and progress. In contrast, our work relies on video data for detecting objects in the surgical workspace, thus enabling the consideration of *all* objects of interest in the environment as well as being platform-independent.

Contributions. We present a proof of concept of an autonomous camera system for the Raven II robot, an open-source platform for robotic surgery research [175].

- We introduce a custom-built camera arm, the Independent Binocular Imaging System (IBIS) (Section B.2.1), supporting a 3D stereoscopic camera (ZED Mini by Stereolabs Inc. [194]). The IBIS reports its joint and camera positions, and accepts commands from foot pedals and serial communication which facilitates system integration.

- We present a Perception module (Section B.2.2) for the automated perception of the surgical field using transfer learning to localize and classify objects of interest (end effectors and blocks in dry lab) in a given image of the surgical field. We use a Mask Region-based CNN (MRCNN) [195] (with convolutional layers pre-trained using the COCO dataset [196]) to identify objects of interest by generating bounding boxes, classifying, and then creating masks. The coordinates of all objects of interest provided by the Perception module are then used to calculate the centroid of the objects and to adjust the camera position.

- We present a Control module (Section B.2.3) that adjusts the zoom, pan, and tilt of the camera to align the center of the view with the centroid of the objects and maximize the view of all objects of interest. The Control module computes the projection of each object's position onto an image plane containing the centroid. Then, control rules determine camera movements based on the position of the centroid relative to the desired view area of the camera.

The training set for the MRCNN consisted of 1,686 annotated images (3,372 after augmentation), and the validation set consisted of 600 images. The images were collected from dry lab experiments of the Pick and Place task. The dataset included both left and right images from the ZED Mini camera, but there was not a 50/50 ratio between the two. After tuning the hyperparameters using the early stopping technique, the overall loss (defined as the sum of the region proposal class loss, region proposal bounding box loss, MRCNN class loss, MRCNN bounding box loss, and MRCNN mask loss) was 0.2672.

We evaluated the camera system using a separate set of 27 pairs of left and right images of a Block Transfer task, and a survey was used to determine the ground truth desired commands for each image. The Perception module correctly identified 61.21% of the objects, and the Control module demonstrated acceptable behavior when tilting and panning, but the desired zoom area should be decreased to provide a wider field of view.

B.2 Autonomous Camera System

Our autonomous camera system consists of a custom-built camera arm integrated with a ZED Mini and a software pipeline for perception of the environment and control of the camera arm, as shown in Figure B.1 and described next.

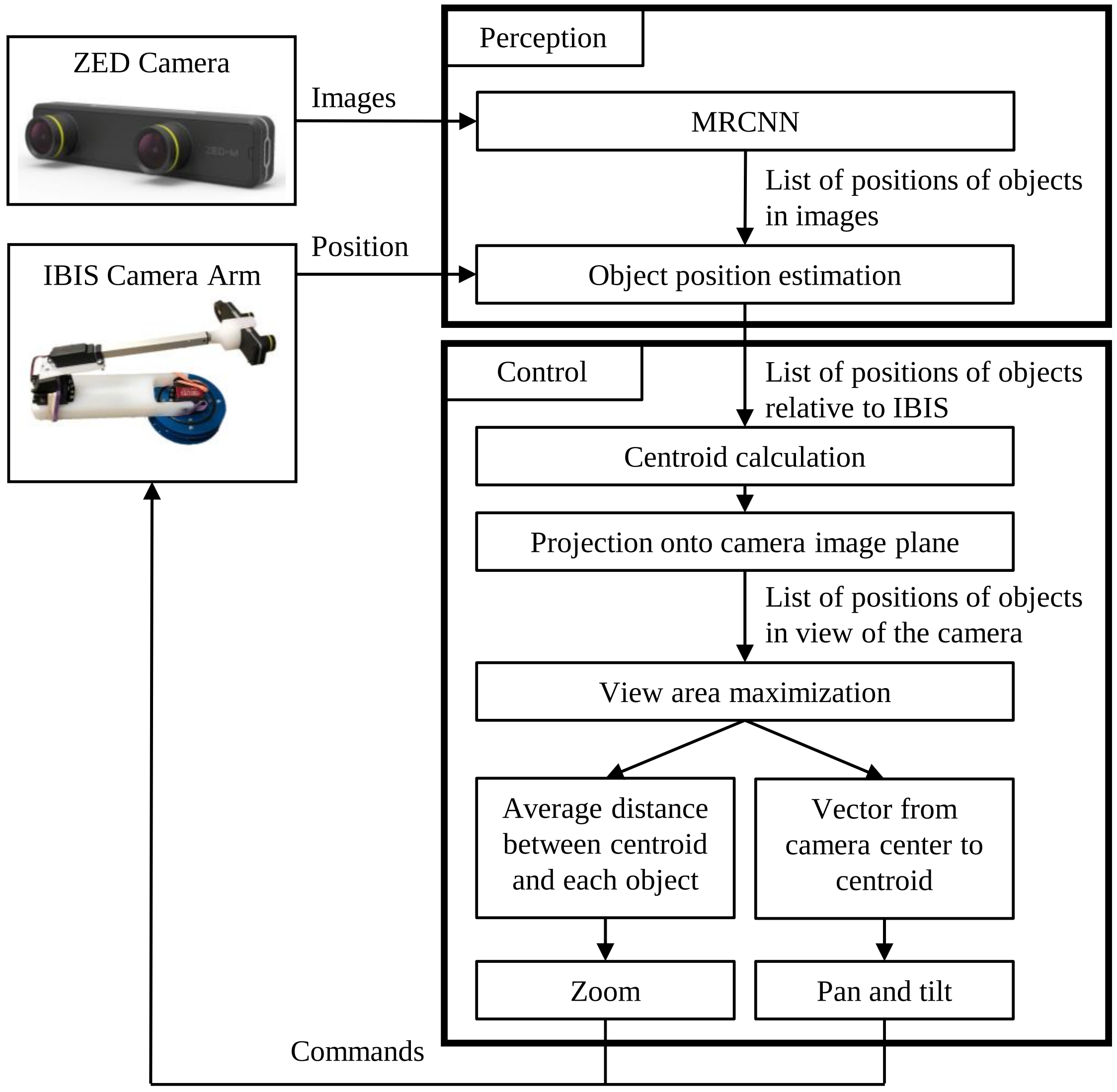

ZED Camera
Images
IBIS Camera Arm
Position
Perception
MRCNN
List of positions of objects in images
Object position estimation
Control
List of positions of objects relative to IBIS
Centroid calculation
Projection onto camera image plane
List of positions of objects in view of the camera
View area maximization
Average distance between centroid and each object
Vector from camera center to centroid
Zoom
Pan and tilt
Commands

Figure B.1: Autonomous camera pipeline.

B.2.1 IBIS Camera Arm

A custom robotic camera arm, shown in Figure B.2, was developed to hold the stereoscopic camera and provide control over the position of the camera during teleoperation of the Raven II. This Independent Binocular Imaging System (IBIS) can also be controlled using a set of foot pedals making it platform-independent and enabling future experimentation with alternative control interfaces for surgical endoscopes. An Arduino Uno R3 runs custom forward kinematic and inverse kinematic models, and communicates through a serial port to report its position and accept commands. This allows for open-source development and facilitates integration with other systems.

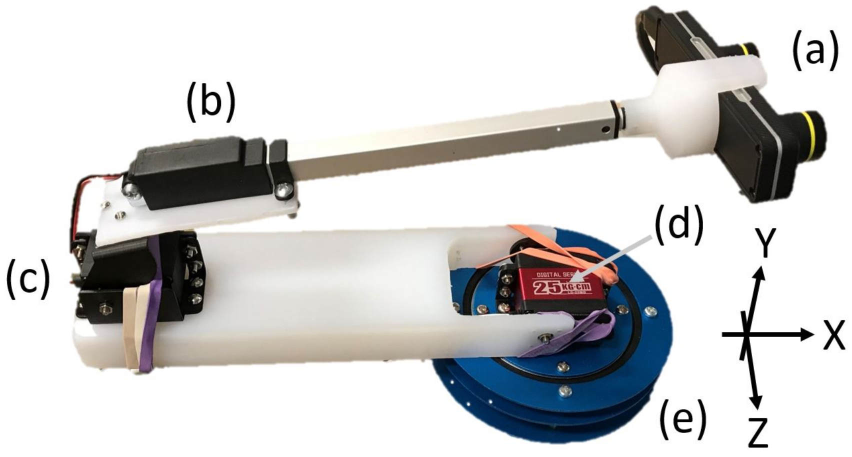


Figure B.2: IBIS Camera Arm: (a) ZED Mini, (b) Linear Actuator, (c) Top Servo, (d) Bottom Servo, and (e) Base Servo.

The mechanical design of the IBIS consists of upper and lower arms positioned using three servos and a linear actuator. The Base Servo, embedded in the blue base of the IBIS shown in Figure B.2, controls the pan of the camera by rotating the entire arm left or right. The Bottom Servo is a high torque servo that supports the rest of the arm and camera. The Top Servo, at the joint between the upper and lower arms, is a standard servo with an angled bracket that joins to the mounting plate for the linear actuator. The ZED Mini is held by a custom-machined adapter that fits on the end of the linear actuator.

The Base Servo pans the camera arm, while the other two servos and linear actuator move together to change the vertical and horizontal position of the camera. These movements are calculated using the inverse kinematics functions based on incremental changes in the position of the Bottom Servo. This enables the calculation of a deterministic solution for the positions of the Top Servo and linear actuator. The Base Servo is the origin of the coordinate system of the IBIS where the positive x-axis points into the operating space, the positive y-axis points upwards, and the positive z-axis points right.

The software on the Arduino Uno uses an open source library [197] to control the servos and the linear actuator. The main function of the code is to listen for commands from the foot pedals or serial line, use inverse kinematic functions to move the arm, update the state of the arm using the kinematic functions, and report the position of the arm over a serial line at a baud rate of 9600. A Python script on the receiving computer parses and logs the position of the IBIS enabling real-time and closed loop control of the arm.

B.2.2 Perception

Object Detection and Localization

We used Transfer Learning to apply the knowledge obtained from a pre-trained model to our task. This allowed us to obtain competitive accuracy with a smaller training dataset. As the perception task involves both object detection and localization, we used the Mask Region-based Convolutional Neural Network (MRCNN) architecture, which has a backbone network, in our case pre-trained Residual Nets (ResNet) [96], followed by the network head. The backbone network performs feature extraction on a given image, and is followed by the Region Proposal Network (RPN) which examines each region of interest (ROI), before feeding the extracted features into the fine-tuned classification layers. The classification layers generate the bounding boxes and masks for each class. The bounding boxes indicate where an object has a high probability of being found while the masks reveal where the object was actually found.

In our task, the classes of the objects were “Left Grasper”, “Right Grasper”, “Red Block”, “Green Block”, and “Background”. We fine-tuned the final two layers of the MRCNN, pre-trained on the COCO dataset [196], for our application of detecting and localizing objects of interest in the surgical workspace. The classification layers were trained on a set of 1,686 images (3,372 after augmentation) that were manually annotated using the VGG Image Annotator (VIA) [198]. The images were annotated by five students and each image was annotated once. Each object’s boundary was outlined and it was labeled with the appropriate class. These images were collected using the ZED Mini from the execution of the Fundamentals of Laparoscopy (FLS) Pick and Place task on the Raven II robot. The diversity in scenes was ensured to a certain degree by picking up different blocks with the graspers and showing incremental movements in the images.

We performed image augmentation to diversify object orientations in the images of the training set and improve the model’s ability to detect objects of interest. After image augmentation, the training set consisted of 3,372 images. This is one method of artificially expanding a dataset. Some of the techniques that can be used for image augmentation include scaling, translation, rotation, flipping, adding noise, and changing lighting conditions. The techniques used for augmenting our dataset were flipping the images left/right 50% of the time and generating images with random blending between the original images and their canny edges.

Object Position Estimation

For each left and right image obtained from the ZED Mini, the MRCNN returns a list of the objects and the coordinates for the upper left and lower right corners of their bounding boxes. The centers of the bounding boxes are assumed to be the center of the object in the image, and are used for subsequent calculations. These objects are sorted and paired, and if an object was detected in one image but not the other, it is discarded because there is not enough information to reconstruct that object’s position.

The locations of the objects in the left and right images are used to estimate each object's position with respect to the camera arm. Then, the centroid of the identified objects is calculated so that an image plane containing this point and perpendicular to the optical axis of the camera can be constructed. Each object is then projected onto this image plane in order to relate their locations to a desired field of view of the camera.

For each object in the list returned by the MRCNN, the center of the bounding box is calculated by taking the average of the x and y pixel coordinates of the corners. This estimates the center of each object in the left and right images. The lists are sorted to pair an object's location in the left image with its corresponding location in the right image.

The position of the camera and its optical axis are obtained from the IBIS. A unit vector parallel to the camera's optical axis is defined as the normal unit vector, $\hat{\mathbf{n}}$. The camera is assumed to be horizontal which allows the construction of a horizontal unit vector, $\hat{\mathbf{h}}$, perpendicular to the optical axis of the camera. A third, orthogonal unit vector is defined as the cross product of the normal unit vector and the horizontal unit vector. This vector is named the vertical unit vector, $\hat{\mathbf{v}}$.

The distance, d, of an object from the camera is calculated using the difference in the horizontal position of the object between the left and right images as shown in Equation B.1. The ZED Mini has a focal distance of $f = 700$ pixels and a camera separation of $S = 63$ mm. L_{px} and R_{px} are the horizontal positions, in pixels, of the object in the left and right images, respectively.

$$d = \frac{fS}{|L_{\text{px}} - R_{\text{px}}|} \tag{B.1}$$

Then, the horizontal displacement of the object, d_{h}, in the direction of $\hat{\mathbf{h}}$, from the center of the camera's view is calculated using Equation B.2. This is added to $\frac{1}{2}S$ to account for the position of the right camera offset from the center of the ZED Mini. The images used in our experiments are 720x1280, so $R_{\text{px}} - 640$ represents the horizontal location of the object relative to the center of the right image.

$$d_{\mathrm{h}} = S\Big(\frac{1}{2} + \frac{(R_{\mathrm{px}} - 640)}{|L_{\mathrm{px}} - R_{\mathrm{px}}|}\Big) \tag{B.2}$$

Likewise, the vertical position, R_{py}, of the object in the right image is used to calculate the vertical displacement of the object, d_{v}, in the direction of $\hat{\mathbf{v}}$, from the center of the camera's view, as shown in Equation B.3. Similarly, $R_{\mathrm{py}} - 360$ represents the vertical location of the object relative to the center of the right image.

$$d_{\mathrm{v}} = S\Big(\frac{360 - R_{\mathrm{py}}}{|L_{\mathrm{px}} - R_{\mathrm{px}}|}\Big) \tag{B.3}$$

The object's position relative to the camera arm is then calculated as the position of the camera added to the distances and displacements multiplied by their respective unit vectors, as shown in Equation B.4.

$$P_{\mathrm{object}} = P_{\mathrm{camera}} + d\hat{\mathbf{n}} + d_{\mathrm{h}}\hat{\mathbf{h}} + d_{\mathrm{v}}\hat{\mathbf{v}} \tag{B.4}$$

B.2.3 Control

The Control module uses the list of the objects' positions relative to the camera arm to calculate the centroid of the objects as their average position. An image plane containing the centroid and defined by $\hat{\mathbf{v}}$ and $\hat{\mathbf{h}}$ is constructed. The distance from the camera to the image plane, d_{cam}, is calculated using Equation B.5 where $\vec{p}_{\mathrm{centroid}}$ and $\vec{p}_{\mathrm{camera}}$ represent vectors from the origin to the centroid and camera's position, respectively. The camera's position on the image plane defines the origin of the image plane.

$$d_{\mathrm{cam}} = (\hat{\mathbf{n}} \cdot \vec{p}_{\mathrm{centroid}}) - (\hat{\mathbf{n}} \cdot \vec{p}_{\mathrm{camera}}) \tag{B.5}$$

Then, each object is projected onto the image plane and the average distance from the centroid to each object's projected position is calculated. The height of the image plane within view of the camera is calculated using Equation B.6, proportional to the distance from the image plane to the camera. The radius of the desired view was set at 50% and defined a circle that should maximally overlap the circle drawn around the centroid by the average distance to the objects.

$$h_{\text{visable}} = d_{\text{cam}} \tan(30^\circ) \tag{B.6}$$

Based on the location of the centroid with respect to the origin of the image plane (the center of the camera's view), the control rules determine the movement of the camera. Zoom is controlled by the difference in size of the circles around the centroid and origin. If the average distance from the centroid to the objects is larger than the desired view ring, then the system proposes zooming out. Conversely, if the average distance from the centroid to the objects is smaller than the desired view ring, then the system proposes zooming in. The tilt and pan of the camera are controlled by the location of the centroid on the image plane relative to the origin of the image plane. If the centroid is further away from the origin than 30% of the visible height of the camera's view in any direction, then the camera is tilted up or down, or panned left or right to bring the centroid closer to the center of the camera's view.

B.3 Experimental Evaluation

The autonomous camera system was evaluated by examining the Perception and Control modules separately as well as assessing the system as a whole. This analysis considered 27 pairs of left and right images of a block transfer task showing the left and right graspers and several blocks. These 54 images were annotated to create the ground truth of the locations of the objects in the images. The accuracy of the MRCNN was defined as the ratio of correctly localized and classified objects to the total number of objects present in the image set. An object was considered correctly localized if the center of the object was within the bounding box generated by the MRCNN. The root mean square error (RMSE) for the horizontal and vertical locations of the centers of the objects were also calculated. To evaluate the Control module, 13 students were shown these 27 pairs of images and asked how they would adjust the camera's zoom, pan, and tilt to achieve a better view of the objects in the environment. For each pair of images, the commands given by the Control module using the object locations directly from the Perception module and the commands given by the Control module using the ground truth object locations were compared to the majority vote of the commands from the survey.

All the experiments were conducted on an x86_64 PC with an Intel Core i7 CPU @ 3.70 GHz and 32 GB RAM, running Linux Ubuntu 18.04 LTS, and an Nvidia 1080 Ti GPU, running CUDA 10.1. We used Keras [199] API v.2.2.4 on top of TensorFlow [200] v.1.13.1 for training our model and Scikit-learn [108] v.0.21.3 for preprocessing and evaluation.

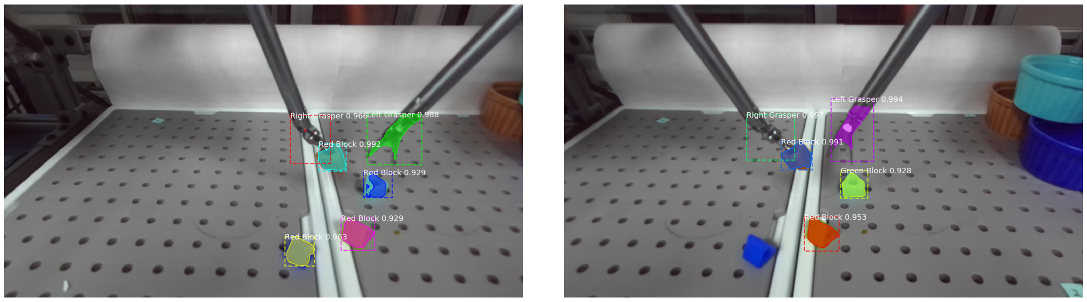
B.3.1 Perception

The model was evaluated in terms of its overall loss, bounding box loss, classification loss, validation bounding box loss, and validation classification loss which are listed in Table B.1. The validation bounding box loss was 0.1346, and the validation class loss was 0.0274. We used hyperparameter tuning to find the optimum learning rate which was 0.01 for 20 epochs, each with a step size of 50 and a batch size of 1 image. Tensorboard was used to visualize the impact that certain hyperparameters had on the model's performance.

Table B.1: MRCNN losses.

Type	Loss
Bounding Box Loss	0.0422
Class Loss	0.0312
Validation Bounding Box Loss	0.1346
Validation Class Loss	0.0274
Overall Loss	0.2672

The MRCNN model predicted on images from the ZED Mini and returned a list of objects identified for each image. One such pair of left and right images is shown in Figure B.3. Although both graspers and four blocks were localized in the left image, only the two graspers and three blocks were detected in the right image. The rightmost block in the right image was classified as a Green Block and was ignored by the algorithm. In addition, the vertical difference in the positions of the right grasper in the left and right images was too large, so it was not correctly paired, and was thus also ignored. This left us with only three objects to use in the proceeding calculations, the left grasper and two blocks.



(a) Left image from the ZED Mini (b) Right image from the ZED Mini

Figure B.3: Pair of images with objects localized and classified by the MRCNN.

The MRCNN consistently misclassified the left grasper as the Right Grasper and the right grasper as the Left Grasper. Errors in the classification of the graspers can be attributed to insignificant differences between the color of the graspers and the background. The graspers appear to blend in with their background and although apparent to the naked eye, the two can be easily confused with a camera. The right and left graspers were also very similar in shape and size, so without information about their relative position in regards to other objects, they would be difficult to differentiate.

Different colored blocks were also used to test the ability of the MRCNN to discern color, but since most of the blocks were classified as Red Blocks regardless of their color, this analysis considered Blocks in general. The model had an accuracy of 52.87% in correctly identifying blocks in the images. Due to the consistent mislabeling of the graspers, this analysis considered Graspers in general as well. The model had an accuracy of 80.77% in correctly identifying graspers in the images. Table B.2 shows the number of objects correctly classified out of the total number of objects for all 54 images. The MRCNN correctly localized but misclassified seven objects and in two cases incorrectly identified the frame of the Raven II as a Right Grasper. Overall, the model had an accuracy of 61.21% in correctly classifying blocks and graspers in the images and provided sufficient information for the Control module to estimate the centroid of the objects and propose commands to move the camera.

Table B.2: Object classification accuracies.

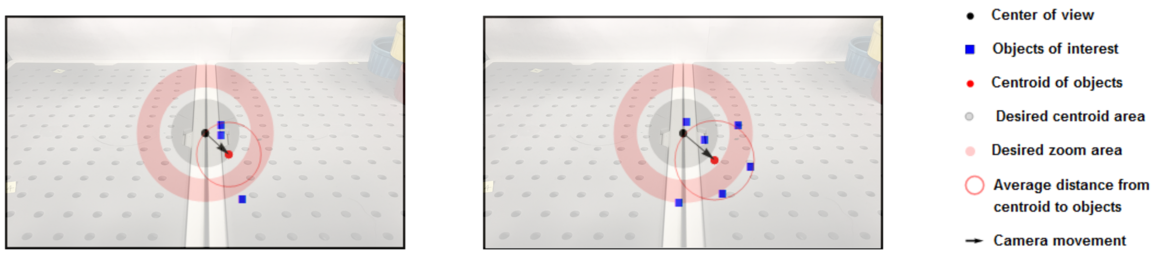
Class	Correctly Classified	Total	Accuracy (%)
Graspers	84	104	80.77
Blocks	129	244	52.87
All objects	213	348	61.21

The ground truth annotations were also used to determine the RMSE in the horizontal and vertical locations of the centers of the objects in each image. The RMSE error in the horizontal and vertical locations of the bounding boxes proposed by the MRCNN and the centers of the ground truth annotations were 20.37 pixels and 16.98 pixels, respectively. The sorting and pairing algorithm used by the system to match the position of an object in the left image with its position in the right image tolerated up to a 20 pixel difference in vertical position, so this was an acceptable amount of error.

B.3.2 Control

Figure B.4 shows the objects in Figure B.3 mapped to the image plane. Given the estimated positions of the objects identified by the Perception module, the Control module mapped them to the image plane as shown in Figure B.4a. The Control module sent commands to the IBIS to zoom in. However, given the list of estimated positions for each object calculated using the ground truth annotations, the Control module mapped these objects to the image plane as shown in Figure B.4b and proposed zooming in, tilting down, and panning right instead.

The Control module determined commands to move the camera based on the location of the centroid relative to the center of the camera's view, and the average distance between the centroid and each object compared to a desired zoom radius.

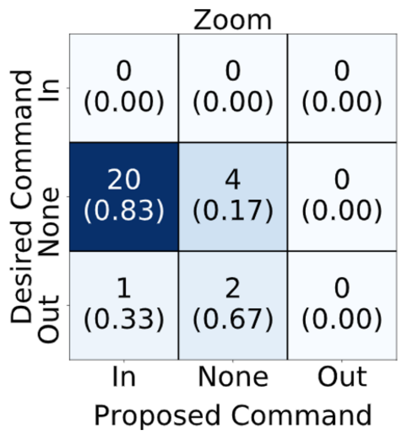


(a) (b)

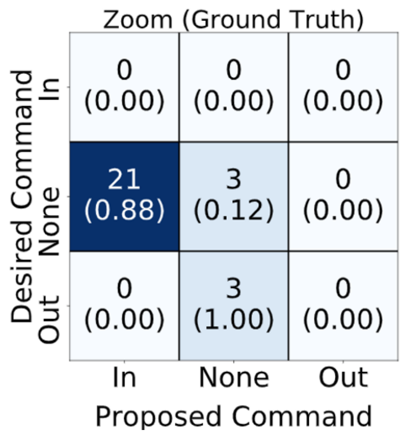
Figure B.4: Projection of objects located in Figure B.3 onto the camera image plane. (a) Object coordinates directly from MRCNN where mispairing resulted in incorrect position estimations, (b) Object coordinates from ground truth.

In Figure B.4, the center of the camera's view is a black dot while the centroid of the objects is a red dot. The gray disk represents the desired area that the centroid should be in and the Control module adjusted the pan and tilt of the camera to move the gray disk towards the centroid. The average distance between the centroid and objects is shown with a red circle around the centroid and represents the size of the region of interest. The thick pink ring around the center of the image represents the desired zoom area and the Control module adjusted zoom so that the red circle was within the pink ring.

