8-2024

Computer Vision Algorithms for Assessment of Surgical Suturing Skill Using Hand and Needle Motion

Jianxin Gao
jianxig@g.clemson.edu

Computer vision algorithms for assessment of surgical suturing skill using hand and needle motion

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Electrical Engineering

---

by
Jianxin Gao
August 2024

Accepted by:
Dr. Richard E. Groff, Committee Chair
Dr. Ravikiran Singapogu, Committee Co-Chair
Dr. Xiaolong Ma
Dr. Ian D. Walker

Abstract

Surgical suturing skill assessment is a crucial part of surgical education. Vascular surgery educators have developed a simulation-based examination called Fundamentals of Vascular Surgery, which includes a clock-face model for assessing open surgical suturing skills. The clock-face model, however, requires the valuable time of expert surgeons to determine examinees' skills. Moreover, expert surgeons have different judgments for appropriate sutures, which leads to inconsistent grading. These limitations motivate us to use sensors to measure examinees' needle motions and hand motions during the clock-face suturing exercises, and then use the measurements for objective suturing skill assessment.

To assess suturing skills based on needle motions, computer vision is used to track the needle trajectory. Skill assessment metrics are then calculated based on the estimated needle trajectory. Improving on a prior design, a more robust suturing simulator was designed and constructed. The new simulator was used to collect data from 97 participants, with different years of surgical experience, who were separated into novice, intermediate, and expert groups. Statistical analysis shows that all the image-based metrics have significantly different means between the novice and the remaining two groups. The groups are even more distinguished when analyzing suturing at the depth condition, for which six of the seven metrics have significantly different means between the intermediate and expert groups. Further, the metric classification performance is examined by ROC curves. The results show that the median metric value for a suturing exercise has higher classification accuracy than the metric value for a suture.

To assess suturing skills based on hand motion, a deep-learning program is developed to analyze hand-motion videos, and then estimate hand rotation at the roll axis. The program includes a hand detection algorithm and a hand roll estimation algorithm. The roll motion estimate is then used to calculate hand-roll metrics for suturing skill assessment. The hand detection algorithm receives

videos and then localizes participants' dominant hand in video frames. Results show that the hand detection algorithm precisely localizes participants' dominant hand in videos with different brightness and backgrounds. The hand roll estimation algorithm receives cropped hand images and then outputs the corresponding hand roll angle or roll velocity. Results show that the hand roll estimation algorithm has consistent performance at different locations. To examine if the algorithm output can be used for surgical suturing skill assessment, the algorithm output and the IMU measurements are used to calculate hand-roll metrics that capture participants' hand roll during the clock-face suturing task. Results show that the estimated hand roll angle has similar statistics as the IMU measurement when calculating the hand-roll metrics. Specifically, 4 out of the 5 metrics have significantly different means between the novice group and the remaining two groups. Also, 4 out of the 5 metrics have significant mean differences between the resident surgeons' and attending surgeons' groups.

Dedication

This dissertation is dedicated to my beloved parents Yonghong and Hongying, and my lovely brother Jianwen for their unconditional support throughout my life. Also, to my grandparents and friends, who support me during my Ph.D. journey.

Acknowledgments

I would like to express my sincere gratitude and deep appreciation to my advisor, Dr. Richard E. Groff, for his guidance, patience, cooperation, and support during the development of this dissertation. Dr. Groff has provided me with excellent supervision, motivation, enthusiasm, and invaluable comments during the work on this dissertation.

I would like to thank Dr. Ravikiran B. Singapogu for his continued support and motivation throughout this work. My sincerest gratitude and deep appreciation also extend to Dr. Xiaolong Ma and Dr. Ian D. Walker for serving on my committee.

I would also like to thank our medical collaborators for their invaluable insight and feedback on this research. I would also like to thank Dr. Joe Bible for his invaluable feedback on statistical analysis. I would also like to thank my project mates Irfan Kil, Amir Mehdi Shayan, Simar P. Singh, Zhanhe Liu, and Ziyang Zhang for their contributions to this research.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Surgical skill is closely related to clinical outcomes. Inadequate surgical skill increases the possibility of post-operative complications [8]. Worse, poor surgical performance can lead to medical errors, which is the third leading cause of death in the US [58]. Therefore, it is crucial to have a process to examine trainees' technical skills and ensure them being competent surgeons.

Our lab focuses on designing simulators for surgical skill assessment, especially suturing skills for open vascular surgery. Like general surgery education, vascular surgery educators encounter the challenge of reduced resident surgeons' working hours, which leads to concerns about resident training [65]. Also, the introduction of endovascular surgery causes vascular training to spend less time teaching open surgical techniques, but open vascular surgery is still required in complex situations [77, 59].

These challenges motivate vascular surgery educators to develop simulation-based training. Inspired by Fundamentals of Endoscopic Surgery (FES) [91], Fundamentals of Laparoscopic Surgery (FLS) [24], and the corresponding simulators [91, 16], vascular surgery educators developed a simulation-based examination named Fundamentals of Vascular Surgery (FVS) [77] for open surgical training. FVS includes three simulation-based vascular skill assessment models, namely the end-to-side anastomosis model, the patch angioplasty model, and the clock-face suturing model. Specifically, the clock-face suturing model aims to assess trainees' radial suturing ability [77], as suturing is an essential surgical technique [78] and can be used to identify trainees' capabilities [48]. The simulation-based models in FVS, however, require the valuable time of expert surgeons to evaluate participants' skills [77, 73]. Also, studies indicate that FVS should include rater training

to guarantee grading consistency [73]. These limitations suggest introducing sensors to the FVS models for objective skill assessment.

The research community is pursuing sensor-embedded simulators for open surgical training, in order to relieve examiners from time-consuming evaluations [108, 62] and to provide objective judgments [20]. Research shows that surgical skills learned on a bench model such as anastomosis, are transferable to realistic settings [3, 79]. Several simulators have been developed to evaluate open surgical skills [11, 6, 10, 21, 81]. The simulators with electromagnetic trackers are able to track hand motion and record task duration [11, 6, 10]. Some simulators are equipped with a force sensor under the suturing platform, which allows measuring the subjects' force applied on the platform during practice [21, 81]. These simulators are proven to be promising in distinguishing participants' skill levels in open surgery.

Suturing skills can be evaluated by various metrics. A popular choice is the motion-based metrics that measure hand or tool movements [11, 6, 10, 21, 104, 90, 75]. Also, force-based [21, 88, 69, 35] and image-based metrics [25, 39, 38, 64, 51, 49] are widely used for evaluating suturing skills. Recent studies propose using deep-learning models to analyze suturing videos but without specifying the metrics for skill assessment [26, 47, 57, 84, 98]. Although those deep-learning models demonstrate high accuracy for classifying participants' skill levels, they have not been widely used in training due to the lack of physical interpretations of the model outputs. Also, our literature review shows that image-based metrics are more popular in minimally invasive surgery [39, 38, 64] than in open surgery.

This work proposes using two video-analysis techniques to objectively assess suturing skills used in open surgery, especially for the clock-face suturing model in FVS. Specifically, in the first half of the dissertation, suturing skills are evaluated by image-based metrics, which are obtained from analyzing videos with needle motion. In the second half of the dissertation, suturing skills are evaluated by hand-roll metrics, which are obtained from analyzing videos with hand motion. Notice that prior research has used needle and hand motion for suturing skill evaluation, but they have some limitations. Kil et al. [51] use image-based metrics to analyze needle motion during a simulated surgical task, but their dataset is comparatively small, so the results only suggest the image-based metrics can distinguish the populations with different experience levels (e.g., resident surgeons and attending surgeons). Shayan et al. [76] show that suturing skills can be evaluated via hand-roll metrics measuring hand rotation at the roll axis, but participants must wear sensors for

the purpose of data collection, which might interfere with their surgical performance. Thus, this work aims to answer the following questions:

- Do populations with different experience levels perform differently in terms of image-based metrics?
- Can a participant's skill level be classified using image-based metrics?
- If hand rotation is estimated via video analysis, do populations with different experience levels perform differently in terms of hand-roll metrics?