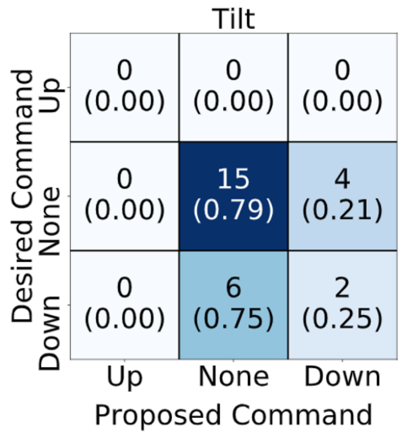
In order to evaluate the control rules used to determine camera movement commands, 13 graduate and undergraduate students were shown the 27 pairs of images and asked to select commands for zoom, tilt, and pan (no movement was also an option for each category). The ground truth for desired movements for each image was determined by majority voting based on the survey responses. The confusion matrices between the desired commands and the Control module's commands are shown in Figure B.5. Figures B.5a, B.5c, and B.5e show the confusion matrices for the commands proposed by the system when directly fed the predictions of the MRCNN, and Figures B.5b, B.5d, and B.5f show the confusion matrices for the commands proposed by the system based on the ground truth annotations for the Perception module (assuming perfect Perception).



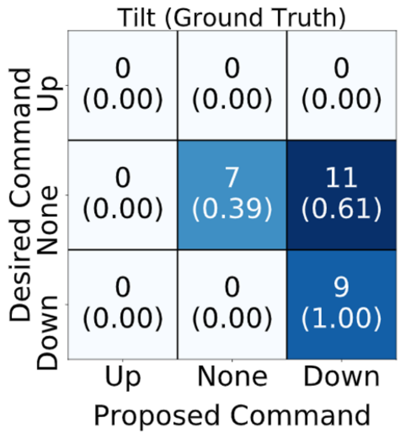
(a)



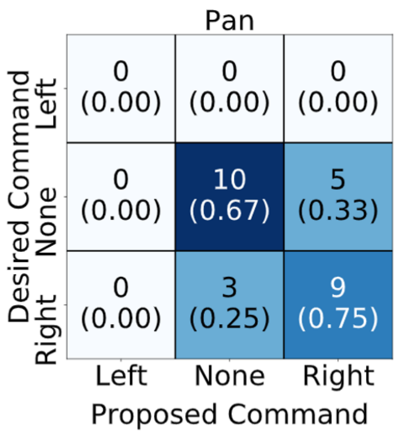
(b)



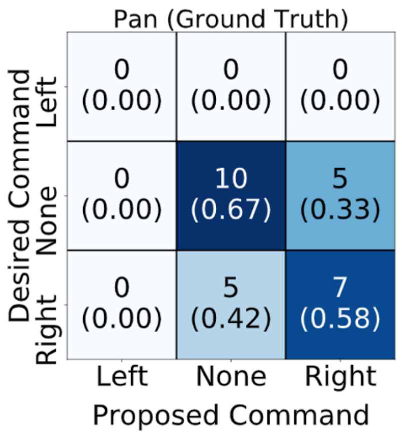
(c)



(d)



(e)



(f)

Figure B.5: Confusion matrices for Control module commands. (a), (c), and (e) are based on MRCNN predictions. (b), (d), and (f) are based on ground truth annotations for Perception.

The set of images used in this analysis included frames showing a Block Transfer task, during which the camera position was held constant. The task occurred in the central, lower, and slightly right regions of the operating space, which meant appropriate camera commands were limited to tilting down, panning right, adjusting zoom, or no movement. Thus, the confusion matrices in Figure B.5 only show data for these movements. The confusion matrices for the zoom command show that the system proposed zooming in when the desired command was no movement. This behavior was consistent even when the Control module was given the ground truth annotations. Since the amount of zoom tends to be subjective, the desired zoom area should be adjustable to accommodate personal preferences. The confusion matrices for the tilt command show that in 75% of cases the system proposed no adjustments to tilt even if the desired command was to tilt down. However, when given the ground truth annotations, the system often proposed tilting down when the desired command was no movement (61%). On the other hand, the system usually selected appropriate commands for panning, even given the ground truth annotations.

The undesired tilt down commands could be explained by the equal weighting of all objects in the centroid calculation. The Perception module had a higher accuracy in detecting graspers, which were usually located more centrally in the images and would have led the system to propose no change in tilt. But using the ground truth, the blocks would have pulled the location of the centroid down in the image resulting in tilt down commands. The difference between Figures B.5c and B.5d suggests that a weight function should be implemented in the centroid calculation to address how some objects are more important to view than others.

B.4 Conclusion

This work presents a proof of concept of an autonomous camera system for teleoperated robotic surgery that tracks the centroid of all objects of interest in the field of view of the camera. The objects of interest were identified using transfer learning, with an MRCNN partly pre-trained on the COCO dataset. A custom-built camera arm was created for positioning the camera and the centroid of objects of interest was tracked using a set of control rules. The system was evaluated using a dataset of images from dry lab experiments and by comparing the proposed motions of the camera to the desired motions. The evaluation results suggest that the system proposes appropriate movements for the tilt and pan of the camera, but the desired zoom area should be decreased to provide a wider field of view and an object weight function should be implemented in the centroid calculation. Future work will focus on improving the accuracy of the Perception module by increasing the size and diversity of the training set and generating more accurate annotations, adjusting the control rules for zoom, and testing the system on a wider variety of object configurations and movement directions.

Code Availability

Designs and code for the IBIS and the perception model are respectively available at `https://github.com/kch4fk/IBIS` and `https://github.com/thatssoharsh/Raven-II-Perception`.

Appendix C

Motion Primitive State Graphs

The following figures show the state graphs for Section 4.3 for the gestures in each task performed by experts, intermediates, and novices where the surgeon-defined motion primitive sequences are highlighted in green and inverse motion primitives are boxed in red.

C.1 Suturing

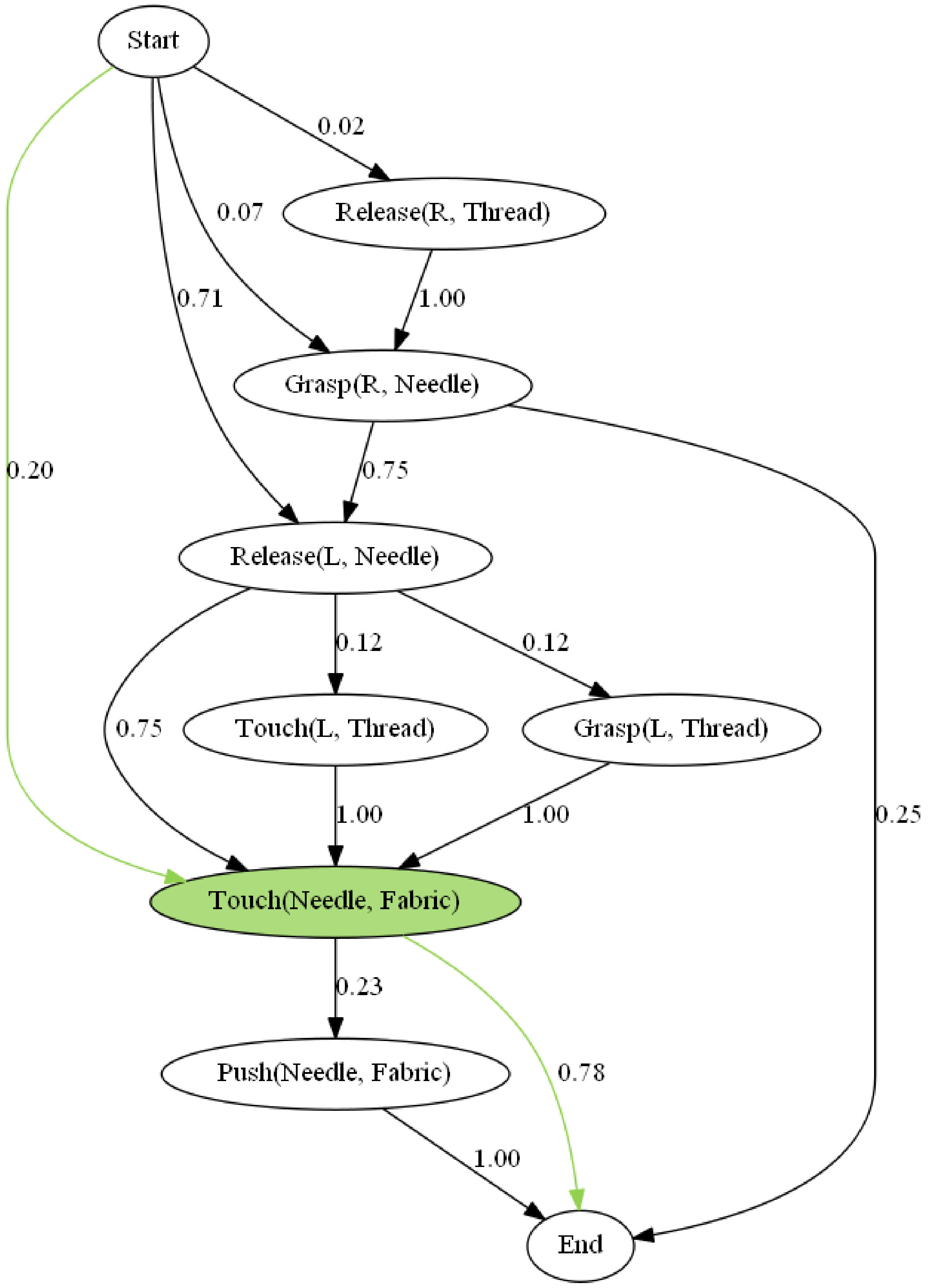


Figure C.1: State graph for Suturing G2 performed by experts.

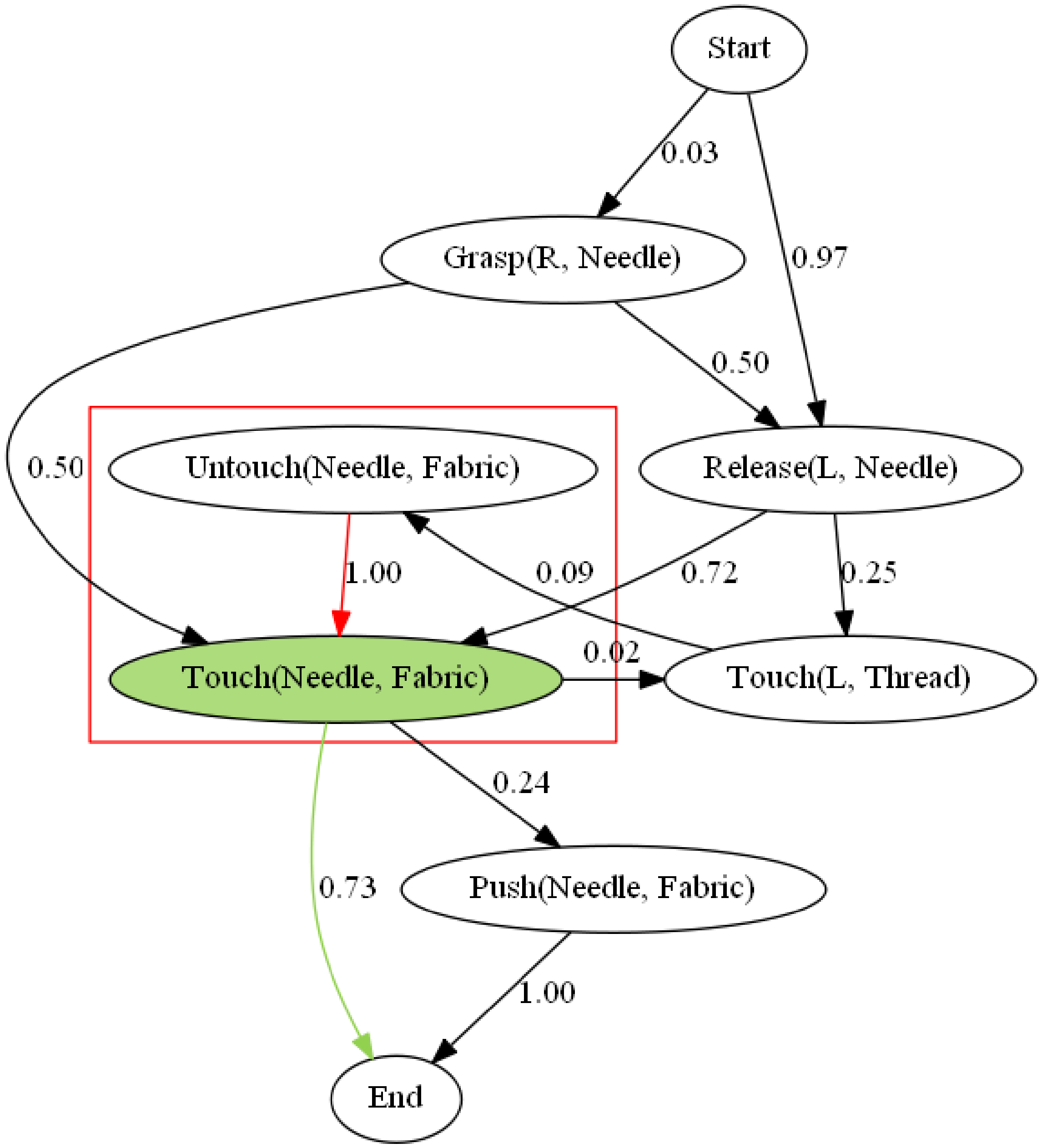


Figure C.2: State graph for Suturing G2 performed by intermediates.

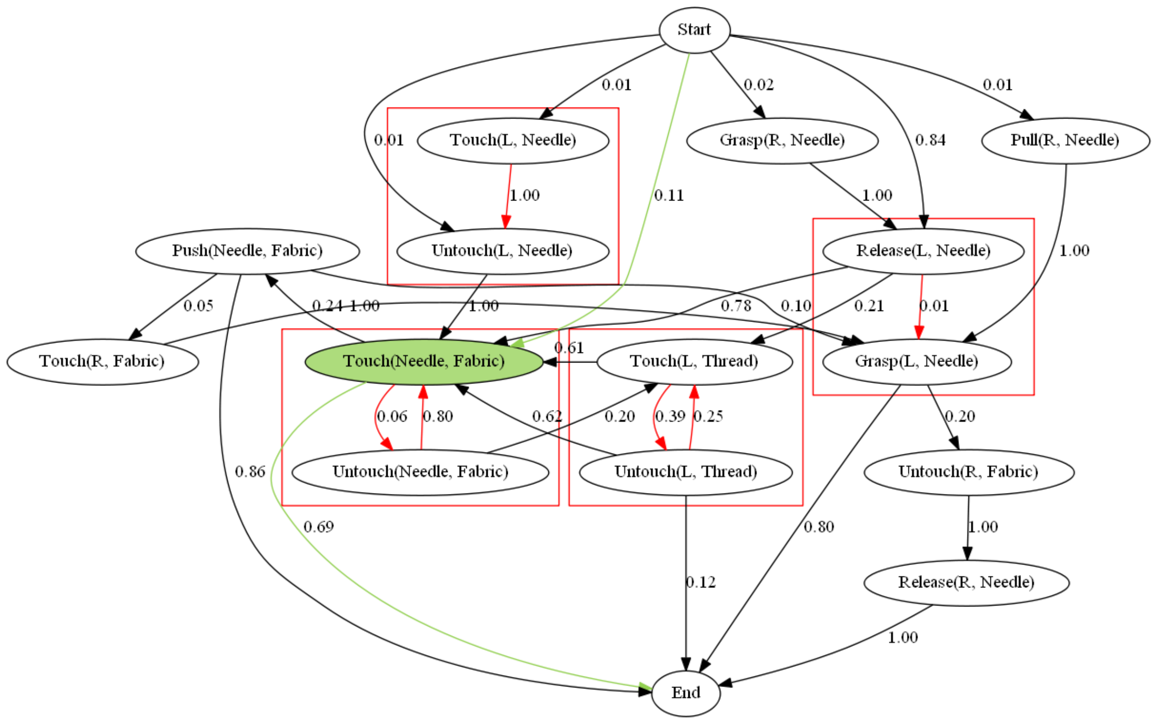


Figure C.3: State graph for Suturing G2 performed by novices.

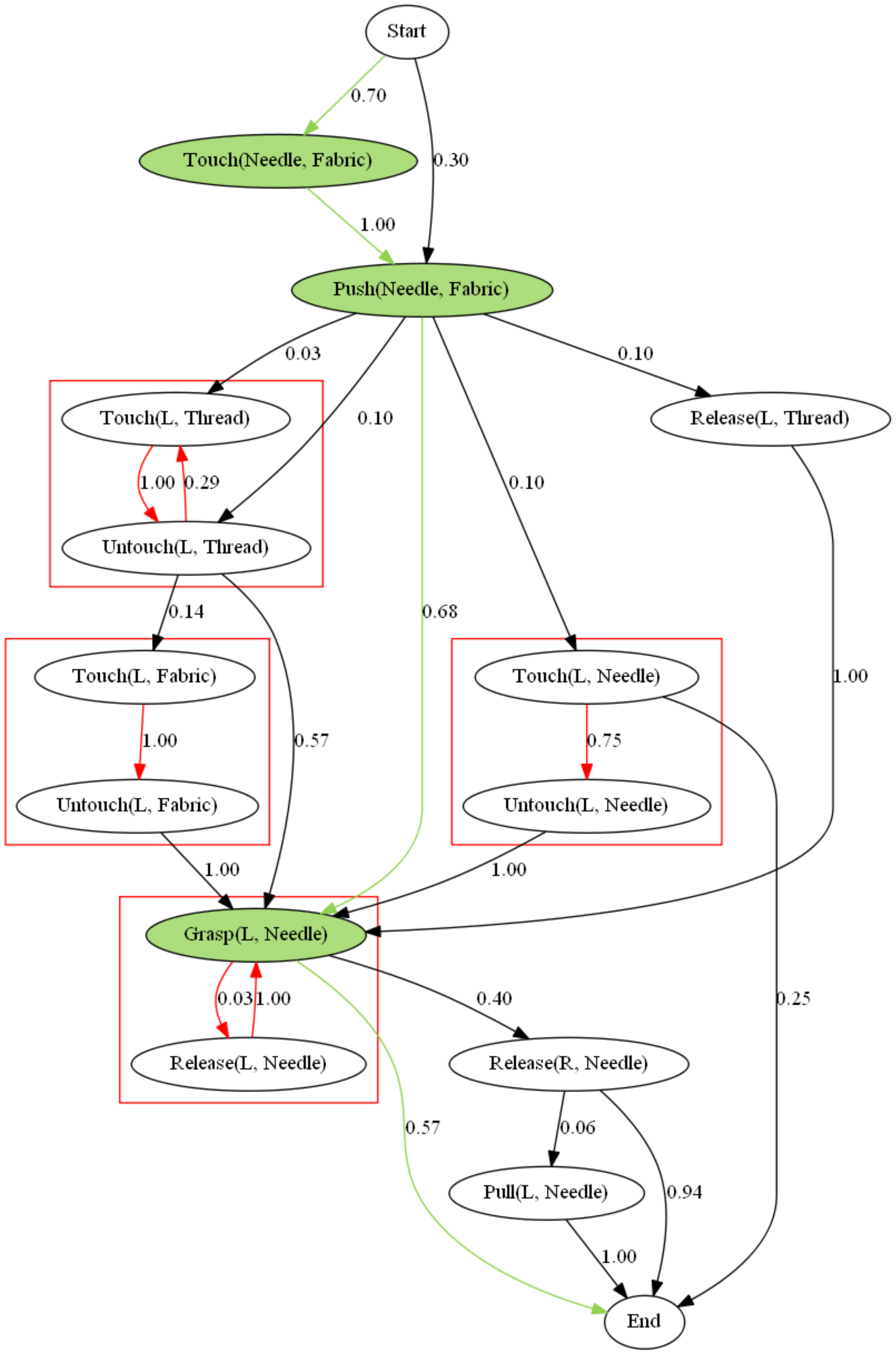


Figure C.4: State graph for Suturing G3 performed by experts.

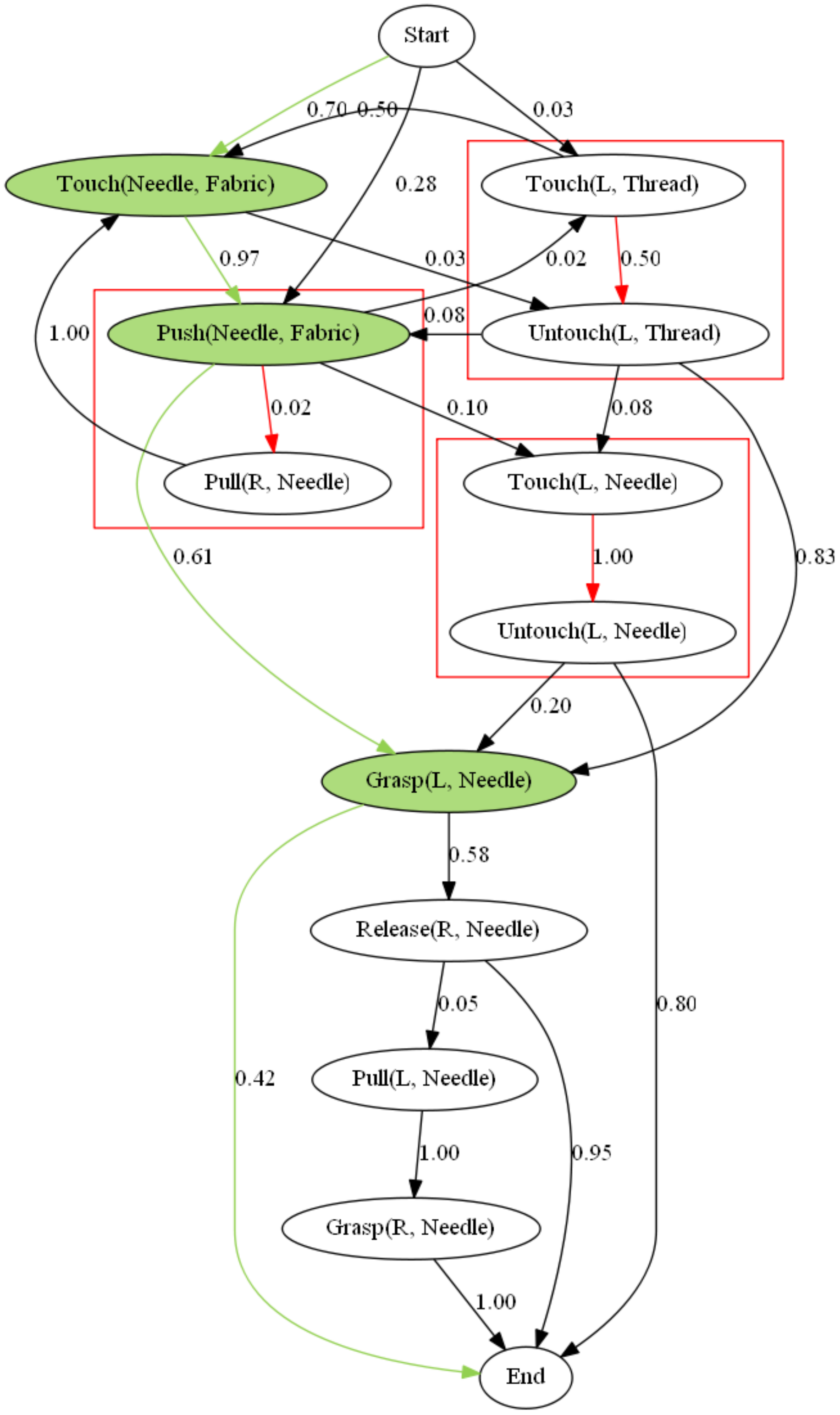


Figure C.5: State graph for Suturing G3 performed by intermediates.

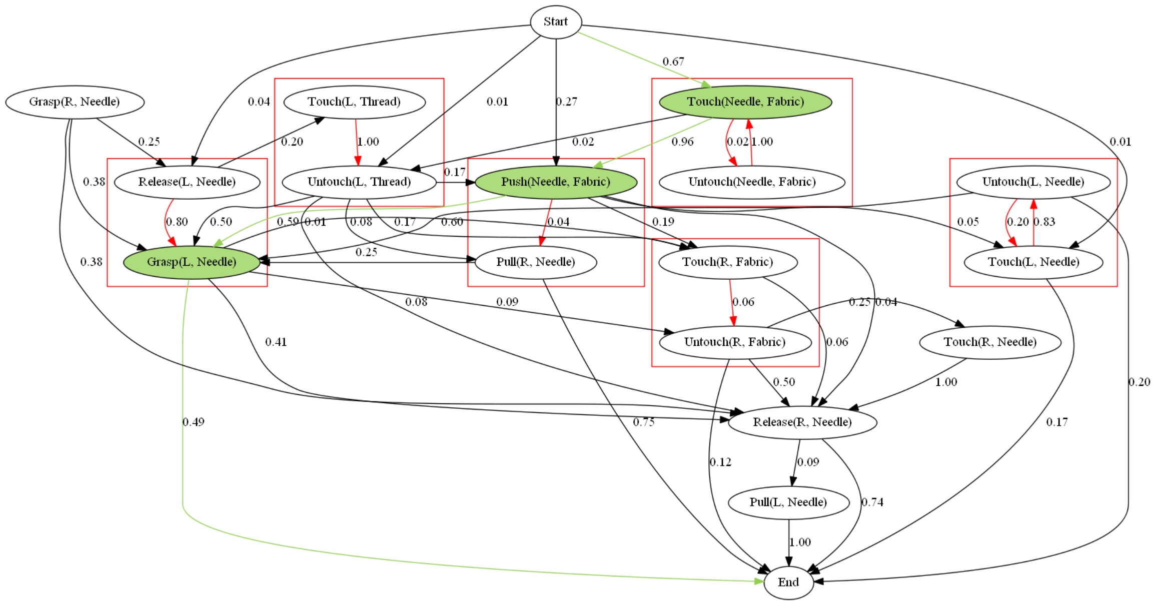


Figure C.6: State graph for Suturing G3 performed by novices.