1.1 Skill evaluation via analyzing videos with needle motion

Several research groups have proposed using image-based metrics to distinguish participants' skill levels [25, 51, 49, 39, 64, 13]. Frischknecht et al. [25] assess participants' suturing skills by analyzing the images with their suturing end product. Specifically, their results show that stitch orientation was significantly different between surgical faculty and medical students. Kil et al. [51, 49] evaluate suturing skills by analyzing the suturing end products and the needle movement during suturing. Islam et al. [39] assess participants' skills by analyzing videos with tool movement. By calculating the optical flow of the tool, they found that motion smoothness and movement efficiency can distinguish resident surgeons and medical students. Similarly, Fernando et al. [64] distinguish experienced and non-experienced surgeons by classifying the tool motion characteristics obtained from a video-tracking system. Specifically, they found that the two groups have significantly different path lengths of the instrument tip. Recently, Deepika et al. [13] assess suturing skills by a deep-learning model. Thanks to the robustness of the pre-trained model, the proposed method is capable of detecting tool motion in the videos recorded in the operating room. Those studies, especially the stitch orientation, inspire us to design new image-based metrics for suturing skill assessment.

The first half of the report proposes using video analysis to track the needle and the thread motion in a suturing simulator named SutureCoach [76]. The detected needle trajectories are then evaluated by image-based metrics for suturing skill evaluation. Compared with the suturing simulator in [50, 49, 51], SutureCoach enhances portability by simplifying the table structure and the vertical constraints. Like other video-based simulators [37, 39, 38], the object tracking system on

SutureCoach avoids the line-of-sight issues [67] by detecting the targets (i.e., the needle and the thread) in a controlled environment (i.e., the membrane housing shown in Fig. 2.3(b) and (c)). Also, the reliability of the object tracking system is enhanced by re-designing the membrane housing, the lighting sources, and the software for data processing. These improvements enable us to collect a dataset larger than the prior work [51]. The dataset, including participants with different experience levels, is then evaluated by the image-based metrics in [51] and the new image-based metrics based on the refined version of suture phase segmentation scheme [45, 49]. Furthermore, statistical methods are used to analyze metric values from the groups with different experience levels. The metric classification performance is also evaluated by receiver operating characteristic (ROC) curves [22]. The result shows that the image-based metrics can evaluate participants' experience levels from novices to resident surgeons to attending surgeons.

1.2 Skill evaluation via analyzing videos with hand motion

Hand motion, including rotational and translational motion, are effective measurements for distinguishing participants' suturing skills in open surgery [11, 6, 10, 21, 76]. Dubrowski et al. [21] measure participants' hand roll angle by an Electromagnetic (EM) tracker. Similarly, Shayan et al. [76] measure hand reversal by analyzing the roll velocity from inertial measurement units (IMU). In comparison, several research groups [11, 6, 10] use EM trackers to measure hand translational motion. These studies show promising results of suturing skill evaluation, but they require participants to wear specific sensors during data collection, which might interfere with their suturing performance.

To avoid the interference of sensors, Goldbraikh et al. [29] propose measuring hand translational motion by analyzing a video stream from a webcam, so participants are not required to wear sensors during the suturing tasks. This work motivates us to create a video-processing algorithm for hand rotation estimation, especially the rotation at the roll axis in 3D rotation.

Estimating objects' 3D rotation and 3D translation, also called 6D object pose estimation, is a fundamental problem in computer vision. Recent studies show that the 6D object pose can be estimated by a single RGB image [46, 101, 95, 96]. A popular process of 6D pose estimation is to crop the targeted object from the original image [46, 95, 96]. After extracting features from the cropped image via the convolutional neural network (CNN), the 6D object pose is estimated by the fully connected layers [101, 95, 96]. Also, recent research [96] suggests parameterizing the

3D rotation by continuous representation rather than the quaternion used in [101, 95]. Compared with the 6D object pose estimation, our hand roll estimation task is simpler, as only the roll axis in 3D rotation is considered. The work in 6D object pose estimation, however, provides us with a guideline for designing the hand roll estimation algorithms.

Estimating hand rotation at a single axis, especially the in-plane rotation [89], has been adopted by hand detection [14] and hand pose estimation tasks [53], as research shows that transforming objects to a canonical pose can improve the performance of algorithms [42]. Deng et al. [14] introduce a rotation estimation network to the hand detection algorithm, so the object orientation is calibrated to the upward direction before performing hand detection. Similarly, Kong et al. [53] use a rotation net to make the hands pointing upward before estimating the hand pose. In comparison, our task is more challenging because we aim to estimate hand roll, which corresponds to out-of-plane rotation [89] of the hand. Specifically, rotation in [14, 53] refers to the rotation about the yaw axis in our IMU configuration, while rotation of hand roll refers to the roll axis in our IMU configuration.

Notice that our task differs from the hand pose estimation, which aims to estimate the 2D or 3D coordinates of the hand joints [18]. Studies show that the locations of hand joints can be obtained by analyzing depth images [28, 12] or RGB images [53, 109], but the latter is more challenging due to the lack of depth information [18]. It is worth noting that Smedt [12] further estimates the hand gestures by analyzing a sequence of estimated hand poses, which motivates us to estimate roll velocity based on a sequence of video frames.

The second half of the report proposes using deep-learning algorithms to analyze videos and estimate the roll angle and roll velocity of participants' dominant hands. The estimated hand roll will be further processed by the motion-based metrics in [76] for suturing skill evaluation. The deep-learning algorithms involve a hand detection algorithm and a hand roll estimation algorithm. The hand detection algorithm, fine-tuned based on YOLOv7 [94], is for localizing participants' dominant hand in video frames. The cropped hand images are then processed by the hand roll estimation algorithm to estimate the corresponding roll angle and roll velocity. Since videos are constructed by a sequence of images, the hand roll estimation algorithms include convolutional layers and long short-term memory (LSTM) layers. Convolutional layers are the major component of the convolutional neural networks (CNN) [63], which are primarily used in image analysis tasks such as image classification [80, 33]. LSTM is a special implementation of the recurrent neural network (RNN). Since LSTM supports remembering temporal information but overcomes RNN's

long-term dependency problem [105], it is widely used in sequence analysis tasks such as trajectory prediction [1, 2]. The combination of CNN and LSTM is known as the convolutional recurrent neural network (CRNN) [97]. Since CRNN takes advantage of CNN for local feature extraction and LSTM for temporal summarization of the extracted features, it shows great success in video classification tasks [106, 99]. Notice that our roll estimation algorithm is inspired by the algorithm structure in [106]. To train the CNN and CRNN for hand roll estimation, SutureCoach is used to collect the videos and the IMU data. To accelerate the training process, the transfer learning technique [15, 86] is adopted.

Chapter 2

Suturing simulator

2.1 The clock-face suturing model in FVS

The FVS examination includes the clock-face suturing model to assess trainees' needle handling and suturing accuracy [87]. The model consists of a clear plastic tube (see Fig. 2.1(a)) and a patch material with a clock-face pattern (see Fig. 2.1(b)). The plastic tube is for simulating the depth constraint of the abdominal cavity [59]. During the suturing exercise, the patch material is stretched with four clamps inside the plastic tube [87].

To complete the clock-face suturing exercise, a participant starts inserting the needle at the intersection of the radius and the inner circle (see Fig. 2.1(c)). The ideal needle pull-out location is at the intersection of the outer circle and the same radius. The exercise continues by applying the same operation on the next radius in a clockwise direction. The exercise is considered to be finished after all 12 radii have been sutured.

For the clock-face suturing model, a participant's suturing skill is evaluated by expert surgeons during the exercise [77, 59], which takes expert surgeons' valuable time. Also, the study in [87] demonstrates significant intra- and inter-assessor variability when evaluating participants' performance on the clock-face suturing model. These limitations motivate us to design a suturing simulator to objectively evaluate the suturing skills.

(a) (b) (c)

Figure 2.1: The clock-face suturing model in FVS: (a) The plastic tube for simulating the depth constraint; (b) The patch material with clock-face pattern; (c) An illustration of the suturing exercise. The figures are from [59].