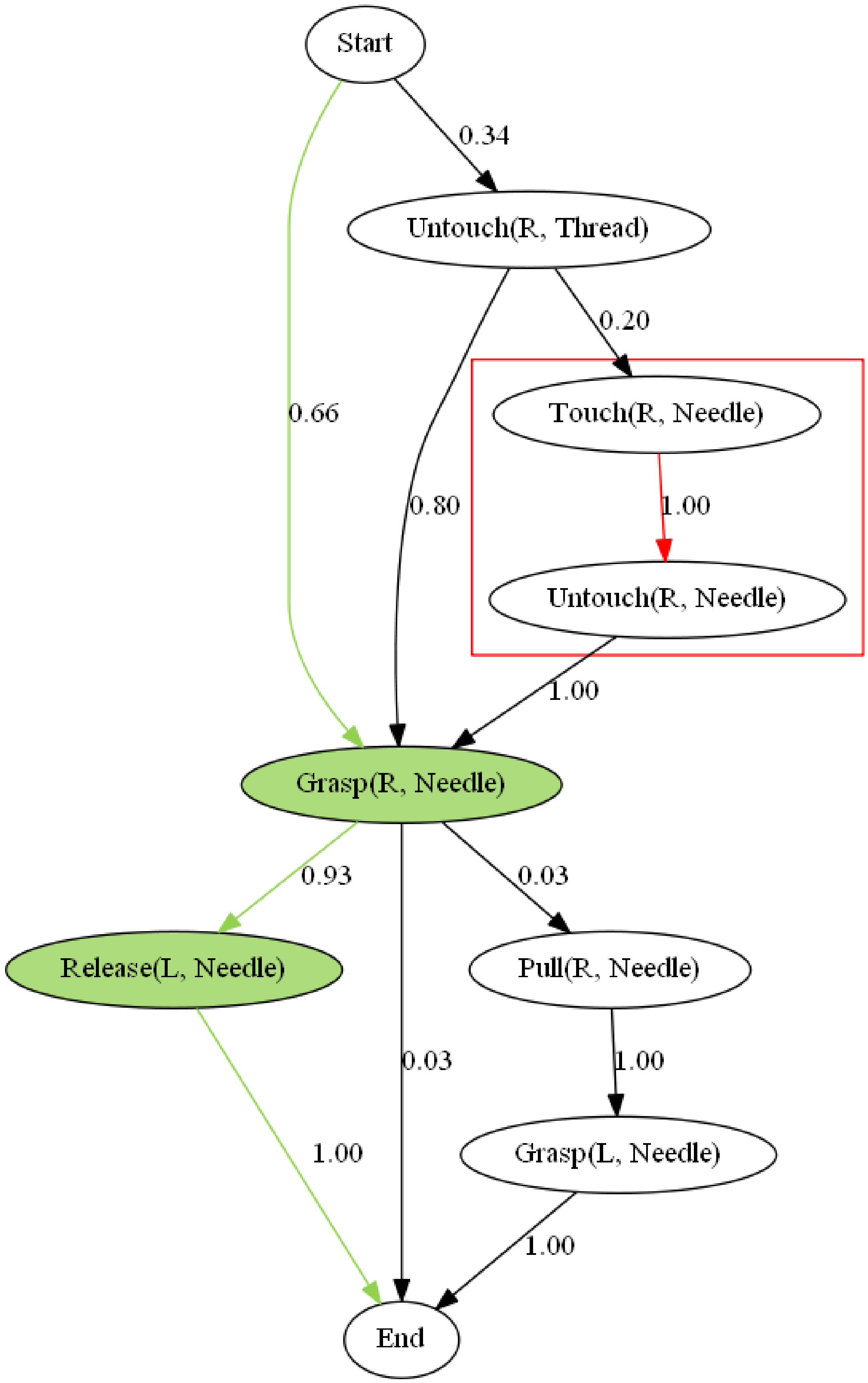


Figure C.7: State graph for Suturing G4 performed by experts.

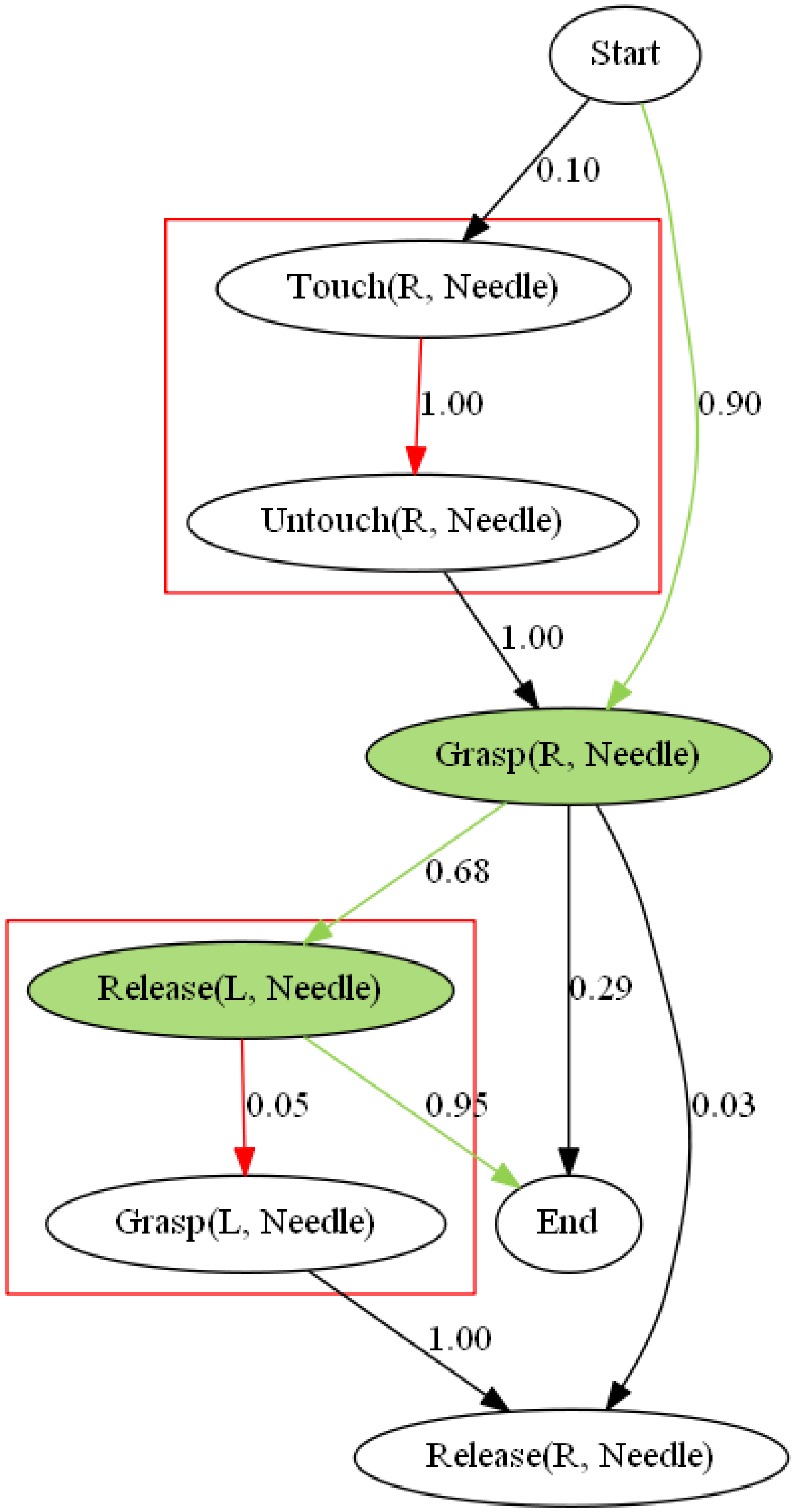


Figure C.8: State graph for Suturing G4 performed by intermediates.

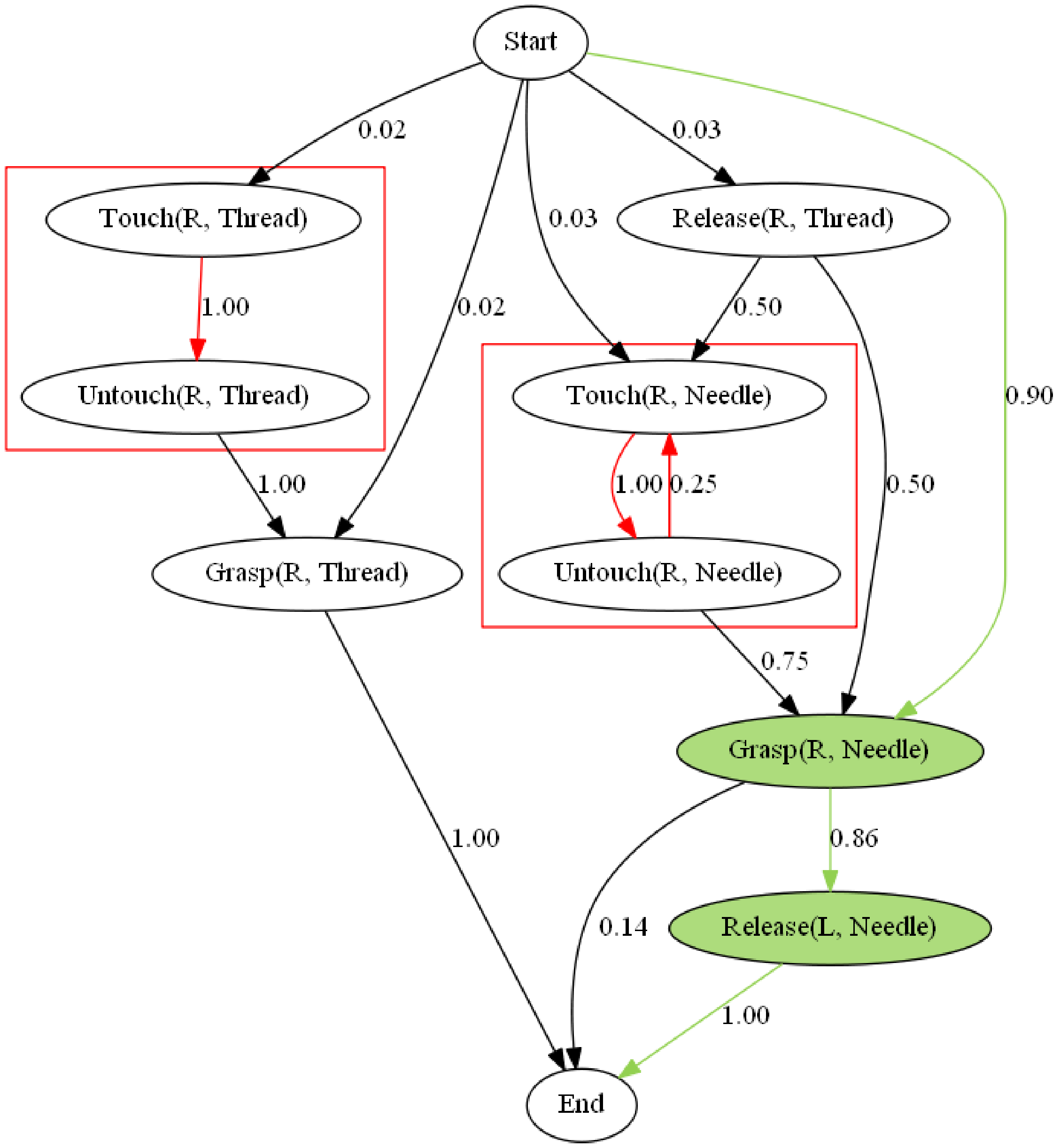


Figure C.9: State graph for Suturing G4 performed by novices.

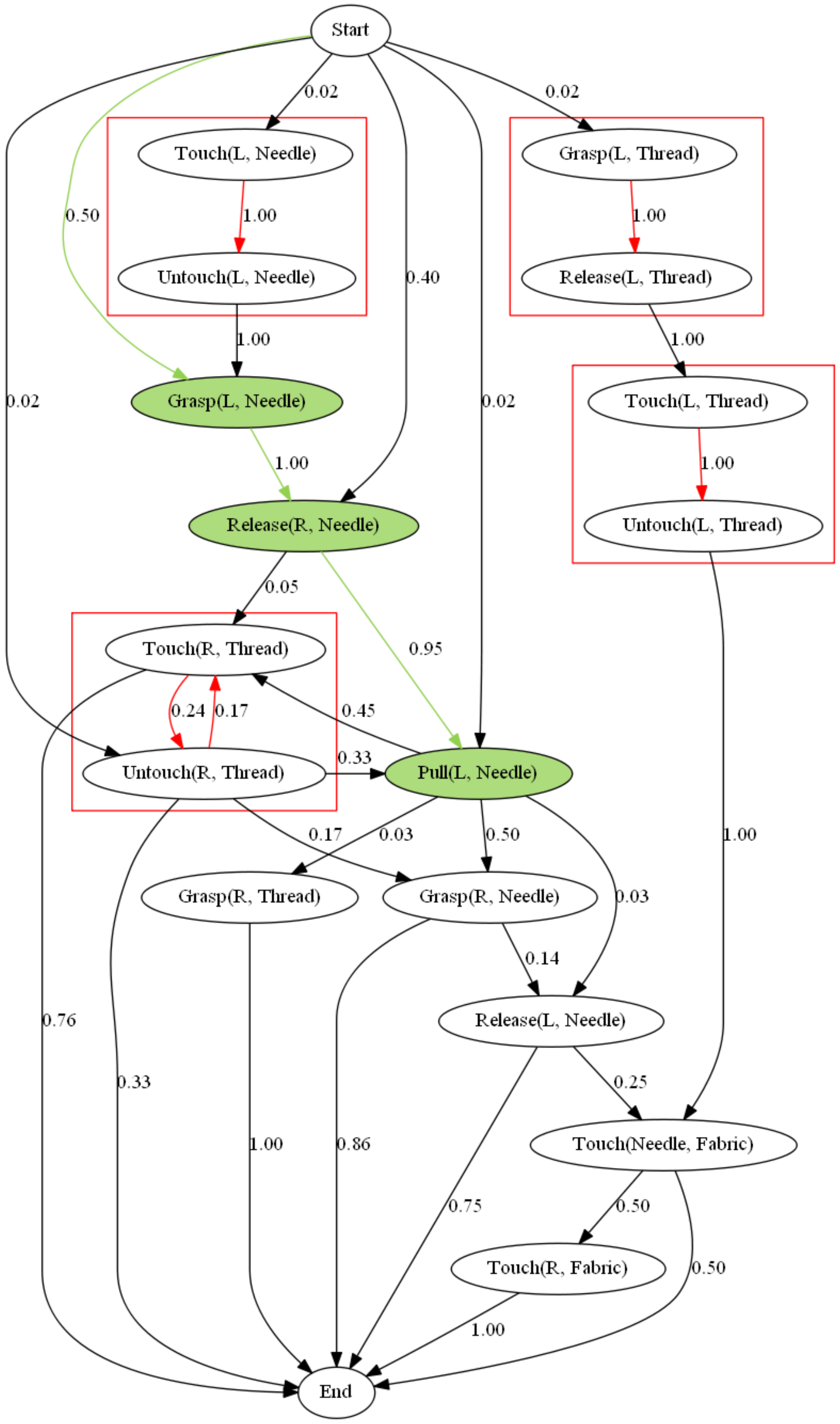


Figure C.10: State graph for Suturing G6 performed by experts.

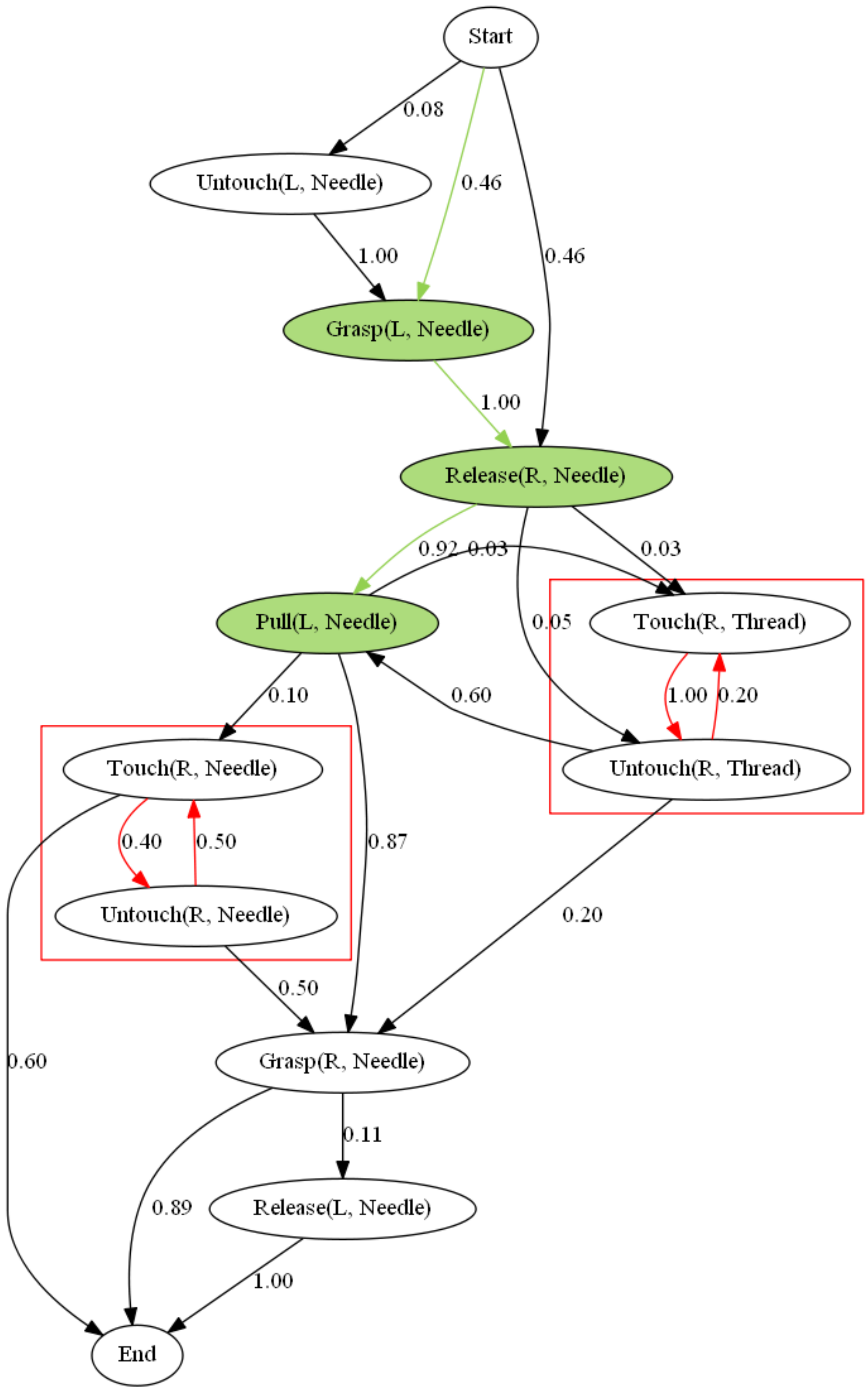


Figure C.11: State graph for Suturing G6 performed by intermediates.

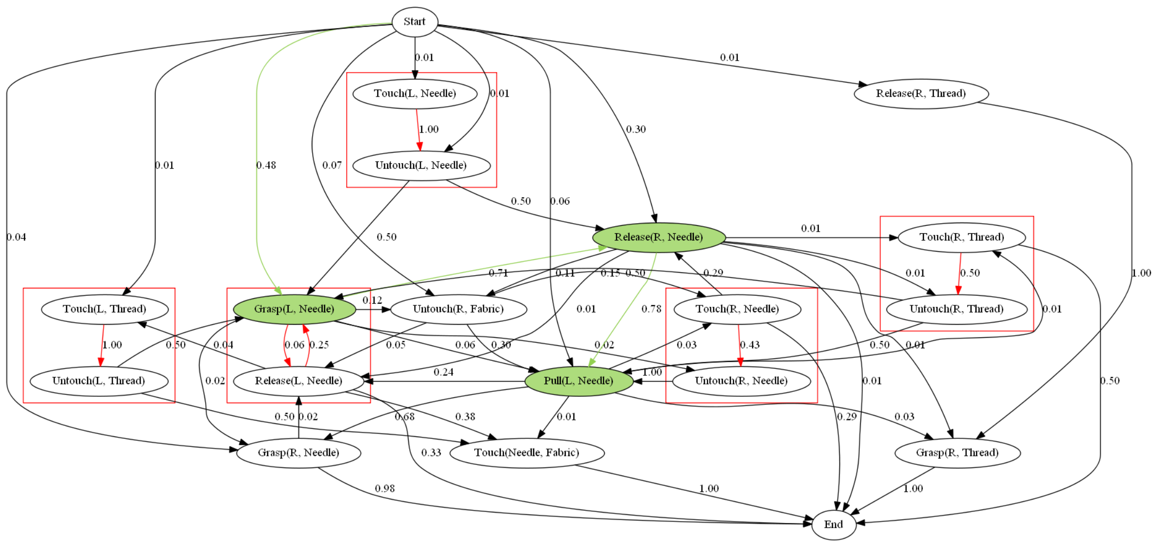


Figure C.12: State graph for Suturing G6 performed by novices.

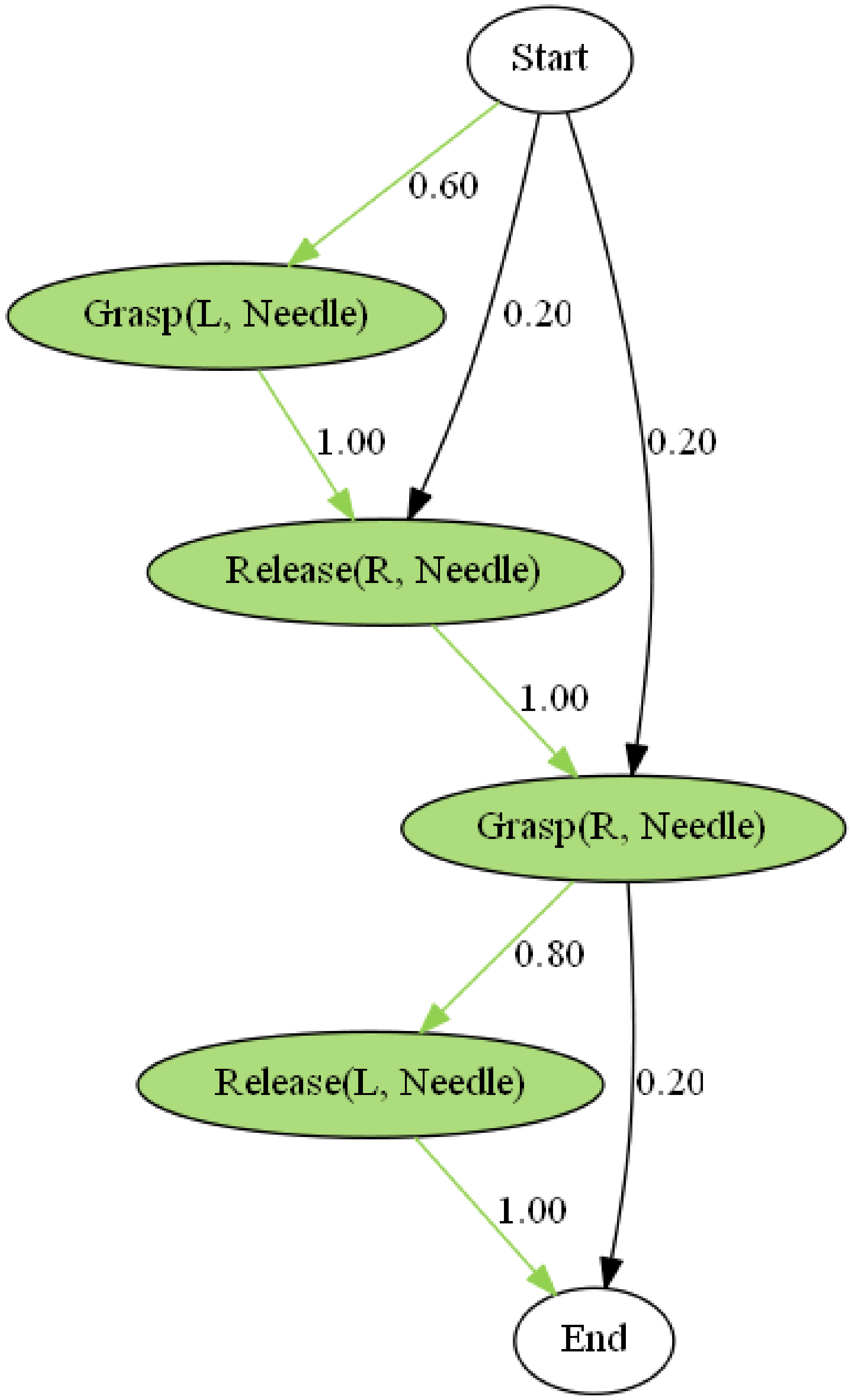


Figure C.13: State graph for Suturing G8 performed by experts.

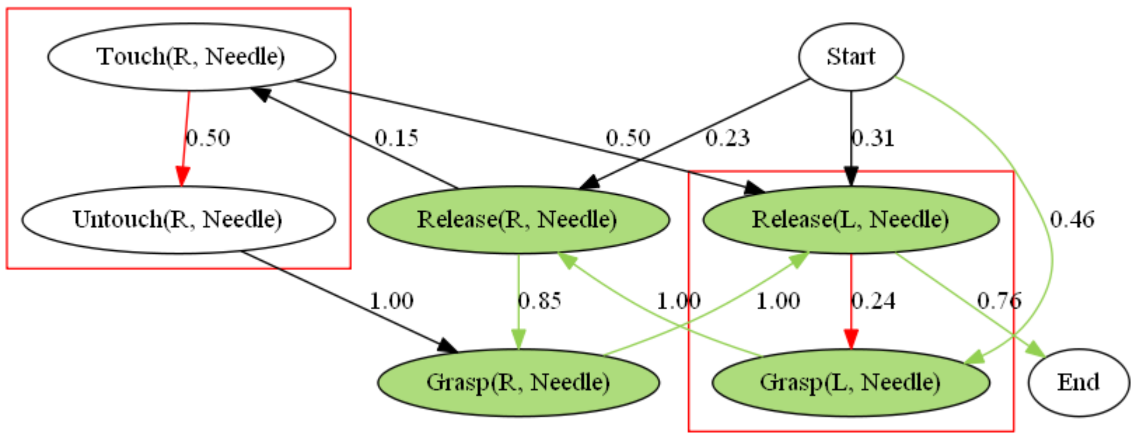


Figure C.14: State graph for Suturing G8 performed by intermediates.

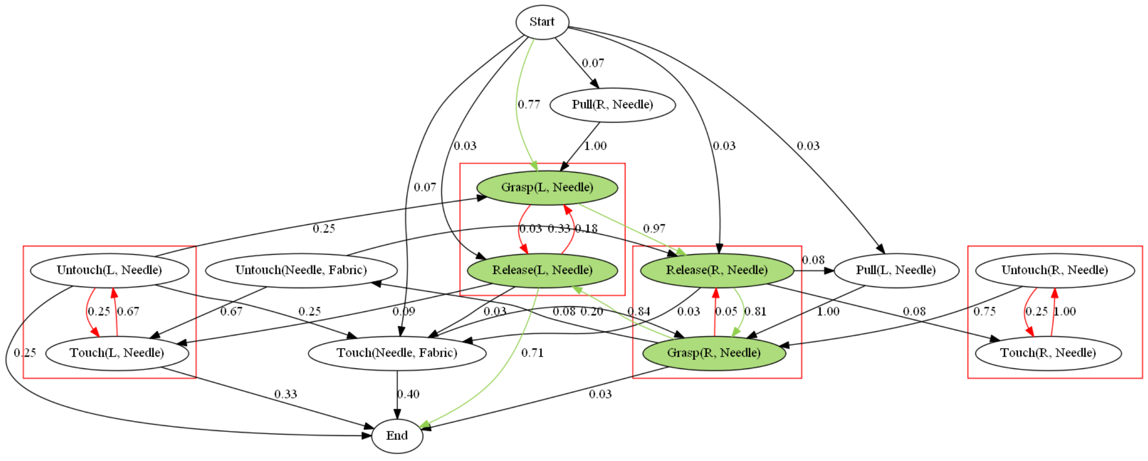


Figure C.15: State graph for Suturing G8 performed by novices.

C.2 Needle Passing

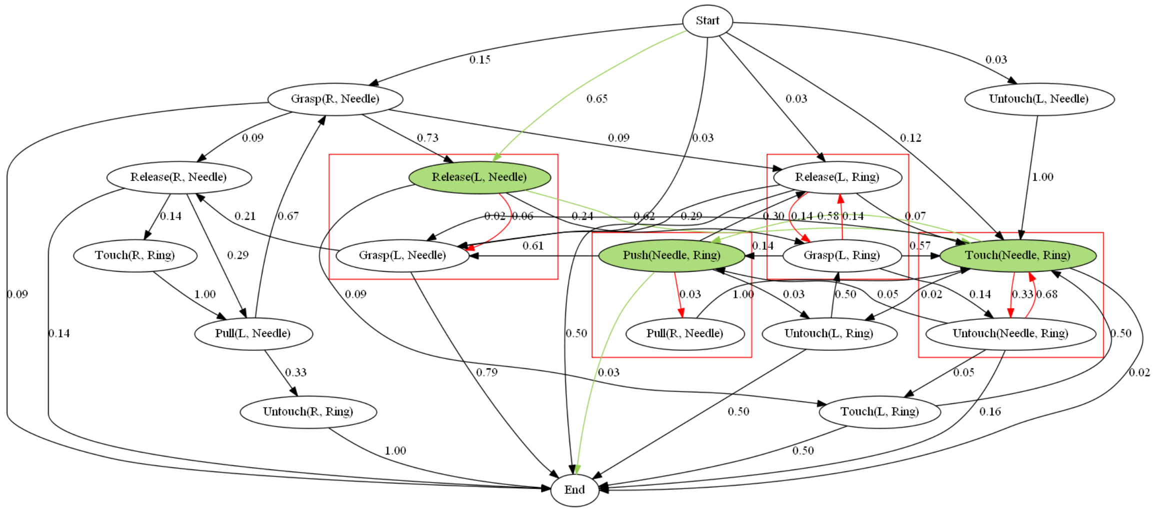


Figure C.16: State graph for Needle Passing G2 performed by experts.

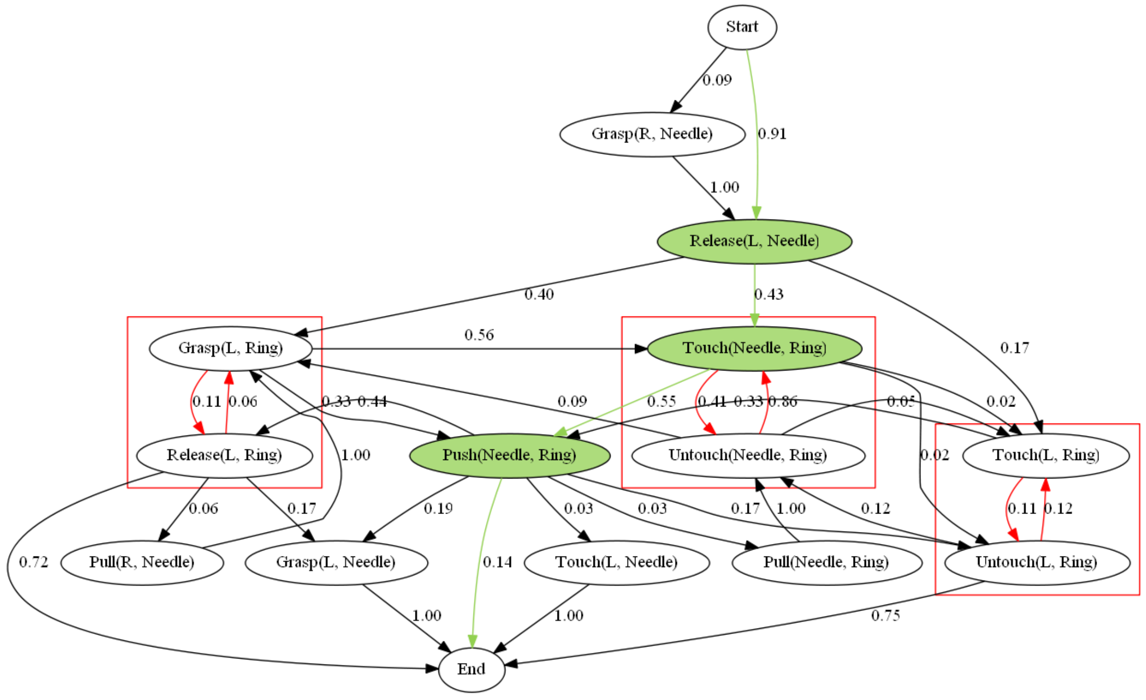


Figure C.17: State graph for Needle Passing G2 performed by intermediates.

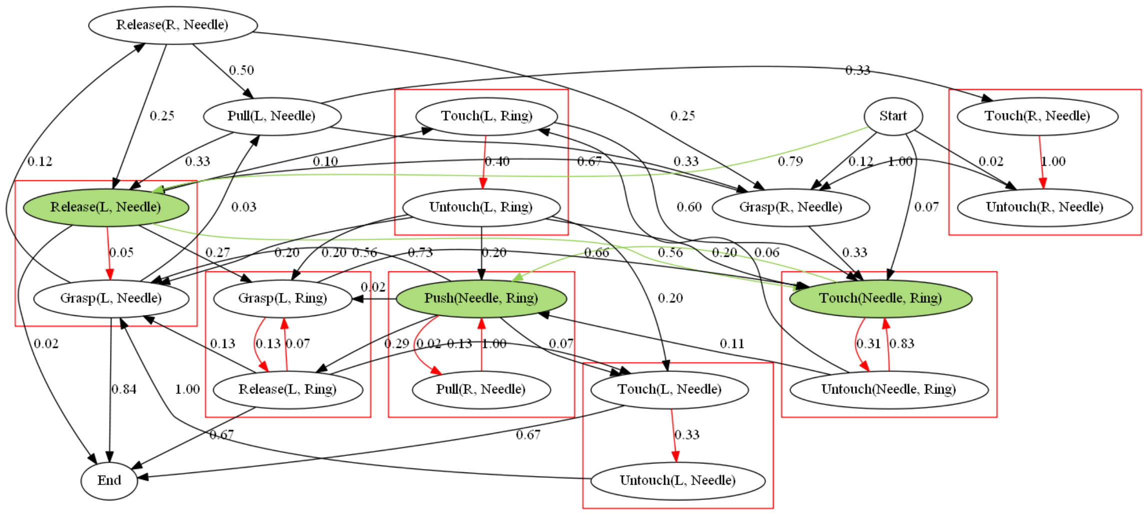


Figure C.18: State graph for Needle Passing G2 performed by novices.

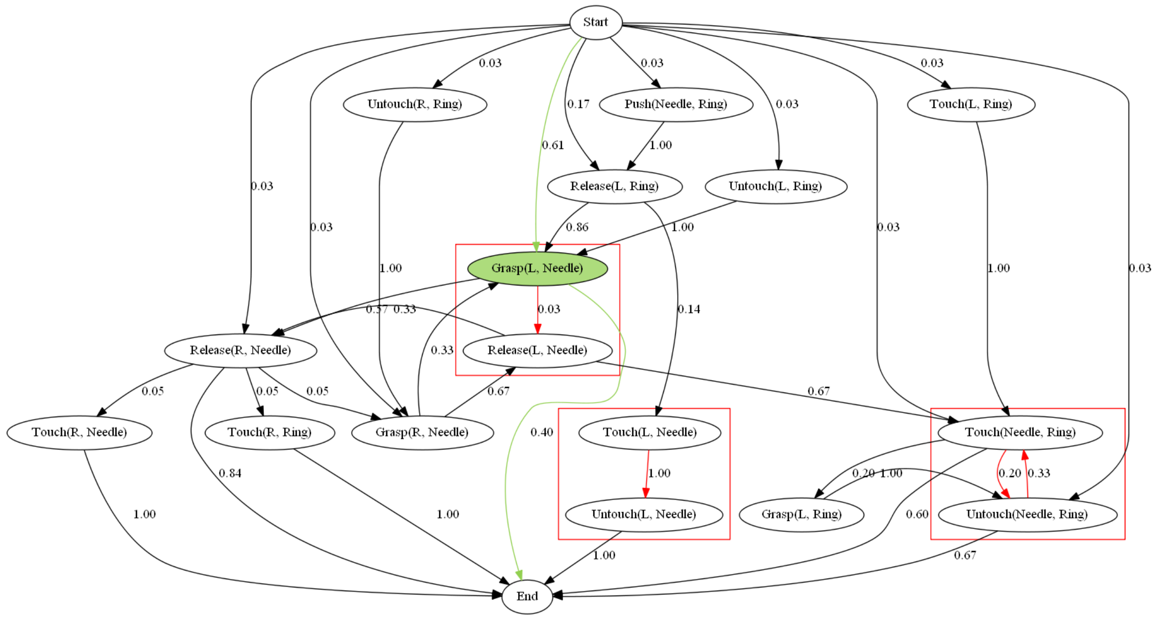


Figure C.19: State graph for Needle Passing G3 performed by experts.

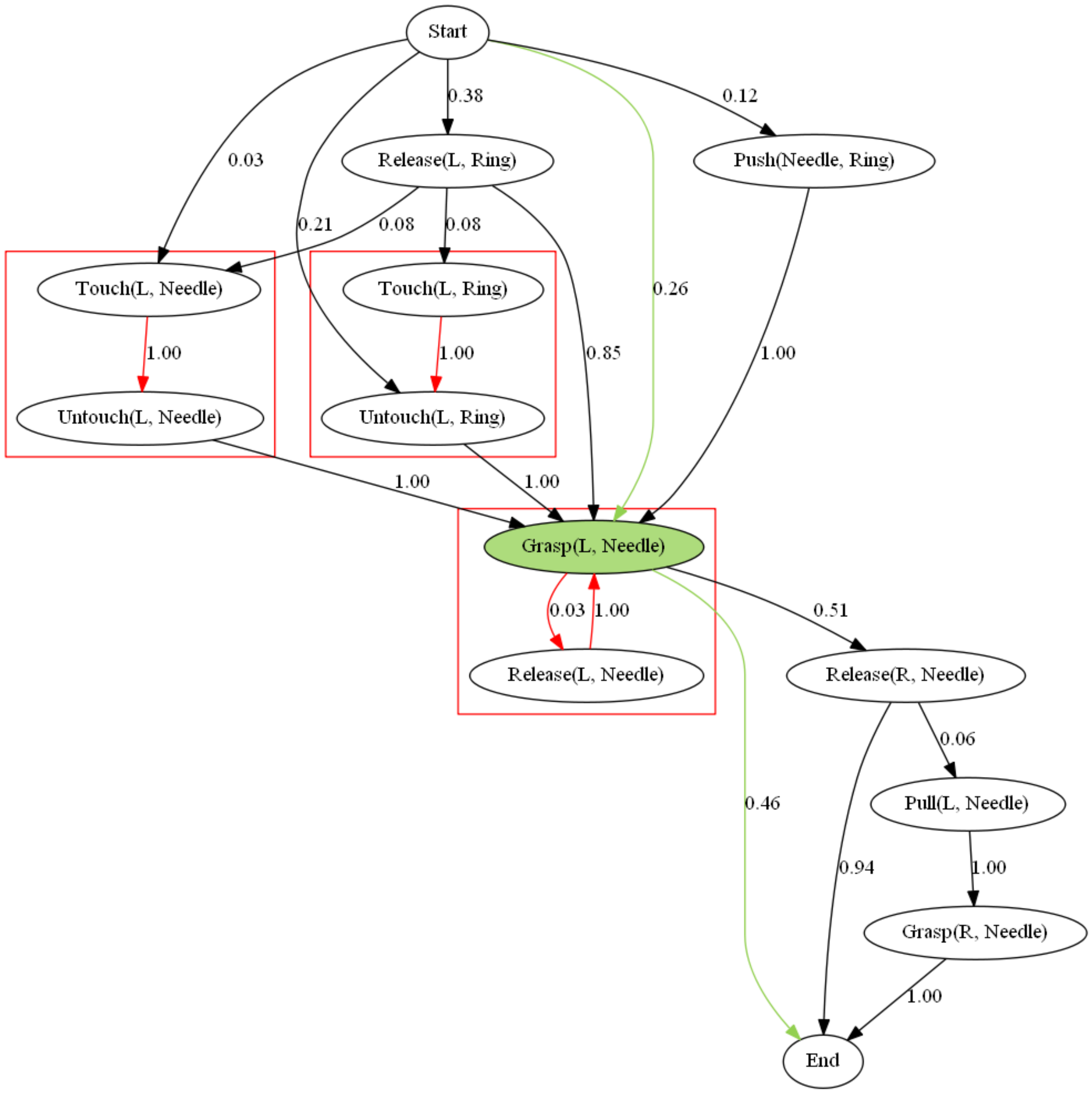


Figure C.20: State graph for Needle Passing G3 performed by intermediates.

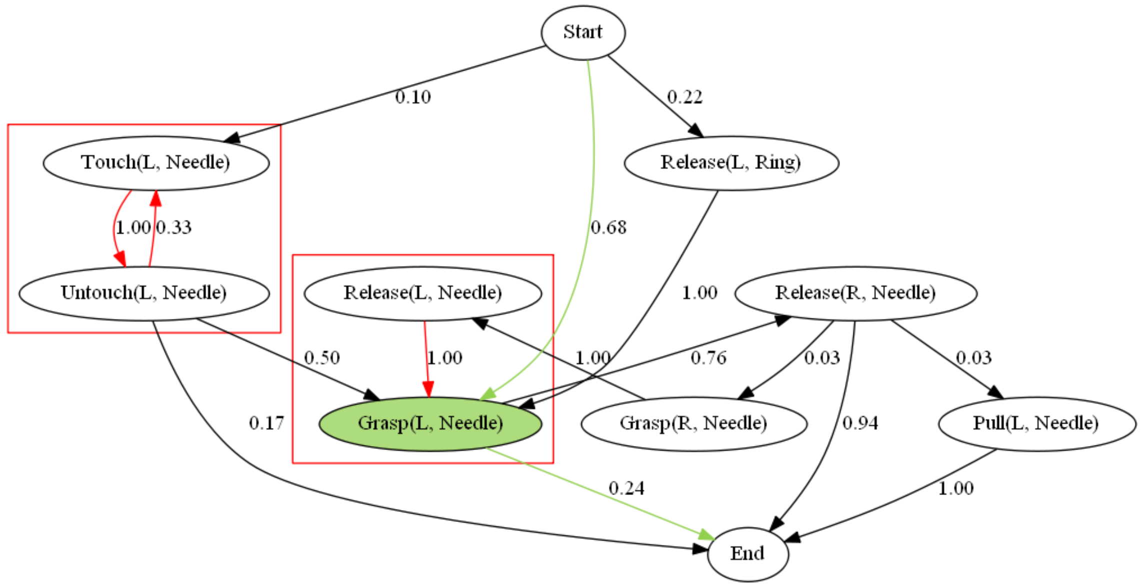


Figure C.21: State graph for Needle Passing G3 performed by novices.

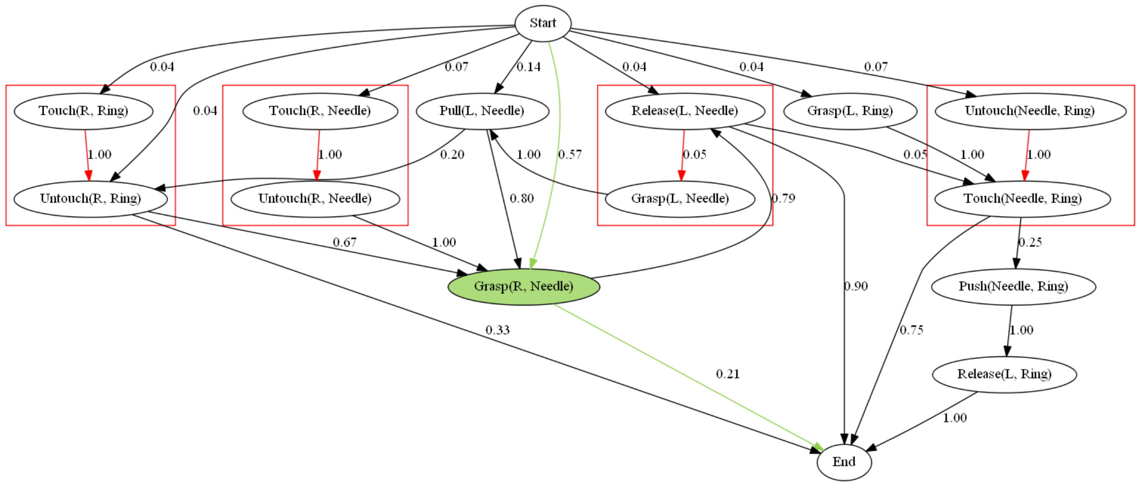


Figure C.22: State graph for Needle Passing G4 performed by experts.

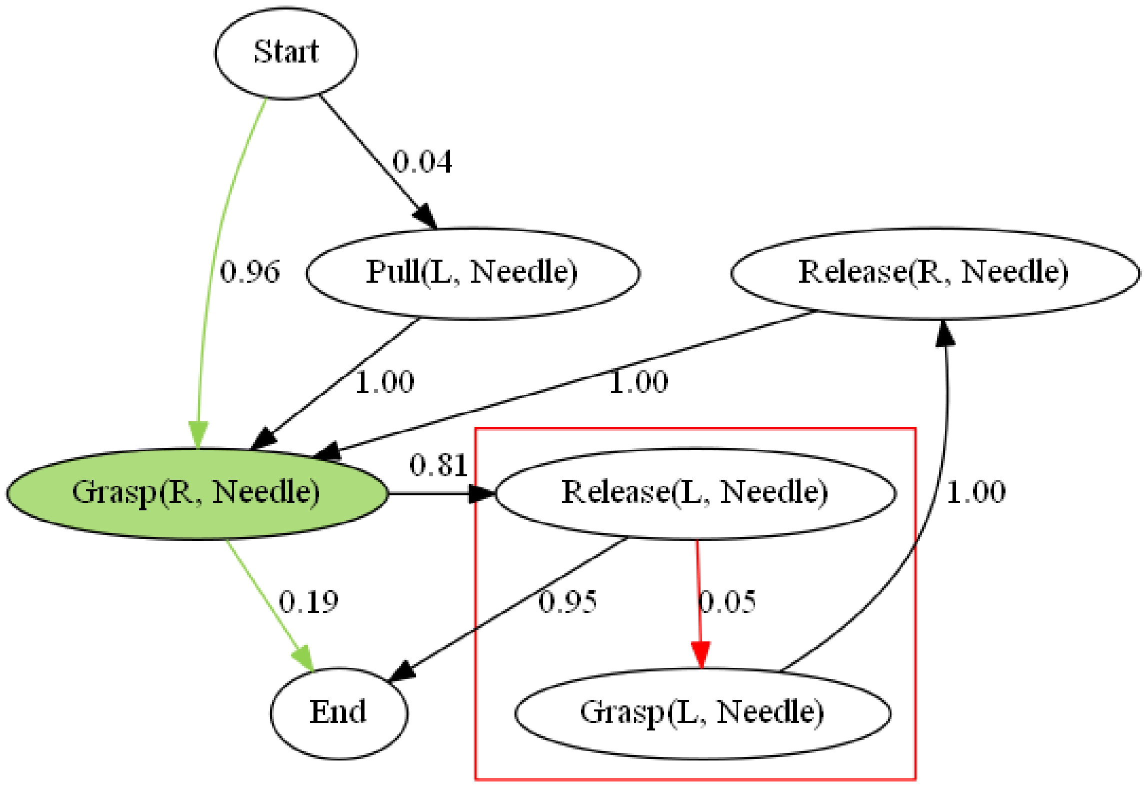


Figure C.23: State graph for Needle Passing G4 performed by intermediates.

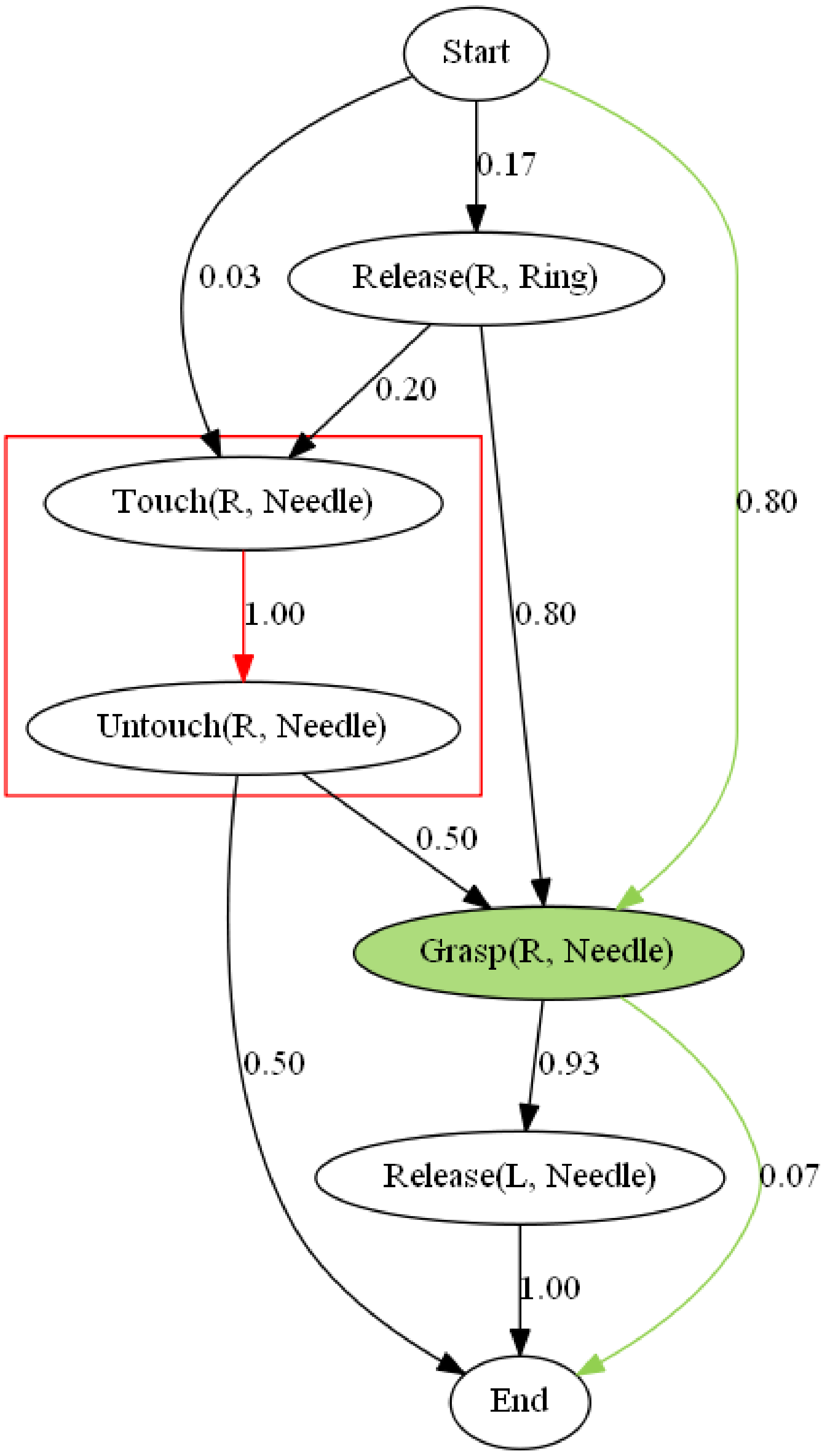


Figure C.24: State graph for Needle Passing G4 performed by novices.

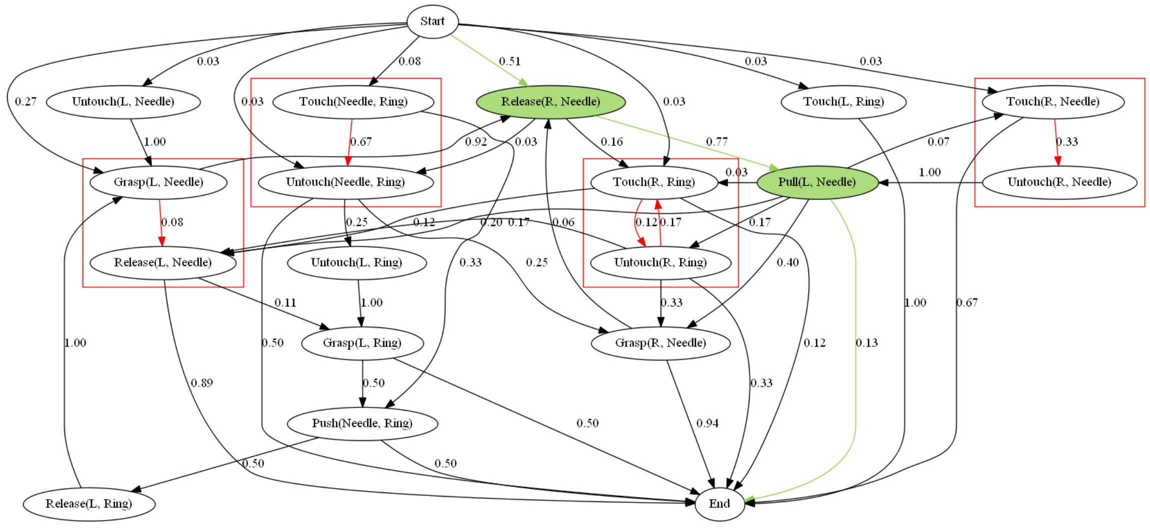


Figure C.25: State graph for Needle Passing G6 performed by experts.