2.2 The original suturing simulator

The suturing simulators were developed at Clemson University over a period of years [44, 43, 60]. In 2019, Irfan et al. [49] integrated the previous development and created a simulator that measures suturing skills on the clock-face model in FVS. The simulator was then used to collect a small dataset for studying objective suturing skill evaluation [50, 49, 51]. Since Irfan's suturing simulator [49] motivates the development of SutureCoach, the former is called the original simulator hereafter.

The original simulator includes a membrane housing (see Fig. 2.2(b) and (c)) to hold the suturing material. Compared with the clock-face model in FVS (see Fig. 2.1(b)), the suturing pattern (see Fig. 2.2(b)) on the simulator does not specify the ideal needle pull-out location. The modification is to avoid novices repeatedly adjusting the needle at the pull-out location. To complete the suturing exercise, a participant starts inserting the needle through the entry point, which is marked on the suturing material (see Fig. 2.2(b)). The ideal needle pull-out location is on the same radius, such that the incision line is in the middle between the entry point and the pull-out location. The exercise continues by applying the same operation on the next radius in a counter-clockwise direction. After all 12 sutures have been finished, the suturing exercise is complete.

The suturing simulator includes a height-adjustable table (see Fig. 2.2(a)) and two cameras (see Fig. 2.2(a) and (c)). The height-adjustable table allows participants to choose a comfortable height for the suturing exercise. To simulate the depth constraint, the table includes a stepper motor (see Fig. 2.2(a)) to raise up/bring down the plastic tube, which surrounds the membrane housing