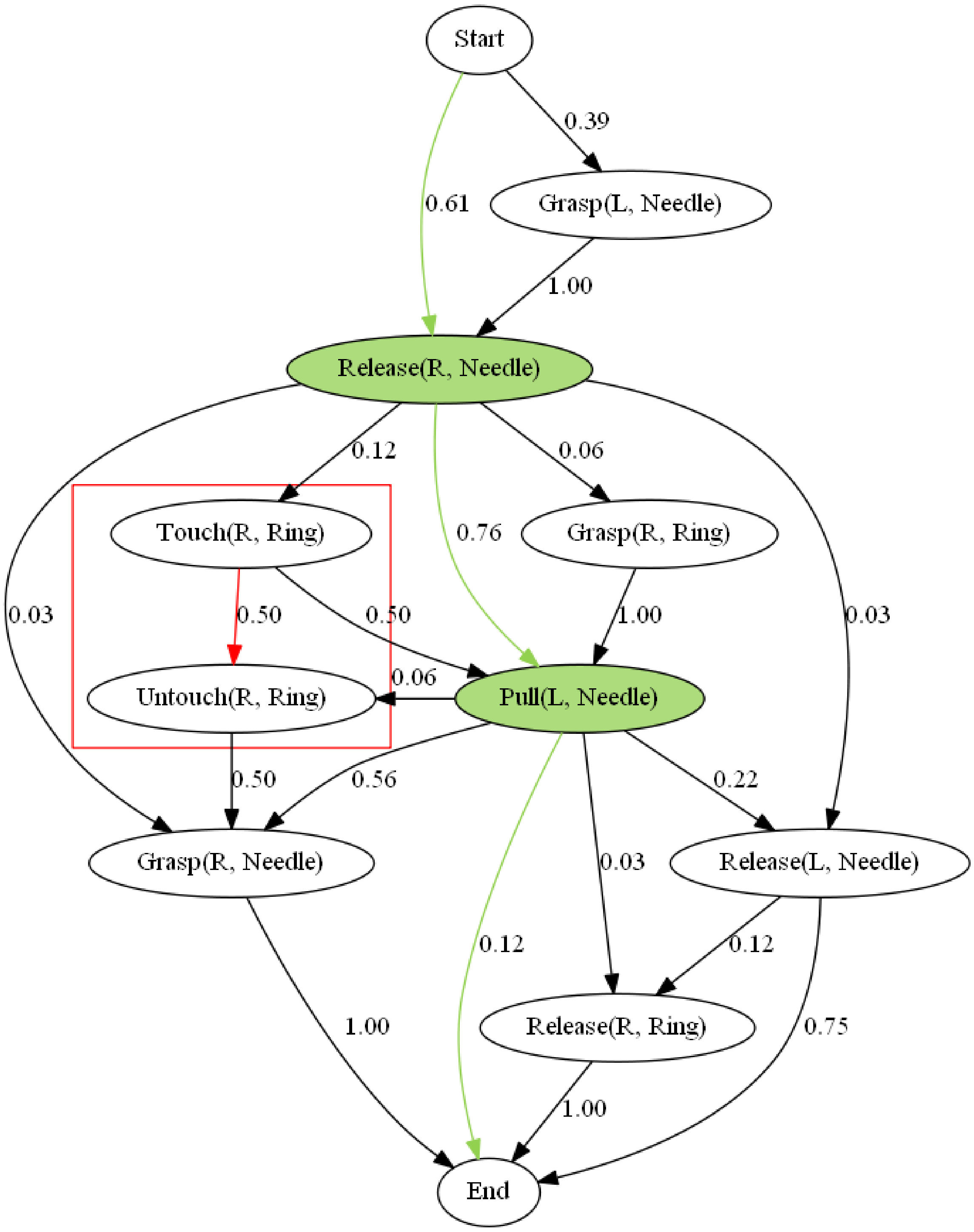


Figure C.26: State graph for Needle Passing G6 performed by intermediates.

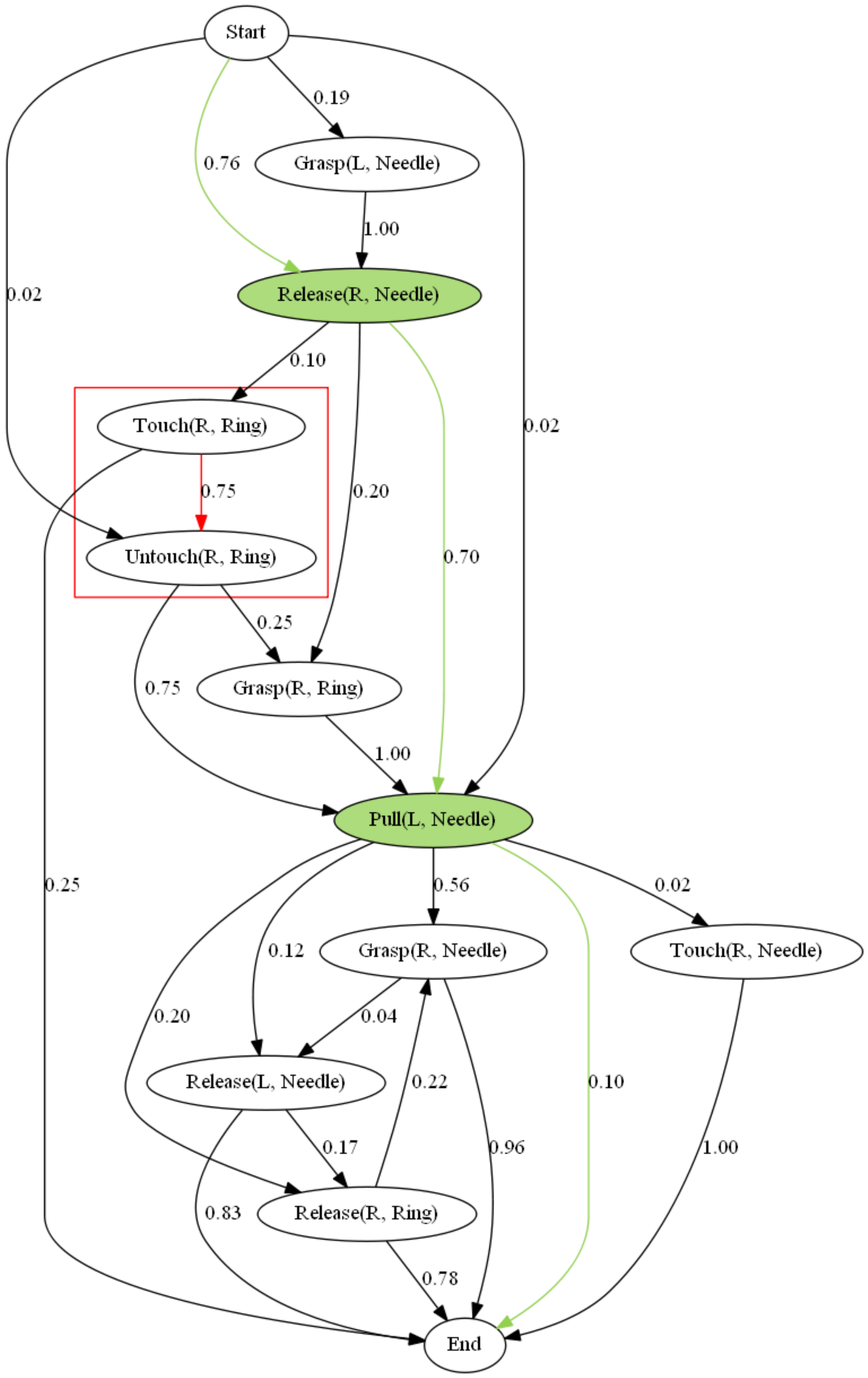


Figure C.27: State graph for Needle Passing G6 performed by novices.

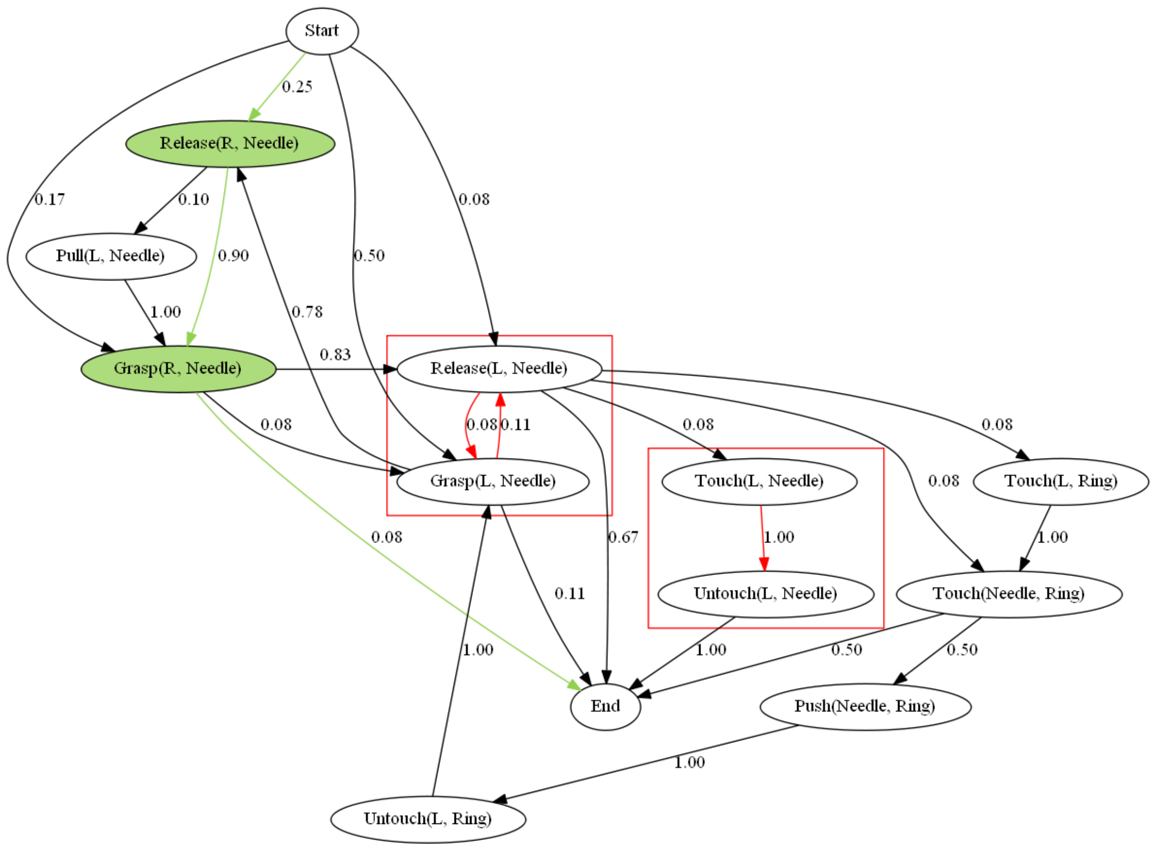


Figure C.28: State graph for Needle Passing G8 performed by experts.

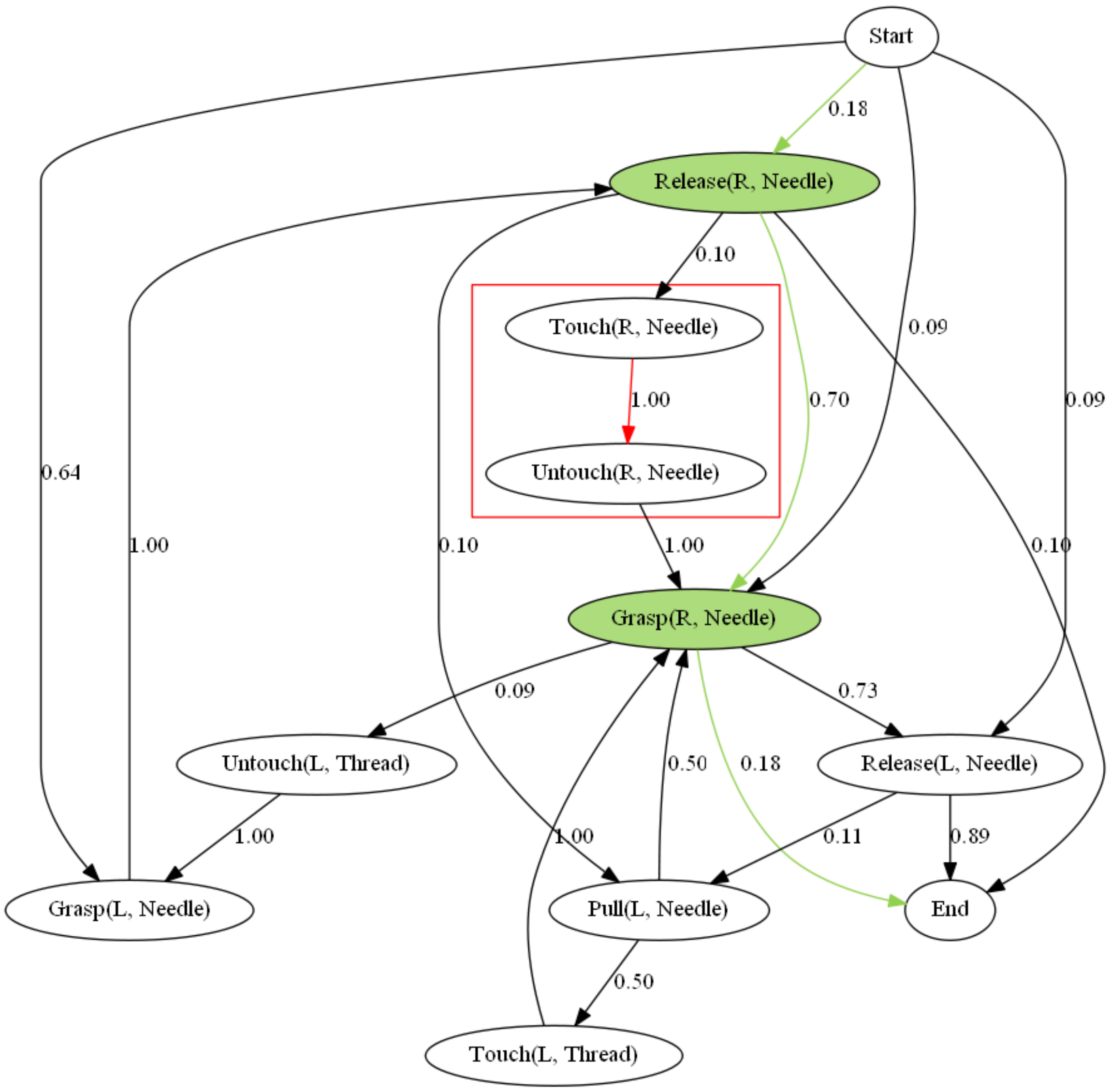


Figure C.29: State graph for Needle Passing G8 performed by intermediates.

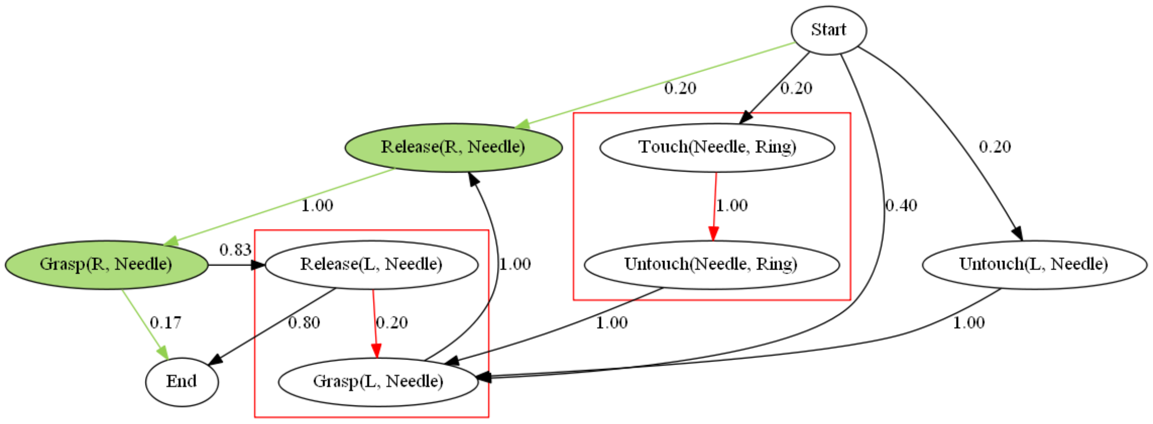


Figure C.30: State graph for Needle Passing G8 performed by novices.

C.3 Knot Tying

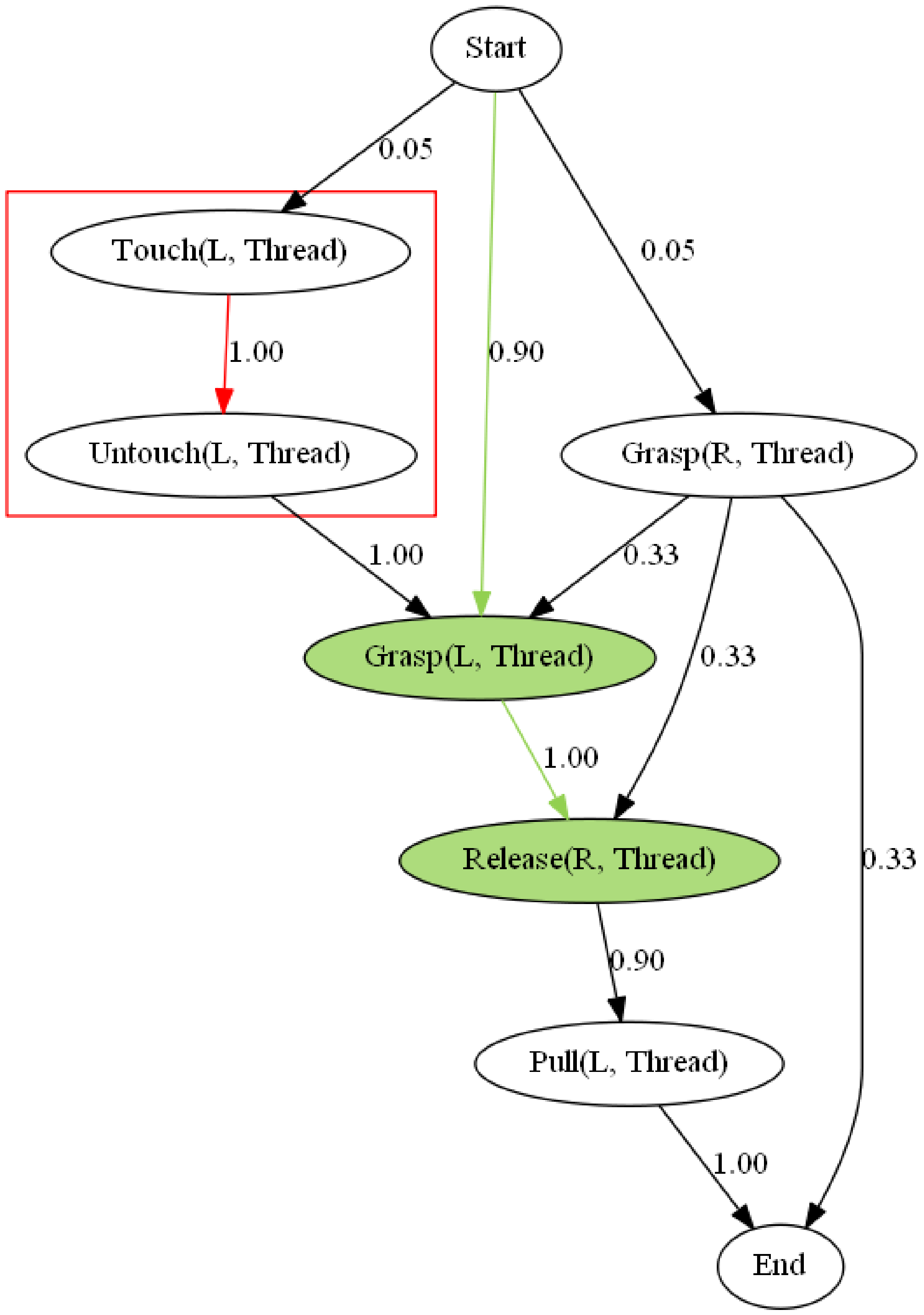


Figure C.31: State graph for Knot Tying G12 performed by experts.

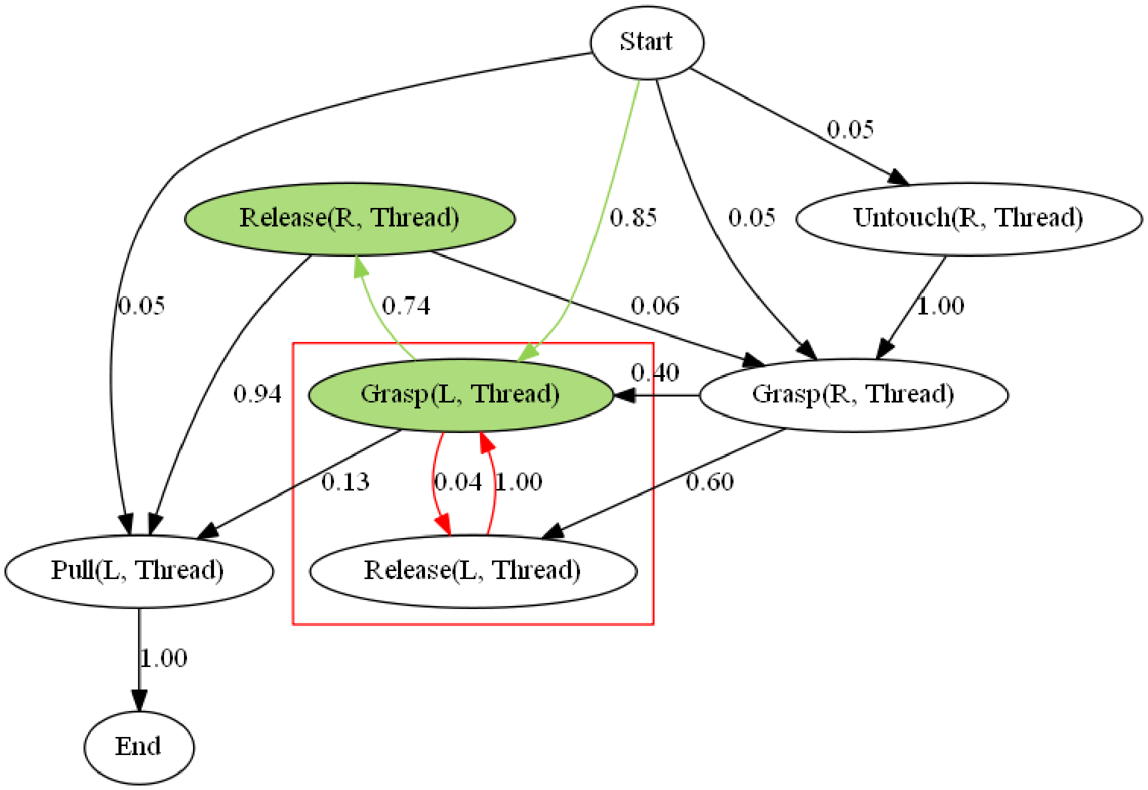


Figure C.32: State graph for Knot Tying G12 performed by intermediates.

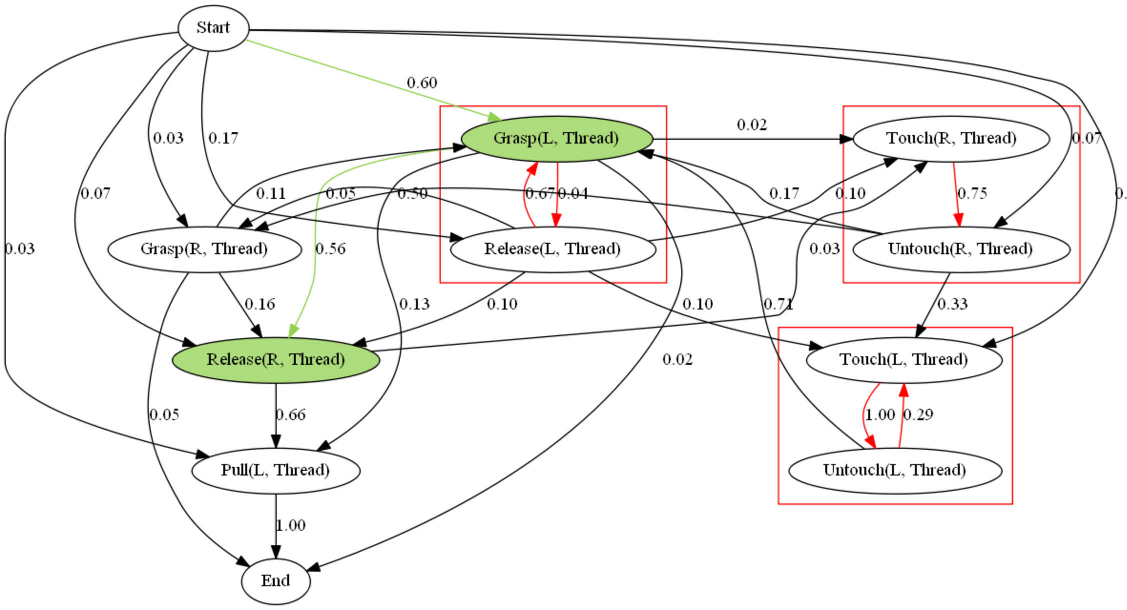


Figure C.33: State graph for Knot Tying G12 performed by novices.

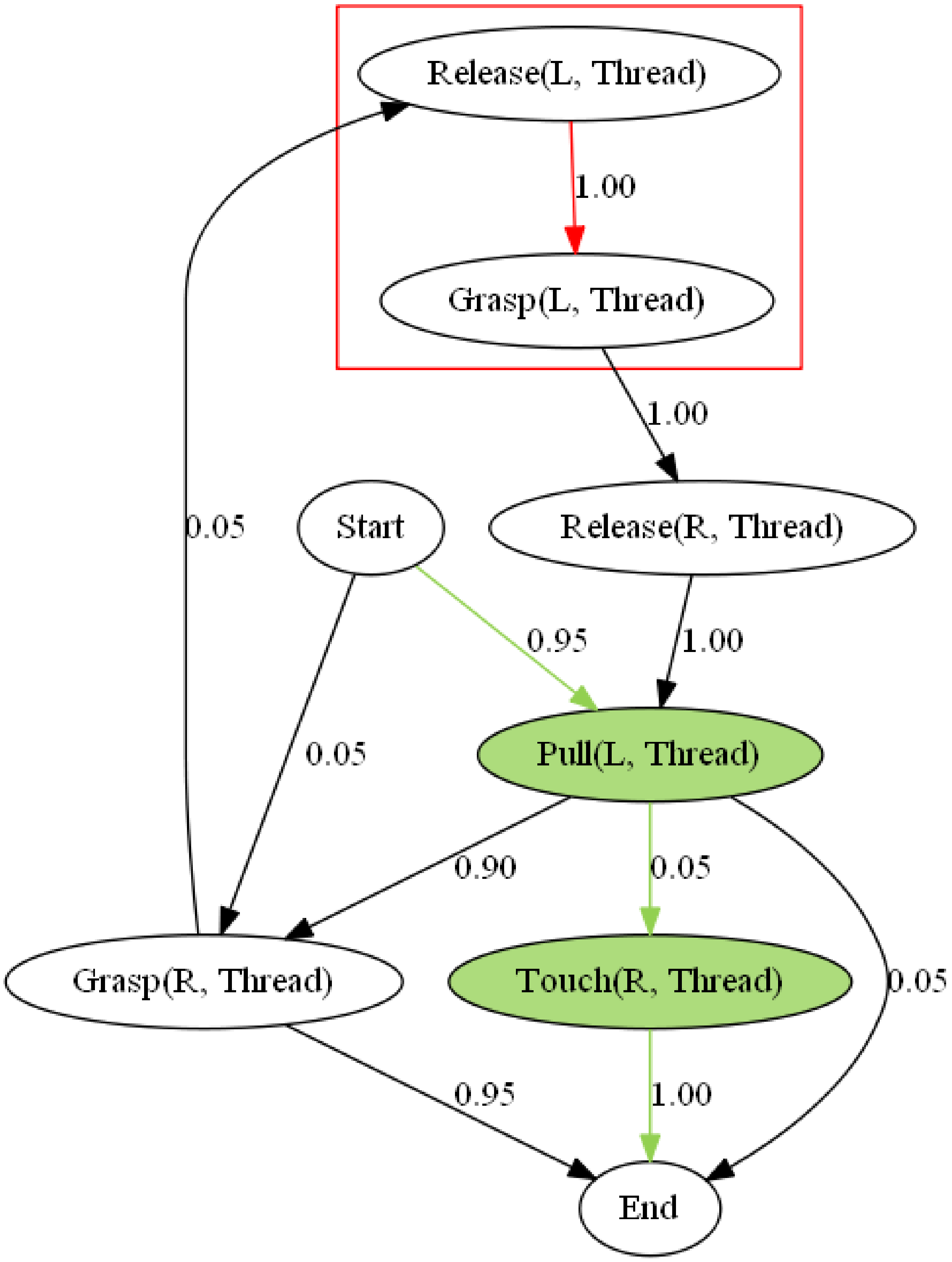


Figure C.34: State graph for Knot Tying G13 performed by experts.

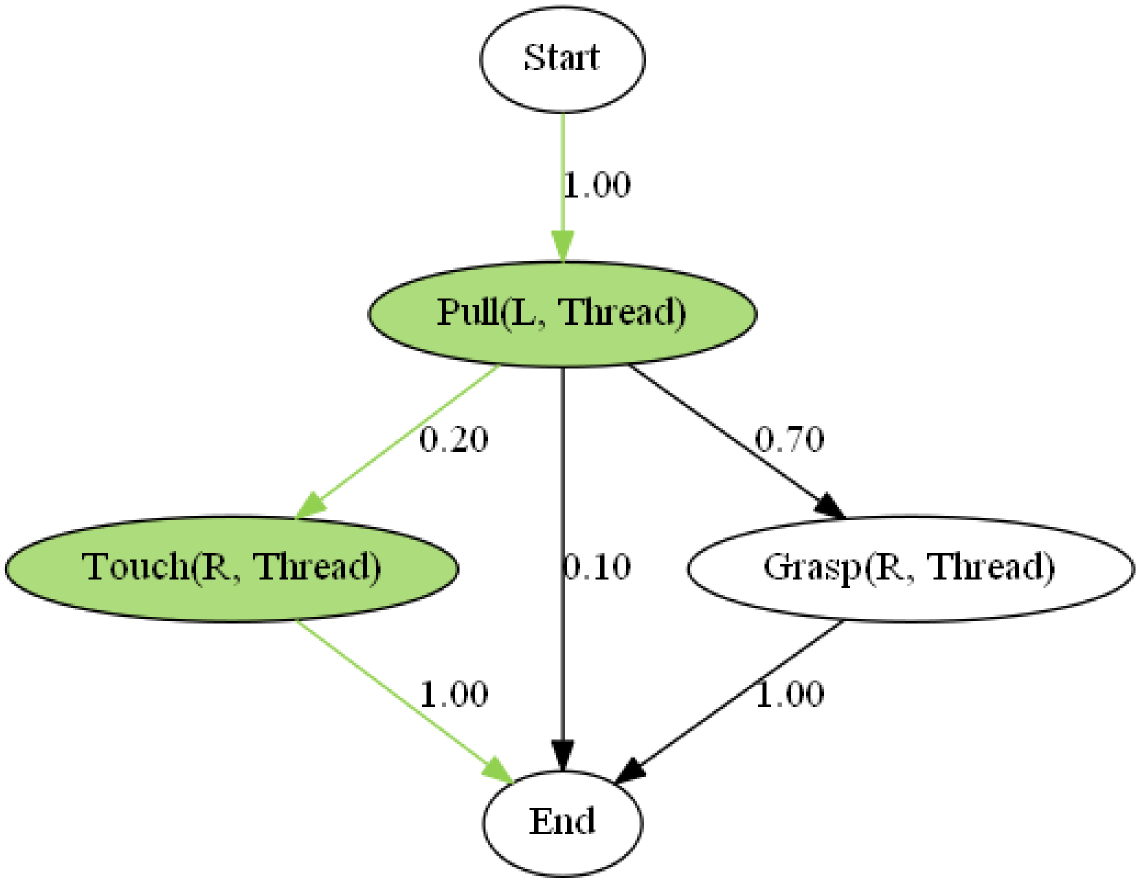


Figure C.35: State graph for Knot Tying G13 performed by intermediates.

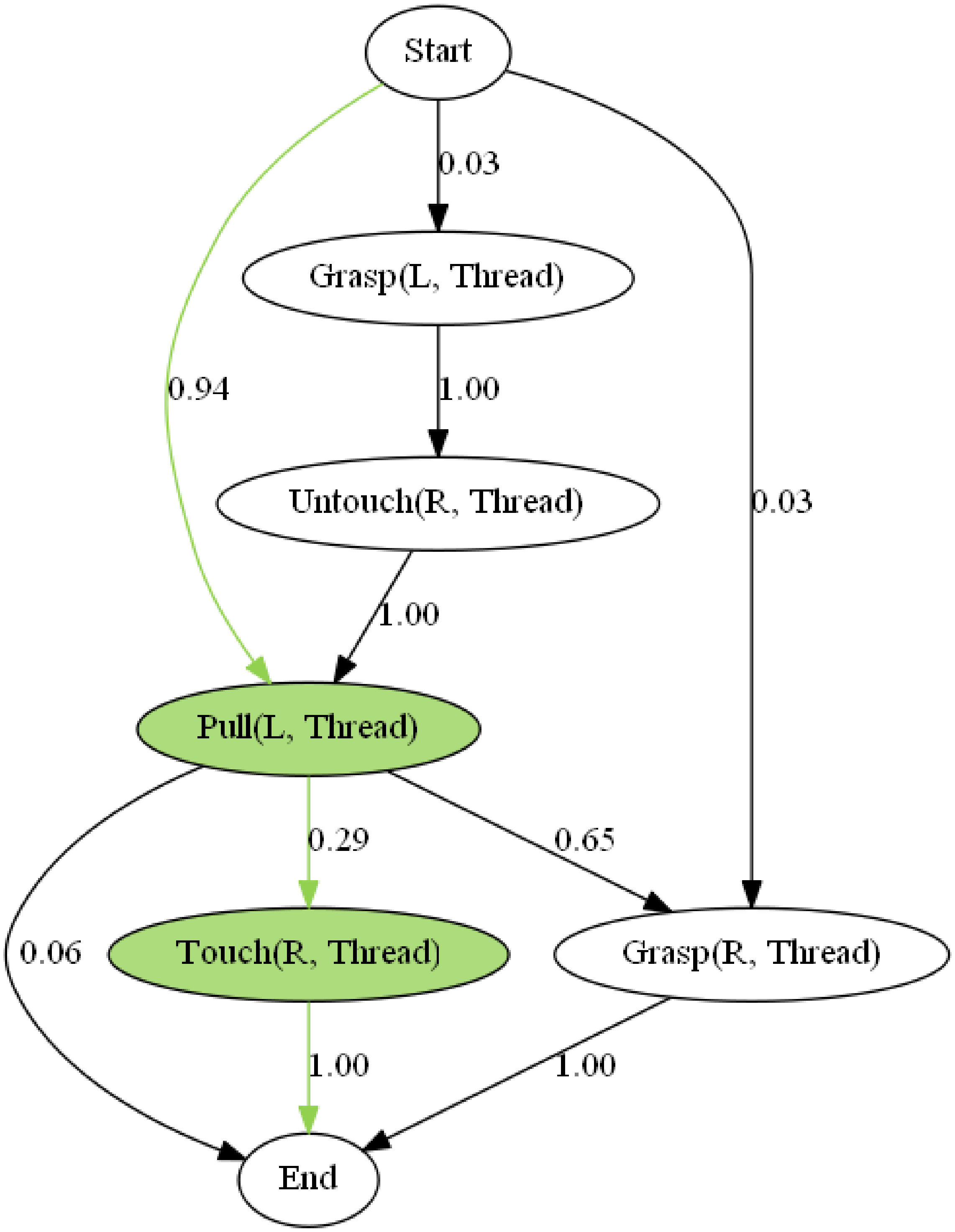


Figure C.36: State graph for Knot Tying G13 performed by novices.

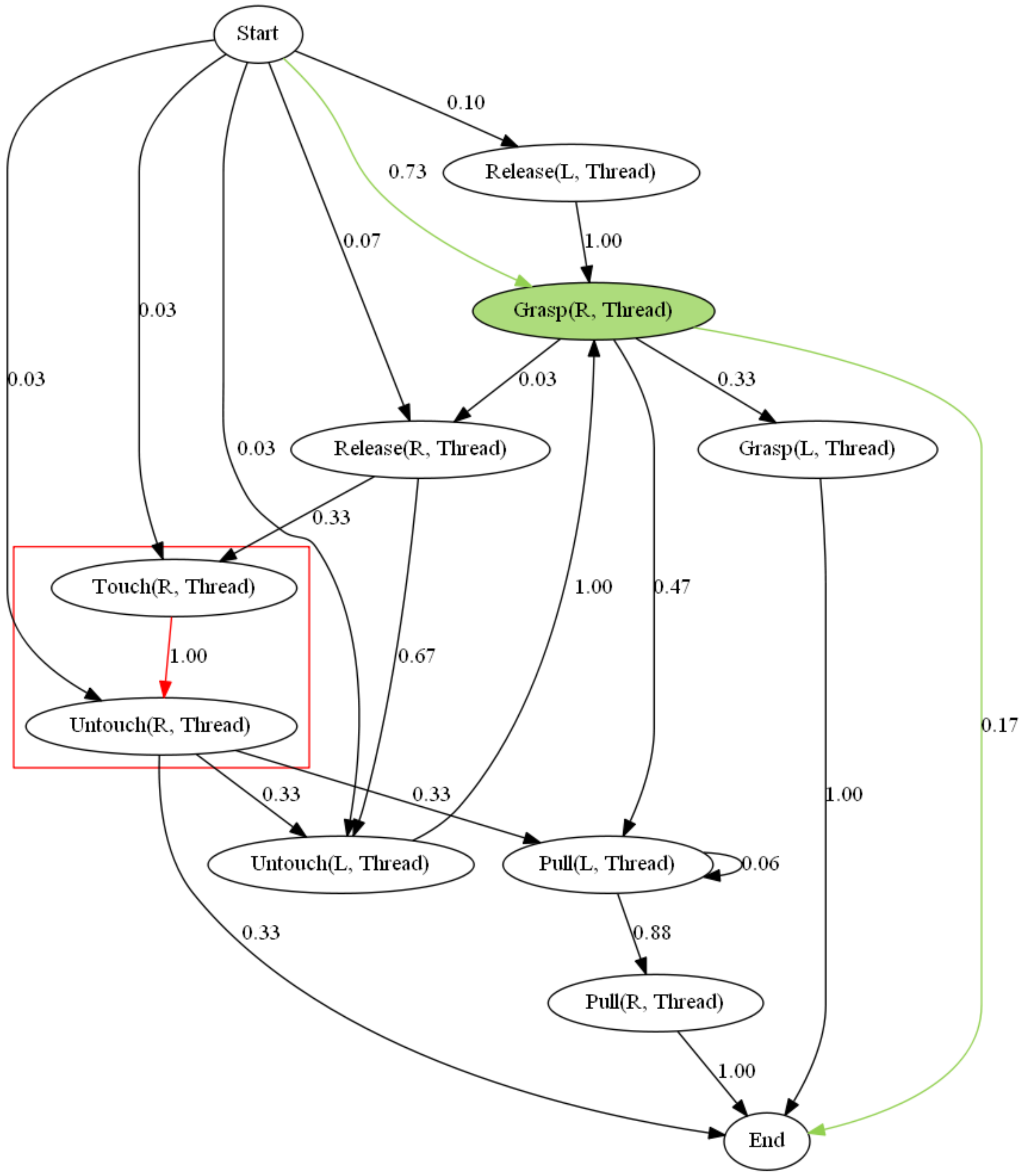


Figure C.37: State graph for Knot Tying G14 performed by experts.

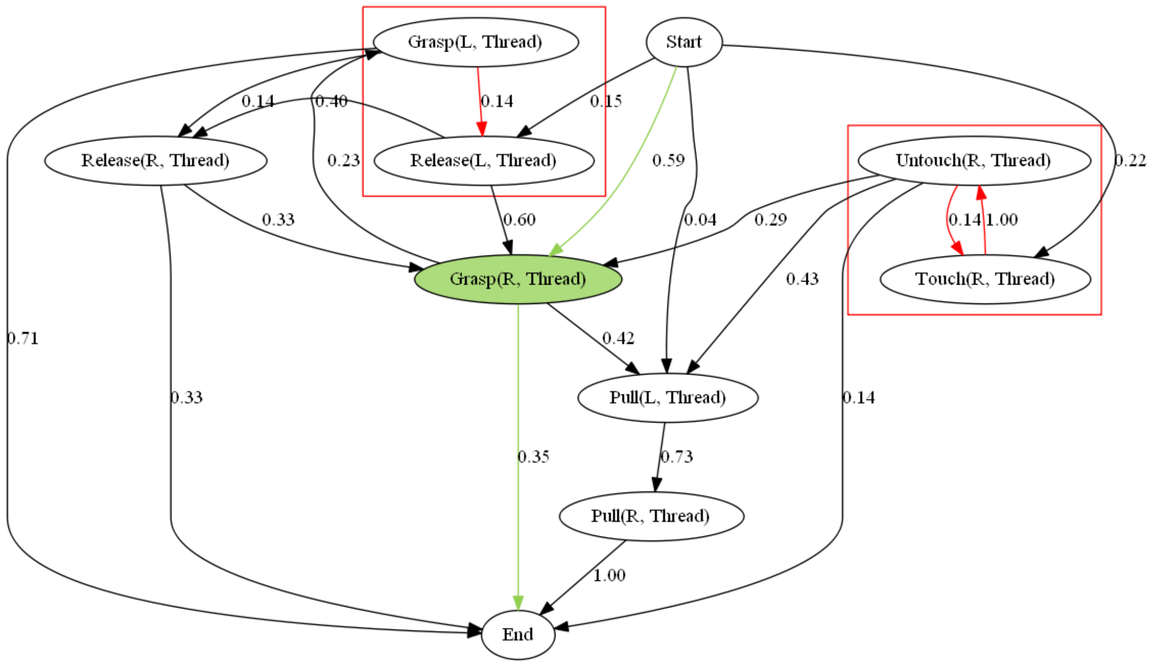


Figure C.38: State graph for Knot Tying G14 performed by intermediates.

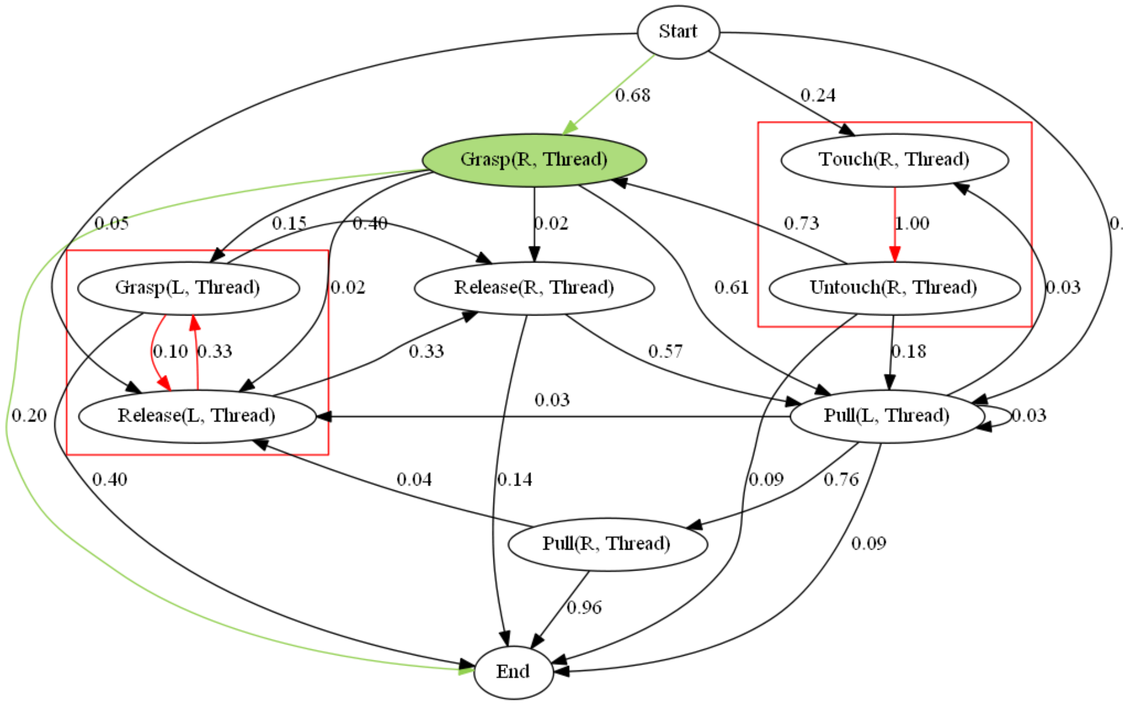


Figure C.39: State graph for Knot Tying G14 performed by novices.

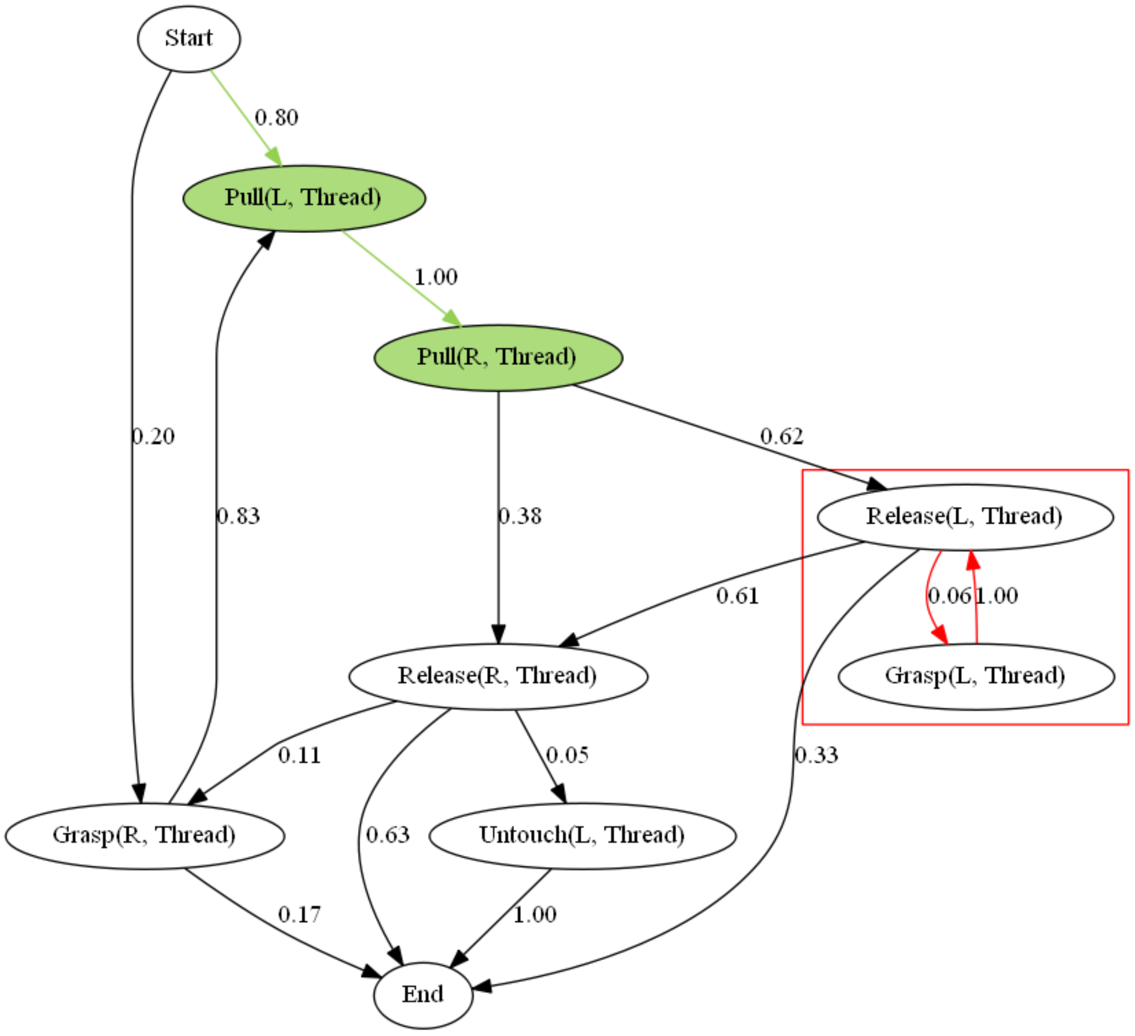


Figure C.40: State graph for Knot Tying G15 performed by experts.

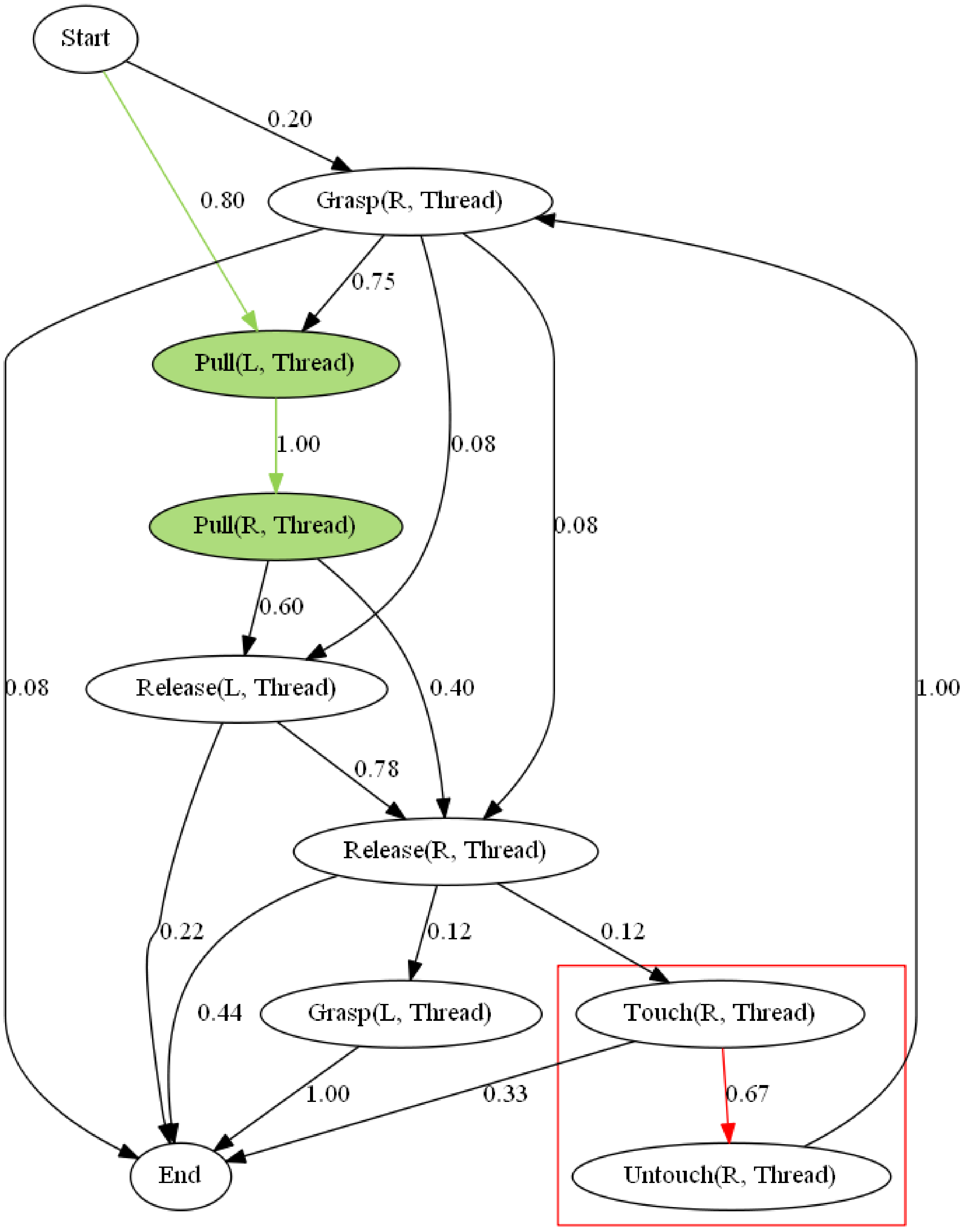


Figure C.41: State graph for Knot Tying G15 performed by intermediates.