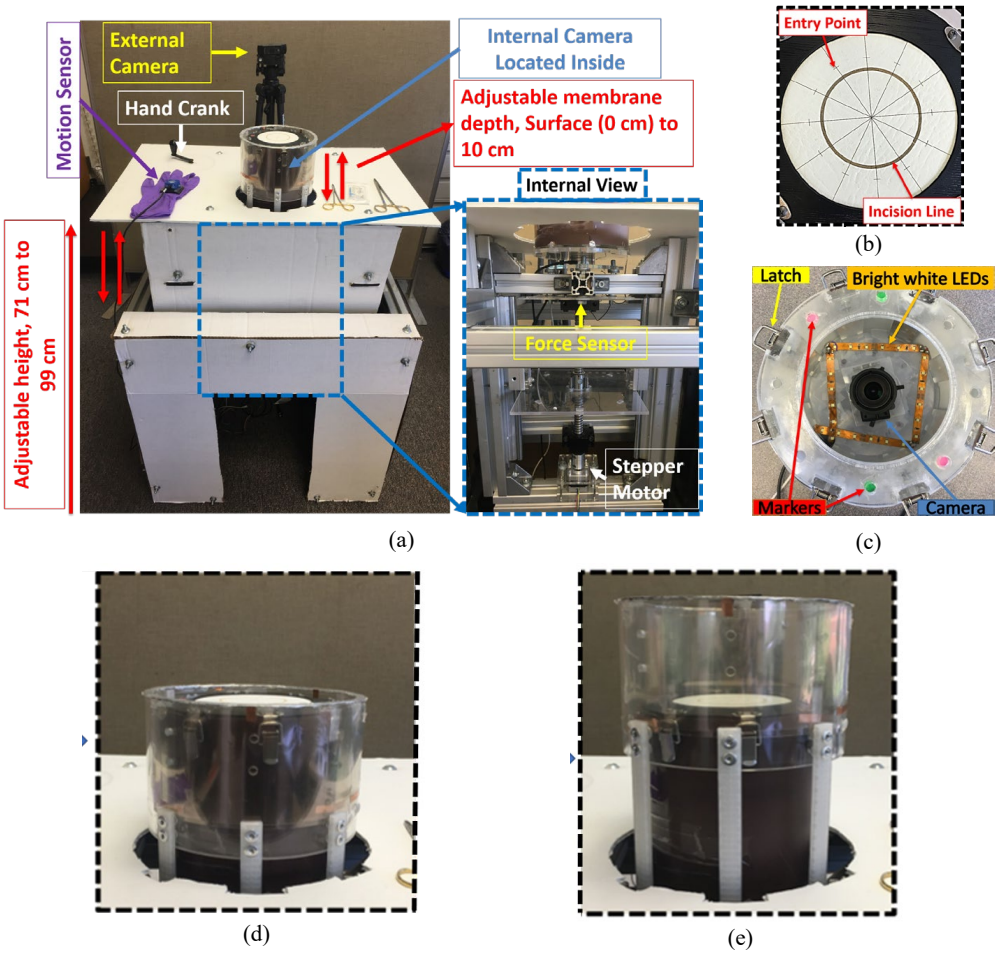


Figure 2.2: The original suturing simulator: (a) Front view; (b) Top view of the membrane housing; (c) The internal structure of the membrane housing; (d) Surface condition; (e) Depth condition. The sub-figures are from [49].

(see Fig. 2.2(d) and (e)). For the two cameras, the camera on the tripod (see Fig. 2.2(a)) is for recording participants' hand motion, while the camera in the membrane housing (see Fig. 2.2(c)) is for recording needle motion during the suturing exercise.

The original simulator has some limitations. First, its height-adjustable table is light, so it is surrounded by an aluminum barrier (see Fig. 2.2(a)) to avoid participants leaning on the table. This extra component increases the cost of transportation. Second, the height-adjustable mechanism for the depth constraint is difficult to maintain. Finally, the camera in Fig. 2.2(a) lacks a stable light source, so it is sensitive to ambient light. These limitations hinder transporting the simulator to different locations for data collection, so the simulator was only used to collect a small dataset

with 15 participants (6 attending surgeons, 8 resident surgeons, and 1 medical student).

2.3 The improved suturing simulator

2.3.1 Hardware

The improved simulator, named SutureCoach, is designed for collecting a large dataset using the clock-face model in FVS. To maintain the consistency of the study, SutureCoach uses the same suturing pattern (see Fig. 2.3(b)) as the original simulator. Since the overview of SutureCoach has been described in [76], this work focuses on the improvement in the table frame, the vertical constraints, and the object tracking system. SutureCoach is based on a height-adjustable table (see Fig. 2.3(a)) heavier than that in the original simulator (see Fig. 2.2(a)), allowing us to remove the peripheral aluminum barrier used in its predecessor. The vertical constraints (see Fig. 2.3(d) and (e)) on SutureCoach simulate two suturing conditions: surface and depth. Those vertical constraints are constructed by 3D printed components and clear PETG sheets but without the automatic height-adjustable mechanism, which simplifies the design in the original simulator. Also, SutureCoach includes a 3D-printed membrane housing (See Fig. 2.3(b) and (c)) and two Intel RealSense D435 cameras. The new membrane housing enhances the structure by replacing the acrylic components with 3D-printed components. The camera in the membrane housing (i.e., Camera 1 in Fig. 2.3(c)) records RGB videos with 640×480 resolution at 60 frames per second (fps). The LED strip and the light diffusing panel guarantee a stable light source for the videos (See Fig. 2.3(c)). Comparatively, the camera on the mount (i.e., Camera 2 in Fig. 2.3(a)) records both RGB and depth videos with 848×480 resolution at 30 fps. It is equipped with a ring light to provide consistent lighting (See Fig. 2.3(a)). Those modifications improve the portability and reliability of the simulator, which guarantees collecting consistent data in different environments.

SutureCoach also improves the configuration of sensors and the ability of reproduction. Compared with the original suturing simulator, SutureCoach improves the configuration of the force sensor and Camera 1. By aligning the axes of the two sensors, the force readings can be directly interpreted by the videos from the camera. This configuration simplifies the force data analysis. Also, the assembly process of SutureCoach has been documented, so the simulator can be duplicated easily. For instance, two SutureCoach were used for data collection in the 2022 Society for Clinical Vascular Surgery (SCVS) Conference.