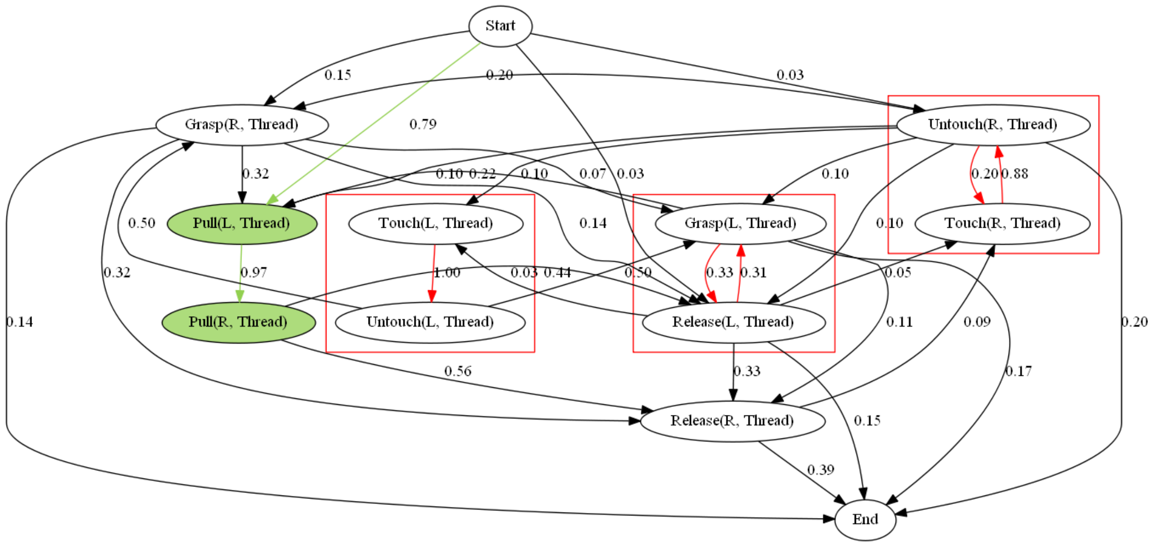


Figure C.42: State graph for Knot Tying G15 performed by novices.

References

[1] B. S. Peters, P. R. Armijo, C. Krause, S. A. Choudhury, and D. Oleynikov, "Review of emerging surgical robotic technology," *Surgical Endoscopy*, vol. 32, no. 4, pp. 1636–1655, 2018.

[2] G. I. Barbash, "New technology and health care costs–the case of robot-assisted surgery," *The New England Journal of Medicine*, vol. 363, no. 8, pp. 701–704, 2010.

[3] Intuitive Surgical Inc., "The da Vinci Surgical System."

[4] Intuitive Surgical Inc., "Intuitive reaches 10 million procedures performed using da Vinci Surgical Systems." url = https://isrg.intuitive.com/news-releases/news-release-details/intuitive-reaches-10-million-procedures-performed-using-da-vinci, 2021.

[5] R. Rajkumar, I. Lee, L. Sha, and J. Stankovic, "Cyber-physical systems: the next computing revolution," in *Proceedings of the 47th Design Automation Conference*, pp. 731–736, IEEE, 2010.

[6] L. Maier-Hein, S. S. Vedula, S. Speidel, N. Navab, R. Kikinis, A. Park, M. Eisenmann, H. Feussner, G. Forestier, S. Giannarou, *et al.*, "Surgical data science for next-generation interventions," *Nature Biomedical Engineering*, vol. 1, no. 9, pp. 691–696, 2017.

[7] Computational Interaction and Robotics Laboratory, “JHU-ISI Gesture and Skill Assessment Working Set (JIGSAWS).” `https://cirl.lcsr.jhu.edu/research/hmm/datasets/jigsaws_release/`.

[8] B. van Amsterdam, M. Clarkson, and D. Stoyanov, “Gesture recognition in robotic surgery: A review,” *IEEE Transactions on Biomedical Engineering*, vol. 68, no. 6, pp. 2021–2035, 2021.

[9] F. Lalys and P. Jannin, “Surgical process modelling: a review,” *International Journal of Computer Assisted Radiology and Surgery*, vol. 9, pp. 495–511, 2014.

[10] D. Neumuth, F. Loebe, H. Herre, and T. Neumuth, “Modeling surgical processes: A four-level translational approach,” *Artificial Intelligence in Medicine*, vol. 51, no. 3, pp. 147–161, 2011.

[11] F. Falezza, N. Piccinelli, G. De Rossi, A. Roberti, G. Kronreif, F. Setti, P. Fiorini, and R. Muradore, “Modeling of surgical procedures using statecharts for semi-autonomous robotic surgery,” *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 4, pp. 888–899, 2021.

[12] L. Tao, E. Elhamifar, S. Khudanpur, G. D. Hager, and R. Vidal, “Sparse hidden markov models for surgical gesture classification and skill evaluation,” in *Information Processing in Computer-Assisted Interventions*, vol. 7330, pp. 167–177, Springer, 2012.

[13] B. Varadarajan, C. Reiley, H. Lin, S. Khudanpur, and G. Hager, “Data-derived models for segmentation with application to surgical assessment and training,” in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009*, pp. 426–434, Springer, 2009.

[14] M. Ginesi, D. Meli, A. Roberti, N. Sansonetto, and P. Fiorini, “Autonomous task planning and situation awareness in robotic surgery,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3144–3150, IEEE, 2020.

[15] D. A. Inouye, R. Ma, J. H. Nguyen, J. Laca, R. Kocielnik, A. Anandkumar, and A. J. Hung, "Assessing the efficacy of dissection gestures in robotic surgery," *Journal of Robotic Surgery*, vol. 17, no. 2, pp. 597–603, 2022.

[16] Z. Li, K. Hutchinson, and H. Alemzadeh, "Runtime detection of executional errors in robot-assisted surgery," in *2022 International Conference on Robotics and Automation (ICRA)*, pp. 3850–3856, IEEE, 2022.

[17] M. S. Yasar, D. Evans, and H. Alemzadeh, "Context-aware monitoring in robotic surgery," in *2019 International Symposium on Medical Robotics (ISMR)*, pp. 1–7, IEEE, 2019.

[18] M. S. Yasar and H. Alemzadeh, "Real-time context-aware detection of unsafe events in robot-assisted surgery," in *2020 50th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pp. 385–397, IEEE, 2020.

[19] M. Allan, S. Kondo, S. Bodenstedt, S. Leger, R. Kadkhodamohammadi, I. Luengo, F. Fuentes, E. Flouty, A. Mohammed, M. Pedersen, *et al.*, "2018 robotic scene segmentation challenge," *arXiv preprint arXiv:2001.11190*, 2020.

[20] M. Wagner, B.-P. Müller-Stich, A. Kisilenko, D. Tran, P. Heger, L. Mündermann, D. M. Lubotsky, B. Müller, T. Davitashvili, M. Capek, *et al.*, "Comparative validation of machine learning algorithms for surgical workflow and skill analysis with the HeiChole benchmark," *Medical Image Analysis*, vol. 86, p. 102770, 2023.

[21] C. I. Nwoye, D. Alapatt, T. Yu, A. Vardazaryan, F. Xia, Z. Zhao, T. Xia, F. Jia, Y. Yang, H. Wang, *et al.*, "Cholectriplet2021: A benchmark challenge for surgical action triplet recognition," *Medical Image Analysis*, vol. 86, p. 102803, 2023.

[22] C. I. Nwoye, C. Gonzalez, T. Yu, P. Mascagni, D. Mutter, J. Marescaux, and N. Padoy, "Recognition of instrument-tissue interactions in endoscopic

videos via action triplets," in *Medical Image Computing and Computer Assisted Intervention–MICCAI 2020*, pp. 364–374, Springer, 2020.

[23] D. Katić, A.-L. Wekerle, F. Gärtner, H. Kenngott, B. P. Müller-Stich, R. Dillmann, and S. Speidel, "Knowledge-driven formalization of laparoscopic surgeries for rule-based intraoperative context-aware assistance," in *Information Processing in Computer-Assisted Interventions*, pp. 158–167, Springer, 2014.

[24] D. Kitaguchi, N. Takeshita, H. Hasegawa, and M. Ito, "Artificial intelligence-based computer vision in surgery: Recent advances and future perspectives," *Annals of Gastroenterological Surgery*, vol. 6, no. 1, pp. 29–36, 2022.

[25] B. van Amsterdam, M. J. Clarkson, and D. Stoyanov, "Multi-task recurrent neural network for surgical gesture recognition and progress prediction," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1380–1386, IEEE, 2020.

[26] M. Wagner, S. Bodenstedt, M. Daum, A. Schulze, R. Younis, J. Brandenburg, F. R. Kolbinger, M. Distler, L. Maier-Hein, J. Weitz, B.-P. Ḿuller-Stich, and S. Speidel, "The importance of machine learning in autonomous actions for surgical decision making," *Artificial Intelligence Surgery*, vol. 2, pp. 64–79, 2022.

[27] S. Chakraborty, R. Tomsett, R. Raghavendra, D. Harborne, M. Alzantot, F. Cerutti, M. Srivastava, A. Preece, S. Julier, R. M. Rao, T. D. Kelley, D. Braines, M. Sensoy, C. J. Willis, and P. Gurram, "Interpretability of deep learning models: A survey of results," in *2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, pp. 1–6, IEEE, 2017.

[28] M. Allan, A. Shvets, T. Kurmann, Z. Zhang, R. Duggal, Y.-H. Su, N. Rieke, I. Laina, N. Kalavakonda, S. Bodenstedt, *et al.*, "2017 robotic instrument segmentation challenge," *arXiv preprint arXiv:1902.06426*, 2019.

[29] Z.-L. Ni, G.-B. Bian, X.-H. Zhou, Z.-G. Hou, X.-L. Xie, C. Wang, Y.-J. Zhou, R.-Q. Li, and Z. Li, "RAUNet: Residual attention U-Net for semantic segmentation of cataract surgical instruments," in *Neural Information Processing*, pp. 139–149, Springer, 2019.

[30] N. Yong, P. Grange, and D. Eldred-Evans, "Impact of laparoscopic lens contamination in operating theaters: a study on the frequency and duration of lens contamination and commonly utilized techniques to maintain clear vision," *Surgical Laparoscopy, Endoscopy & Percutaneous Techniques*, vol. 26, no. 4, pp. 286–289, 2016.

[31] J. C. Allers, A. A. Hussein, N. Ahmad, L. Cavuoto, J. F. Wing, R. M. Hayes, N. Hinata, A. M. Bisantz, and K. A. Guru, "Evaluation and impact of workflow interruptions during robot-assisted surgery," *Urology*, vol. 92, pp. 33–37, 2016.

[32] E. Palagonia, E. Mazzone, G. De Naeyer, F. D'Hondt, J. Collins, P. Wisz, F. W. Van Leeuwen, H. Van Der Poel, P. Schatteman, A. Mottrie, and P. Dell'Oglio, "The safety of urologic robotic surgery depends on the skills of the surgeon," *World Journal of Urology*, vol. 38, no. 6, pp. 1373–1383, 2020.

[33] P. Gupta, J. Schomburg, S. Krishna, O. Adejoro, Q. Wang, B. Marsh, A. Nguyen, J. R. Genere, P. Self, E. Lund, and B. R. Konety, "Development of a classification scheme for examining adverse events associated with medical devices, specifically the DaVinci Surgical System as reported in the FDA MAUDE database," *Journal of Endourology*, vol. 31, no. 1, pp. 27–31, 2017.

[34] C. G. Cao, C. L. MacKenzie, and S. Payandeh, "Task and motion analyses in endoscopic surgery," in *Proceedings of the ASME 1996 International Mechanical Engineering Congress and Exposition*, pp. 583–590, ASME, 1996.

[35] C. E. Reiley and G. D. Hager, "Task versus subtask surgical skill evaluation of robotic minimally invasive surgery," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2009*, pp. 435–442, Springer, 2009.

[36] J. Koskinen, A. Huotarinen, A.-P. Elomaa, B. Zheng, and R. Bednarik, “Movement-level process modeling of microsurgical bimanual and unimanual tasks,” *International Journal of Computer Assisted Radiology and Surgery*, vol. 17, no. 2, pp. 305–314, 2022.

[37] S. S. Vedula, A. O. Malpani, L. Tao, G. Chen, Y. Gao, P. Poddar, N. Ahmidi, C. Paxton, R. Vidal, S. Khudanpur, G. D. Hager, and C. C. G. Chen, “Analysis of the structure of surgical activity for a suturing and knot-tying task,” *PloS ONE*, vol. 11, no. 3, p. e0149174, 2016.

[38] M. Uemura, P. Jannin, M. Yamashita, M. Tomikawa, T. Akahoshi, S. Obata, R. Souzaki, S. Ieiri, and M. Hashizume, “Procedural surgical skill assessment in laparoscopic training environments,” *International Journal of Computer Assisted Radiology and Surgery*, vol. 11, no. 4, pp. 543–552, 2016.

[39] K. C. Brown, K. D. Bhattacharyya, S. Kulason, A. Zia, and A. Jarc, “How to bring surgery to the next level: Interpretable skills assessment in robotic-assisted surgery,” *Visceral Medicine*, vol. 36, no. 6, pp. 463–470, 2020.

[40] S. Khalid and F. Rudzicz, “SurGNN: Explainable visual scene understanding and assessment of surgical skill using graph neural networks,” *arXiv preprint arXiv:2308.13073*, 2023.

[41] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, “Accurate and interpretable evaluation of surgical skills from kinematic data using fully convolutional neural networks,” *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, no. 9, pp. 1611–1617, 2019.

[42] G. Forestier, F. Petitjean, P. Senin, F. Despinoy, A. Huaulmé, H. I. Fawaz, J. Weber, L. Idoumghar, P.-A. Muller, and P. Jannin, “Surgical motion analysis using discriminative interpretable patterns,” *Artificial Intelligence in Medicine*, vol. 91, pp. 3–11, 2018.

[43] Y. Gao, S. S. Vedula, C. E. Reiley, N. Ahmidi, B. Varadarajan, H. C. Lin, L. Tao, L. Zappella, B. Béjar, D. D. Yuh, C. C. G. Chen, R. Vidal, S. Khudanpur, and G. D. Hager, "JHU-ISI gesture and skill assessment working set (JIGSAWS): A surgical activity dataset for human motion modeling," in *MICCAI workshop: M2cai*, vol. 3, 2014.

[44] G. T. Gonzalez, U. Kaur, M. Rahman, V. Venkatesh, N. Sanchez, G. Hager, Y. Xue, R. Voyles, and J. Wachs, "From the dexterous surgical skill to the battlefield—a robotics exploratory study," *Military Medicine*, vol. 186, no. Supplement_1, pp. 288–294, 2021.

[45] I. Rivas-Blanco, C. J. P. Del-Pulgar, A. Mariani, G. Tortora, and A. J. Reina, "A surgical dataset from the da Vinci Research Kit for task automation and recognition," in *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1–6, IEEE, 2023.

[46] K. Hutchinson, I. Reyes, Z. Li, and H. Alemzadeh, "COMPASS: a formal framework and aggregate dataset for generalized surgical procedure modeling," *International Journal of Computer Assisted Radiology and Surgery*, pp. 1–12, 2023. Reproduced with permission from Springer Nature.

[47] T. Nazari, E. Vlieger, M. Dankbaar, J. van Merriënboer, J. Lange, and T. Wiggers, "Creation of a universal language for surgical procedures using the step-by-step framework," *BJS Open*, vol. 2, no. 3, pp. 151–157, 2018.

[48] D. Zhang, Z. Wu, J. Chen, A. Gao, X. Chen, P. Li, Z. Wang, G. Yang, B. P. L. Lo, and G.-Z. Yang, "Automatic microsurgical skill assessment based on cross-domain transfer learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4148–4155, 2020.

[49] C. I. Nwoye, T. Yu, C. Gonzalez, B. Seeliger, P. Mascagni, D. Mutter, J. Marescaux, and N. Padoy, "Rendezvous: Attention mechanisms for the recog-

nition of surgical action triplets in endoscopic videos," *Medical Image Analysis*, vol. 78, p. 102433, 2022.

[50] N. Valderrama, P. Ruiz Puentes, I. Hernández, N. Ayobi, M. Verlyck, J. Santander, J. Caicedo, N. Fernández, and P. Arbeláez, "Towards holistic surgical scene understanding," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2022*, pp. 442–452, Springer, 2022.

[51] A. Huaulmé, D. Sarikaya, K. Le Mut, F. Despinoy, Y. Long, Q. Dou, C.-B. Chng, W. Lin, S. Kondo, L. Bravo-Sánchez, P. Arbeláez, W. Reiter, M. Mitsuishi, K. Harada, and P. Jannin, "Micro-surgical anastomose workflow recognition challenge report," *Computer Methods and Programs in Biomedicine*, vol. 212, p. 106452, 2021.

[52] N. Ahmidi, L. Tao, S. Sefati, Y. Gao, C. Lea, B. B. Haro, L. Zappella, S. Khudanpur, R. Vidal, and G. D. Hager, "A dataset and benchmarks for segmentation and recognition of gestures in robotic surgery," *IEEE Transactions on Biomedical Engineering*, vol. 64, no. 9, pp. 2025–2041, 2017.

[53] G. De Rossi, M. Minelli, S. Roin, F. Falezza, A. Sozzi, F. Ferraguti, F. Setti, M. Bonfè, C. Secchi, and R. Muradore, "A first evaluation of a multi-modal learning system to control surgical assistant robots via action segmentation," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 3, pp. 714–724, 2021.

[54] D. Hu, Y. Gong, B. Hannaford, and E. J. Seibel, "Semi-autonomous simulated brain tumor ablation with RAVENII surgical robot using behavior tree," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3868–3875, IEEE, 2015.

[55] D. Meli and P. Fiorini, "Unsupervised identification of surgical robotic actions from small non-homogeneous datasets," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 8205–8212, 2021.

[56] B. Gibaud, G. Forestier, C. Feldmann, G. Ferrigno, P. Gonçalves, T. Haidegger, C. Julliard, D. Katić, H. Kenngott, L. Maier-Hein, K. März, E. d. Momi, D. Á. Nagy, H. Nakawala, J. Neumann, T. Neumuth, J. R. Balderrama, S. Speidel, M. Wagner, and P. Jannin, "Toward a standard ontology of surgical process models," *International Journal of Computer Assisted Radiology and Surgery*, vol. 13, no. 9, pp. 1397–1408, 2018.

[57] O. R. Meireles, G. Rosman, M. S. Altieri, L. Carin, G. Hager, A. Madani, N. Padoy, C. M. Pugh, P. Sylla, T. M. Ward, D. A. Hashimoto, and the SAGES Video Annotation for AI Working Groups, "SAGES consensus recommendations on an annotation framework for surgical video," *Surgical Endoscopy*, vol. 35, no. 9, pp. 4918–4929, 2021.

[58] N. Madapana, M. M. Rahman, N. Sanchez-Tamayo, M. V. Balakuntala, G. Gonzalez, J. P. Bindu, L. V. Venkatesh, X. Zhang, J. B. Noguera, T. Low, R. M. Voyles, Y. Xue, and J. Wachs, "DESK: A robotic activity dataset for dexterous surgical skills transfer to medical robots," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6928–6934, IEEE, 2019.

[59] G. Menegozzo, D. Dall'Alba, C. Zandonà, and P. Fiorini, "Surgical gesture recognition with time delay neural network based on kinematic data," in *2019 International Symposium on Medical Robotics (ISMR)*, pp. 1–7, IEEE, 2019.

[60] Y. Qin, S. Feyzabadi, M. Allan, J. W. Burdick, and M. Azizian, "daVinciNet: Joint prediction of motion and surgical state in robot-assisted surgery," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2921–2928, IEEE, 2020.

[61] B. Van Amsterdam, I. Funke, E. Edwards, S. Speidel, J. Collins, A. Sridhar, J. Kelly, M. J. Clarkson, and D. Stoyanov, "Gesture recognition in robotic surgery with multimodal attention," *IEEE Transactions on Medical Imaging*, vol. 41, no. 7, pp. 1677–1687, 2022.

[62] J. R. Boehm, N. P. Fey, and A. M. Fey, "Online recognition of bimanual coordination provides important context for movement data in bimanual teleoperated robots," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6248–6255, IEEE, 2021.

[63] S. A. Bowyer, B. L. Davies, and F. R. y Baena, "Active constraints/virtual fixtures: A survey," *IEEE Transactions on Robotics*, vol. 30, no. 1, pp. 138–157, 2013.

[64] G. R. Sutherland, Y. Maddahi, L. S. Gan, S. Lama, and K. Zareinia, "Robotics in the neurosurgical treatment of glioma," *Surgical Neurology International*, vol. 6, no. Suppl 1, pp. S1–S8, 2015.

[65] S. Park, G. Mohammadi, R. Artstein, and L.-P. Morency, "Crowdsourcing micro-level multimedia annotations: The challenges of evaluation and interface," in *Proceedings of the ACM multimedia 2012 workshop on Crowdsourcing for multimedia*, pp. 29–34, 2012.

[66] K. Krippendorff, "Computing Krippendorff's alpha-reliability," 2011.

[67] J. R. Landis and G. G. Koch, "The measurement of observer agreement for categorical data," *Biometrics*, vol. 33, no. 1, pp. 159–174, 1977.

[68] K. L. Gwet, "Benchmarking agreement coefficients," 2014.

[69] J. Hughes, "krippendorffsalpha: An R package for measuring agreement using Krippendorff's alpha coefficient," *arXiv preprint arXiv:2103.12170*, 2021.

[70] C. Lea, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks: A unified approach to action segmentation," in *Computer Vision – ECCV 2016 Workshops*, pp. 47–54, Springer, 2016.

[71] A. Hagberg, P. Swart, and D. S Chult, "Exploring network structure, dynamics, and function using NetworkX," 2008.

[72] H. C. Lin, *Structure in surgical motion.* PhD thesis, The Johns Hopkins University, 2010.

[73] R. DiPietro, N. Ahmidi, A. Malpani, M. Waldram, G. I. Lee, M. R. Lee, S. S. Vedula, and G. D. Hager, "Segmenting and classifying activities in robot-assisted surgery with recurrent neural networks," *International Journal of Computer Assisted Radiology and Surgery*, vol. 14, no. 11, pp. 2005–2020, 2019.

[74] A. Goldbraikh, T. Volk, C. M. Pugh, and S. Laufer, "Using open surgery simulation kinematic data for tool and gesture recognition," *International Journal of Computer Assisted Radiology and Surgery*, vol. 17, no. 6, pp. 965–979, 2022.

[75] Y. Qin, S. A. Pedram, S. Feyzabadi, M. Allan, A. J. McLeod, J. W. Burdick, and M. Azizian, "Temporal segmentation of surgical sub-tasks through deep learning with multiple data sources," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 371–377, IEEE, 2020.

[76] M. Xu, M. Islam, C. Ming Lim, and H. Ren, "Learning domain adaptation with model calibration for surgical report generation in robotic surgery," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 12350–12356, 2021.

[77] V. S. Bawa, G. Singh, F. KapingA, I. Skarga-Bandurova, E. Oleari, A. Leporini, C. Landolfo, P. Zhao, X. Xiang, G. Luo, *et al.*, "The SARAS endoscopic surgeon action detection (ESAD) dataset: Challenges and methods," *arXiv preprint arXiv:2104.03178*, 2021.

[78] L. Li, X. Li, S. Ding, Z. Fang, M. Xu, H. Ren, and S. Yang, "SIRNet: Fine-grained surgical interaction recognition," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4212–4219, 2022.

[79] G. Forestier, F. Lalys, L. Riffaud, B. Trelhu, and P. Jannin, "Classification of surgical processes using dynamic time warping," *Journal of Biomedical Informatics*, vol. 45, no. 2, pp. 255–264, 2012.

[80] R. Ma, E. B. Vanstrum, J. H. Nguyen, A. Chen, J. Chen, and A. J. Hung, "A novel dissection gesture classification to characterize robotic dissection technique for renal hilar dissection," *Journal of Urology*, vol. 205, no. 1, pp. 271–275, 2021.

[81] A. P. Twinanda, S. Shehata, D. Mutter, J. Marescaux, M. De Mathelin, and N. Padoy, "EndoNet: A deep architecture for recognition tasks on laparoscopic videos," *IEEE Transactions on Medical Imaging*, vol. 36, no. 1, pp. 86–97, 2016.

[82] T. M. Ward, D. Hashimoto, Y. Ban, E. R. Witkowski, K. D. Lillemoe, G. Rosman, and O. R. Meireles, "Training with pooled annotations from multiple surgeons has no effect on a deep learning artificial intelligence model's performance," *J Am Coll Surg*, vol. 231, no. 4, p. e203, 2020.

[83] Y. Ban, G. Rosman, J. A. Eckhoff, T. M. Ward, D. A. Hashimoto, T. Kondo, H. Iwaki, O. R. Meireles, and D. Rus, "SUPR-GAN: SUrgical PRediction GAN for event anticipation in laparoscopic and robotic surgery," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5741–5748, 2022.

[84] K. Hutchinson, Z. Li, I. Reyes, and H. Alemzadeh, "Towards surgical context inference and translation to gestures," *arXiv preprint arXiv:2302.14237*, 2023.

[85] K. Hutchinson, I. Reyes, Z. Li, and H. Alemzadeh, "Evaluating the task generalization of temporal convolutional networks for surgical gesture and motion recognition using kinematic data," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 5132–5139, 2023.

[86] M. Ginesi, N. Sansonetto, and P. Fiorini, "Overcoming some drawbacks of dynamic movement primitives," *Robotics and Autonomous Systems*, vol. 144, p. 103844, 2021.

[87] C. Lea, A. Reiter, R. Vidal, and G. D. Hager, "Segmental spatiotemporal CNNs for fine-grained action segmentation," in *Computer Vision – ECCV 2016*, pp. 36–52, Springer, 2016.

[88] R. DiPietro, C. Lea, A. Malpani, N. Ahmidi, S. S. Vedula, G. I. Lee, M. R. Lee, and G. D. Hager, "Recognizing surgical activities with recurrent neural networks," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, pp. 551–558, Springer, 2016.

[89] I. Funke, S. Bodenstedt, F. Oehme, F. von Bechtolsheim, J. Weitz, and S. Speidel, "Using 3D convolutional neural networks to learn spatiotemporal features for automatic surgical gesture recognition in video," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*, pp. 467–475, Springer, 2019.

[90] F. Despinoy, D. Bouget, G. Forestier, C. Penet, N. Zemiti, P. Poignet, and P. Jannin, "Unsupervised trajectory segmentation for surgical gesture recognition in robotic training," *IEEE Transactions on Biomedical Engineering*, vol. 63, no. 6, pp. 1280–1291, 2015.

[91] M. J. Fard, S. Ameri, R. B. Chinnam, and R. D. Ellis, "Soft boundary approach for unsupervised gesture segmentation in robotic-assisted surgery," *IEEE Robotics and Automation Letters*, vol. 2, no. 1, pp. 171–178, 2016.

[92] S. Krishnan, A. Garg, S. Patil, C. Lea, G. Hager, P. Abbeel, and K. Goldberg, "Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning," *The International Journal of Robotics Research*, vol. 36, no. 13-14, pp. 1595–1618, 2017.

[93] J. D. Jones, T. S. Kim, M. Peven, J. Bai, Z. Xiao, Y. Zhang, W. Qiu, A. Yuille, and G. D. Hager, "Zero-shot recognition of complex action sequences," *Advances in Water Resources, Southhampton, UK, Computational Mechanics Publications*, 2019.

[94] L. Clopton, E. Mavroudi, M. Tsakiris, H. Ali, and R. Vidal, "Temporal subspace clustering for unsupervised action segmentation," *CSMR REU*, pp. 1–7, 2017.

[95] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, “Rethinking atrous convolution for semantic image segmentation,” *arXiv preprint arXiv:1706.05587*, 2017.

[96] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[97] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: Common objects in context,” in *Computer Vision – ECCV 2014*, pp. 740–755, Springer, 2014.

[98] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.

[99] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[100] U. Ramer, “An iterative procedure for the polygonal approximation of plane curves,” *Computer Graphics and Image Processing*, vol. 1, no. 3, pp. 244–256, 1972.

[101] D. H. Douglas and T. K. Peucker, “Algorithms for the reduction of the number of points required to represent a digitized line or its caricature,” *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[102] S. Gillies, C. van der Wel, J. Van den Bossche, M. W. Taves, J. Arnott, B. C. Ward, and others, “Shapely,” Oct. 2023.

[103] J. K. H. Andersen, K. L. Schwaner, and T. R. Savarimuthu, “Real-time segmentation of surgical tools and needle using a Mobile-U-Net,” in *2021 20th International Conference on Advanced Robotics (ICAR)*, pp. 148–154, IEEE, 2021.

[104] D. Papp, R. N. Elek, and T. Haidegger, "Surgical tool segmentation on the JIGSAWS dataset for autonomous image-based skill assessment," in *2022 IEEE 10th Jubilee International Conference on Computational Cybernetics and Cyber-Medical Systems (ICCC)*, pp. 000049–000056, IEEE, 2022.

[105] K. Weerasinghe, S. H. R. Roodabeh, K. Hutchinson, and H. Alemzadeh, "Multimodal transformers for real-time surgical activity prediction," *Submitted to the 2024 International Conference on Robotics and Automation (ICRA)*, 2023.

[106] C. Shi, Y. Zheng, and A. M. Fey, "Recognition and prediction of surgical gestures and trajectories using transformer models in robot-assisted surgery," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 8017–8024, IEEE, 2022.

[107] Y. Long, J. Y. Wu, B. Lu, Y. Jin, M. Unberath, Y.-H. Liu, P. A. Heng, and Q. Dou, "Relational graph learning on visual and kinematics embeddings for accurate gesture recognition in robotic surgery," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13346–13353, IEEE, 2021.

[108] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and É. Duchesnay, "Scikit-Learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, no. Oct, pp. 2825–2830, 2011.

[109] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[110] Y. A. Farha and J. Gall, "MS-TCN: Multi-stage temporal convolutional network for action segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3575–3584, 2019.

[111] C. Lea, G. D. Hager, and R. Vidal, "An improved model for segmentation and recognition of fine-grained activities with application to surgical training tasks," in *2015 IEEE Winter Conference on Applications of Computer Vision*, pp. 1123–1129, IEEE, 2015.

[112] Y. Chu, X. M. Yang, H. Li, D. Ai, Y. Ding, J. Fan, H. Song, and J. Yang, "Multi-level feature aggregation network for instrument identification of endoscopic image," *Physics in Medicine & Biology*, vol. 65, no. 16, p. 165004, 2020.

[113] T. A. Alshirbaji, N. A. Jalal, P. D. Docherty, T. Neumuth, and K. Möller, "A deep learning spatial-temporal framework for detecting surgical tools in laparoscopic videos," *Biomedical Signal Processing and Control*, vol. 68, p. 102801, 2021.

[114] Y. Yang, Z. Zhao, P. Shi, and S. Hu, "An efficient one-stage detector for real-time surgical tools detection in robot-assisted surgery," in *Medical Image Understanding and Analysis*, pp. 18–29, Springer, 2021.

[115] T. Kurmann, P. Márquez-Neila, M. Allan, S. Wolf, and R. Sznitman, "Mask then classify: multi-instance segmentation for surgical instruments," *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, no. 7, pp. 1227–1236, 2021.

[116] A. Mohammed, S. Yildirim, I. Farup, M. Pedersen, and Ø. Hovde, "StreoScenNet: surgical stereo robotic scene segmentation," in *Medical Imaging 2019: Image-Guided Procedures, Robotic Interventions, and Modeling*, vol. 10951, p. 109510P, International Society for Optics and Photonics, SPIE, 2019.

[117] R. C. Jackson, R. Yuan, D.-L. Chow, W. S. Newman, and M. C. Çavuşoğlu, "Real-time visual tracking of dynamic surgical suture threads," *IEEE Transactions on Automation Science and Engineering*, vol. 15, no. 3, pp. 1078–1090, 2017.

[118] B. Lu, X. Yu, J. Lai, K. Huang, K. C. Chan, and H. K. Chu, "A learning approach for suture thread detection with feature enhancement and segmentation for 3-D shape reconstruction," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 2, pp. 858–870, 2019.

[119] Y. Hu, Y. Gu, J. Yang, and G.-Z. Yang, "Multi-stage suture detection for robot assisted anastomosis based on deep learning," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4826–4833, IEEE, 2018.

[120] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pp. 234–241, Springer, 2015.

[121] V. Iglovikov and A. Shvets, "TernausNet: U-net with VGG11 encoder pre-trained on ImageNet for image segmentation," *arXiv preprint arXiv:1801.05746*, 2018.

[122] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting encoder representations for efficient semantic segmentation," in *2017 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4, IEEE, 2017.

[123] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 801–818, 2018.

[124] D. Itzkovich, Y. Sharon, A. Jarc, Y. Refaely, and I. Nisky, "Generalization of deep learning gesture classification in robotic-assisted surgical data: from dry lab to clinical-like data," *IEEE Journal of Biomedical and Health Informatics*, vol. 26, no. 3, pp. 1329–1340, 2021.

[125] A. Huaulmé, K. Harada, Q.-M. Nguyen, B. Park, S. Hong, M.-K. Choi, M. Peven, Y. Li, Y. Long, Q. Dou, S. Kumar, S. Lalithkumar, R. Hongliang,

H. Matsuzaki, Y. Ishikawa, Y. Harai, S. Kondo, M. Mitsuishi, and P. Jannin, "PEg TRAnsfer Workflow recognition challenge report: Do multimodal data improve recognition?," *Computer Methods and Programs in Biomedicine*, vol. 236, p. 107561, 2023.

[126] J. Zhang, Y. Nie, Y. Lyu, H. Li, J. Chang, X. Yang, and J. J. Zhang, "Symmetric dilated convolution for surgical gesture recognition," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pp. 409–418, Springer, 2020.

[127] T. Kurmann, P. Marquez Neila, X. Du, P. Fua, D. Stoyanov, S. Wolf, and R. Sznitman, "Simultaneous recognition and pose estimation of instruments in minimally invasive surgery," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2017*, pp. 505–513, Springer, 2017.

[128] N. Rieke, D. J. Tan, F. Tombari, J. P. Vizcaíno, C. A. d. San Filippo, A. Eslami, and N. Navab, "Real-time online adaption for robust instrument tracking and pose estimation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*, vol. 9900, pp. 422–430, Springer, 2016.

[129] N. Joglekar, F. Liu, R. Orosco, and M. Yip, "Suture thread spline reconstruction from endoscopic images for robotic surgery with reliability-driven keypoint detection," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4747–4753, IEEE, 2023.

[130] M. M. Rahman, N. Sanchez-Tamayo, G. Gonzalez, M. Agarwal, V. Aggarwal, R. M. Voyles, Y. Xue, and J. Wachs, "Transferring dexterous surgical skill knowledge between robots for semi-autonomous teleoperation," in *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pp. 1–6, IEEE, 2019.

[131] K. Hutchinson, Z. Li, L. A. Cantrell, N. S. Schenkman, and H. Alemzadeh, "Analysis of executional and procedural errors in dry-lab robotic surgery exper-

iments," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 18, no. 3, p. e2375, 2022.

[132] K. Hutchinson, K. Chen, and H. Alemzadeh, "Towards interpretable motion-level skill assessment in robotic surgery," *arXiv preprint arXiv:2311.05838*, 2023.

[133] P. Joice, G. Hanna, and A. Cuschieri, "Errors enacted during endoscopic surgery - a human reliability analysis," *Applied Ergonomics*, vol. 29, no. 6, pp. 409–414, 1998.

[134] T. R. Eubanks, R. H. Clements, D. Pohl, N. Williams, D. C. Schaad, S. Horgan, and C. Pellegrini, "An objective scoring system for laparoscopic cholecystectomy," *Journal of the American College of Surgeons*, vol. 189, no. 6, pp. 566–574, 1999.

[135] O. Elhage, B. Challacombe, A. Shortland, and P. Dasgupta, "An assessment of the physical impact of complex surgical tasks on surgeon errors and discomfort: a comparison between robot-assisted, laparoscopic and open approaches," *BJU International*, vol. 115, no. 2, pp. 274–281, 2015.

[136] H. Alemzadeh, J. Raman, N. Leveson, Z. Kalbarczyk, and R. K. Iyer, "Adverse events in robotic surgery: a retrospective study of 14 years of fda data," *PloS ONE*, vol. 11, no. 4, pp. 1–20, 2016.

[137] D. Anastasiou, Y. Jin, D. Stoyanov, and E. Mazomenos, "Keep your eye on the best: Contrastive regression transformer for skill assessment in robotic surgery," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1755–1762, 2023.

[138] T. Wang, Y. Wang, and M. Li, "Towards accurate and interpretable surgical skill assessment: A video-based method incorporating recognized surgical gestures and skill levels," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*, pp. 668–678, Springer, 2020.

[139] S. Sukumar and C. G. Rogers, "Robotic partial nephrectomy: surgical technique," *BJU International*, vol. 108, no. 6b, pp. 942–947, 2011.

[140] J. H. Kaouk, A. Khalifeh, S. Hillyer, G.-P. Haber, R. J. Stein, and R. Autorino, "Robot-assisted laparoscopic partial nephrectomy: Step-by-step contemporary technique and surgical outcomes at a single high-volume institution," *European Urology*, vol. 62, no. 3, pp. 553–561, 2012.

[141] E. M. Bonrath, B. Zevin, N. J. Dedy, and T. P. Grantcharov, "Error rating tool to identify and analyse technical errors and events in laparoscopic surgery," *The British Journal of Surgery*, vol. 100, p. 1080–1088, Jul 2013.

[142] K. Moorthy, Y. Munz, A. Dosis, F. Bello, A. Chang, and A. Darzi, "Bimodal assessment of laparoscopic suturing skills," *Surgical Endoscopy And Other Interventional Techniques*, vol. 18, no. 11, pp. 1608–1612, 2004.

[143] M. Shokoohi-Yekta, J. Wang, and E. Keogh, "On the non-trivial generalization of dynamic time warping to the multi-dimensional case," in *Proceedings of the 2015 SIAM International Conference on Data Mining (SDM)*, pp. 289–297, SIAM, 2015.

[144] W. Hu, X. Xiao, Z. Fu, D. Xie, T. Tan, and S. Maybank, "A system for learning statistical motion patterns," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 9, pp. 1450–1464, 2006.

[145] Y. Sharon, A. M. Jarc, T. S. Lendvay, and I. Nisky, "Rate of orientation change as a new metric for robot-assisted and open surgical skill evaluation," *IEEE Transactions on Medical Robotics and Bionics*, vol. 3, no. 2, pp. 414–425, 2021.

[146] C. E. Reiley and G. D. Hager, "Decomposition of robotic surgical tasks: an analysis of subtasks and their correlation to skill," in *M2CAI workshop. MICCAI, London*, vol. 36, pp. 40–43, 2009.

[147] T. Neumuth, G. Strauß, J. Meixensberger, H. U. Lemke, and O. Burgert, "Acquisition of process descriptions from surgical interventions," in *Database and Expert Systems Applications*, pp. 602–611, Springer, 2006.

[148] J. Martin, G. Regehr, R. Reznick, H. Macrae, J. Murnaghan, C. Hutchison, and M. Brown, "Objective structured assessment of technical skill (OSATS) for surgical residents," *British Journal of Surgery*, vol. 84, no. 2, pp. 273–278, 1997.

[149] J. Rosen, J. D. Brown, L. Chang, M. N. Sinanan, and B. Hannaford, "Generalized approach for modeling minimally invasive surgery as a stochastic process using a discrete markov model," *IEEE Transactions on Biomedical Engineering*, vol. 53, no. 3, pp. 399–413, 2006.

[150] M. J. Fard, S. Ameri, R. Darin Ellis, R. B. Chinnam, A. K. Pandya, and M. D. Klein, "Automated robot-assisted surgical skill evaluation: Predictive analytics approach," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 14, no. 1, p. e1850, 2018.

[151] M. E. Tarr, C. Rivard, A. E. Petzel, S. Summers, E. R. Mueller, L. M. Rickey, M. A. Denman, R. Harders, R. Durazo-Arvizu, and K. Kenton, "Robotic objective structured assessment of technical skills: A randomized multicenter dry laboratory training pilot study," *Female Pelvic Medicine & Reconstructive Surgery*, vol. 20, no. 4, p. 228–236, 2014.

[152] M. C. Vassiliou, L. S. Feldman, C. G. Andrew, S. Bergman, K. Leffondré, D. Stanbridge, and G. M. Fried, "A global assessment tool for evaluation of intraoperative laparoscopic skills," *The American Journal of Surgery*, vol. 190, no. 1, pp. 107–113, 2005.

[153] R. Sánchez, O. Rodríguez, J. Rosciano, L. Vegas, V. Bond, A. Rojas, and A. Sanchez-Ismayel, "Robotic surgery training: construct validity of global evaluative assessment of robotic skills (GEARS)," *Journal of Robotic Surgery*, vol. 10, no. 3, pp. 227–231, 2016.

[154] J. Chen, N. Cheng, G. Cacciamani, P. J. Oh, M. Lin-Brande, D. Remulla, I. Gill, and A. Hung, "Objective assessment of robotic surgical technical skill: A systematic review," *Journal of Urology*, vol. 201, no. 3, p. 461–469, 2019.

[155] B. Poursartip, M.-E. LeBel, R. V. Patel, M. D. Naish, and A. L. Trejos, "Analysis of energy-based metrics for laparoscopic skills assessment," *IEEE Transactions on Biomedical Engineering*, vol. 65, no. 7, pp. 1532–1542, 2018.

[156] A. J. Chowriappa, Y. Shi, S. J. Raza, K. Ahmed, A. Stegemann, G. Wilding, J. Kaouk, J. O. Peabody, M. Menon, J. M. Hassett, T. Kesavadas, and K. A. Guru, "Development and validation of a composite scoring system for robot-assisted surgical training—the robotic skills assessment score," *Journal of Surgical Research*, vol. 185, no. 2, pp. 561–569, 2013.

[157] A. J. Hung, J. Chen, A. Jarc, D. Hatcher, H. Djaladat, and I. S. Gill, "Development and validation of objective performance metrics for robot-assisted radical prostatectomy: A pilot study," *The Journal of Urology*, vol. 199, no. 1, pp. 296–304, 2018.

[158] A. Guni, N. Raison, B. Challacombe, S. Khan, P. Dasgupta, and K. Ahmed, "Development of a technical checklist for the assessment of suturing in robotic surgery," *Surgical Endoscopy*, vol. 32, no. 11, pp. 4402–4407, 2018.

[159] E. M. Bonrath, N. J. Dedy, B. Zevin, and T. P. Grantcharov, "Defining technical errors in laparoscopic surgery: a systematic review," *Surgical Endoscopy*, vol. 27, no. 8, pp. 2678–2691, 2013.

[160] R. Ma, A. Ramaswamy, J. Xu, L. Trinh, D. Kiyasseh, T. N. Chu, E. Y. Wong, R. S. Lee, I. Rodriguez, G. DeMeo, A. Desai, M. X. Otiato, S. I. Roberts, J. H. Nguyen, J. Laca, Y. Liu, K. Urbanova, C. Wagner, A. Anandkumar, J. C. Hu, and A. J. Hung, "Surgical gestures as a method to quantify surgical performance and predict patient outcomes," *npj Digital Medicine*, vol. 5, no. 1, p. 187, 2022.

[161] J. Zhang, Y. Nie, Y. Lyu, X. Yang, J. Chang, and J. J. Zhang, "SD-Net: joint surgical gesture recognition and skill assessment," *International Journal of Computer Assisted Radiology and Surgery*, vol. 16, no. 10, pp. 1675–1682, 2021.

[162] N. Ahmidi, Y. Gao, B. Béjar, S. S. Vedula, S. Khudanpur, R. Vidal, and G. D. Hager, "String motif-based description of tool motion for detecting skill and gestures in robotic surgery," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2013*, pp. 26–33, Springer, 2013.

[163] S. Zhao, Y. Liu, Q. Wang, D. Sun, R. Liu, and S. K. Zhou, "MURPHY: Relations matter in surgical workflow analysis," *arXiv preprint arXiv:2212.12719*, 2022.

[164] L. Chang, R. Satava, C. Pellegrini, and M. Sinanan, "Robotic surgery: identifying the learning curve through objective measurement of skill," *Surgical Endoscopy And Other Interventional Techniques*, vol. 17, no. 11, pp. 1744–1748, 2003.

[165] S. I. Roberts, S. Y. Cen, J. H. Nguyen, L. C. Perez, L. G. Medina, R. Ma, S. Marshall, R. Kocielnik, A. Anandkumar, and A. J. Hung, "The relationship between technical skills, cognitive workload, and errors during robotic surgical exercises," *Journal of Endourology*, vol. 36, no. 5, pp. 712–720, 2022.

[166] S. Khalid, V. Palter, T. Grantcharov, and F. Rudzicz, "OR Vision: Objective, explainable assessment of surgical skill with deep learning," 2022.

[167] S. Yibulayimu, Y. Wang, Y. Liu, Z. Sun, Y. Wang, H. Jiang, and F. Li, "An explainable machine learning method for assessing surgical skill in liposuction surgery," *International Journal of Computer Assisted Radiology and Surgery*, vol. 17, no. 12, pp. 2325–2336, 2022.

[168] A. Soleymani, X. Li, and M. Tavakoli, "A domain-adapted machine learning approach for visual evaluation and interpretation of robot-assisted surgery skills," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8202–8208, 2022.

[169] A. Hendricks, M. Panoff, K. Xiao, and C. Bobda, "Exploring the limitations of the JIGSAWS dataset for robotic surgery," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2023.

[170] R. H. Taylor, A. Menciassi, G. Fichtinger, P. Fiorini, and P. Dario, "Medical robotics and computer-integrated surgery," in *Springer Handbook of Robotics*, pp. 1657–1684, Springer, 2016.

[171] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1961–1970, 2016.

[172] S. Stein and S. J. McKenna, "Combining embedded accelerometers with computer vision for recognizing food preparation activities," in *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, pp. 729–738, Association for Computing Machinery, 2013.

[173] A. Fathi, X. Ren, and J. M. Rehg, "Learning to recognize objects in egocentric activities," in *CVPR 2011*, pp. 3281–3288, IEEE, 2011.

[174] F. Cepolina and R. P. Razzoli, "An introductory review of robotically assisted surgical systems," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 18, no. 4, p. e2409, 2022.

[175] B. Hannaford, J. Rosen, D. W. Friedman, H. King, P. Roan, L. Cheng, D. Glozman, J. Ma, S. N. Kosari, and L. White, "Raven-II: An open platform for surgical robotics research," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 954–959, 2012.

[176] X. Li, H. Alemzadeh, D. Chen, Z. Kalbarczyk, R. K. Iyer, and T. Kesavadas, "A hardware-in-the-loop simulator for safety training in robotic surgery," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5291–5296, IEEE, 2016.

[177] Y.-H. Su, A. Munawar, A. Deguet, A. Lewis, K. Lindgren, Y. Li, R. H. Taylor, G. S. Fischer, B. Hannaford, and P. Kazanzides, "Collaborative robotics toolkit (CRTK): Open software framework for surgical robotics research," in *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pp. 48–55, 2020.

[178] Regional West, "DA VINCI ROBOTIC SURGERY SYSTEM," 2015.

[179] NDI, "Tool navigation with electromagnetic tracking technology," 2022.

[180] O. Kramer, "Scikit-Learn," *Machine Learning for Evolution Strategies*, pp. 45–53, 2016.

[181] K. Hutchinson, M. S. Yasar, H. Bhatia, and H. Alemzadeh, "A reactive autonomous camera system for the RAVEN II surgical robot," in *2020 International Symposium on Medical Robotics (ISMR)*, pp. 195–201, IEEE, 2020.

[182] G.-Z. Yang, J. Cambias, K. Cleary, E. Daimler, J. Drake, P. E. Dupont, N. Hata, P. Kazanzides, S. Martel, R. V. Patel, *et al.*, "Medical robotics—regulatory, ethical, and legal considerations for increasing levels of autonomy," *Science Robotics*, vol. 2, no. 4, p. eaam8638, 2017.

[183] A. Pandya, L. Reisner, B. King, N. Lucas, A. Composto, M. Klein, and R. Ellis, "A review of camera viewpoint automation in robotic and laparoscopic surgery," *Robotics*, vol. 3, no. 3, pp. 310–329, 2014.

[184] H. Alemzadeh, D. Chen, A. Lewis, Z. Kalbarczyk, J. Raman, N. Leveson, and R. Iyer, "Systems-theoretic safety assessment of robotic telesurgical systems," in *Computer Safety, Reliability, and Security*, pp. 213–227, Springer, 2015.

[185] A. V. Mudunuri, *Autonomous camera control system for surgical robots.* PhD thesis, Wayne State University, 2010.

[186] K. Omote, H. Feussner, A. Ungeheuer, K. Arbter, G.-Q. Wei, J. R. Siewert, and G. Hirzinger, "Self-guided robotic camera control for laparoscopic surgery compared with human camera control," *The American Journal of Surgery*, vol. 177, no. 4, pp. 321–324, 1999.

[187] G.-Q. Wei, K. Arbter, and G. Hirzinger, "Real-time visual servoing for laparoscopic surgery. controlling robot motion with color image segmentation," *IEEE Engineering in Medicine and Biology Magazine*, vol. 16, no. 1, pp. 40–45, 1997.

[188] L. Yu, H. Li, L. Zhao, S. Ren, and Q. Gu, "Automatic guidance of laparoscope based on the region of interest for robot assisted laparoscopic surgery," *Computer Assisted Surgery*, vol. 21, no. sup1, pp. 17–21, 2016.

[189] B. Yang, W. Chen, Z. Wang, Y. Lu, J. Mao, H. Wang, and Y.-H. Liu, "Adaptive fov control of laparoscopes with programmable composed constraints," *IEEE Transactions on Medical Robotics and Bionics*, vol. 1, no. 4, pp. 206–217, 2019.

[190] B. W. King, L. A. Reisner, A. K. Pandya, A. M. Composto, R. D. Ellis, and M. D. Klein, "Towards an autonomous robot for camera control during laparoscopic surgery," *Journal of Laparoendoscopic & Advanced Surgical Techniques*, vol. 23, no. 12, pp. 1027–1030, 2013.

[191] S. Eslamian, L. A. Reisner, B. W. King, and A. K. Pandya, "An autonomous camera system using the da Vinci Research Kit," in *Proc. Int. Conf. Intell. Robots Syst.(IROS)*, p. 2, 2017.

[192] S. Eslamian, L. A. Reisner, and A. K. Pandya, "Development and evaluation of an autonomous camera control algorithm on the da Vinci Surgical System," *The International Journal of Medical Robotics and Computer Assisted Surgery*, vol. 16, no. 2, p. e2036, 2019.

[193] P. Kazanzides, Z. Chen, A. Deguet, G. S. Fischer, R. H. Taylor, and S. P. DiMaio, "An open-source research kit for the da Vinci® Surgical System," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6434–6439, IEEE, 2014.

[194] Stereolabs, "ZED Mini Stereo Camera."

[195] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2961–2969, 2017.

[196] W. Abdulla, "Mask R-CNN for object detection and instance segmentation on keras and tensorflow," 2017.

[197] M. Margolis, "Servo library."

[198] A. Dutta and A. Zisserman, "VGG image annotator (VIA)," 2016.

[199] F. Chollet *et al.*, "Keras: The Python deep learning library," *Astrophysics source code library*, pp. ascl–1806, 2018.

[200] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: A system for Large-Scale machine learning," in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283, 2016.