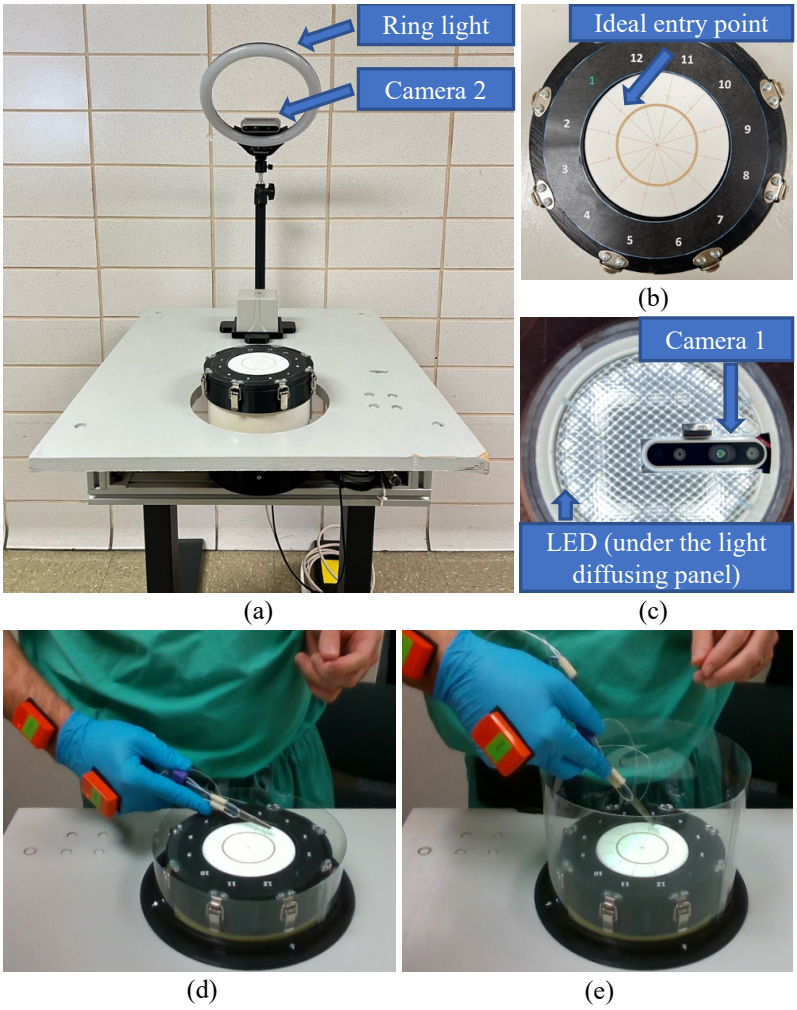


Figure 2.3: The suturing simulator: (a) Front view; (b) Top view of the membrane housing; (c) The internal structure of the membrane housing; (d) Surface condition; (e) Depth condition.

Besides the two cameras, SutureCoach also has a force sensor, two electromagnetic (EM) trackers, and two IMU. the force sensor (ATI MINI 40, ATI Industrial Automation Inc.), operating at 1,000 Hz, is under the membrane housing. The EM sensors (Ascension trakSTAR Model 180, Northern Digital Inc.), operating at 100 Hz, are mounted on the shafts of the needle driver. The IMU sensors (Xsens MTw Awinda wireless motion tracker, Xsens Technologies BV), operating at 120 Hz, are stuck to the participant's hand and wrist during data collection.

2.3.2 Software

SutureCoach processes sensor data by a desktop computer with an Intel Xeon CPU and an NVIDIA Quadro RTX 4000 GPU. Its data processing stages includes (i) the data-collection stage and (ii) the data-analysis stage. In stage (i), two C++ programs (see Fig. 2.4), written on MS Visual Studio 2017 configured with OpenCV CUDA module, are used for data collection and image processing.

The data-collection program is used to synchronize and record data from multiple sensors, namely two cameras, two IMUs, two EM position sensors, and a force sensor. Data synchronization allows us to concurrently study multiple signals and identify erroneous movement [19]. Meanwhile, the program sends the video stream from Camera 1 to the image-processing program. Compared with the code in the original simulator, the data-collection program uses an improved data management scheme and has higher fault tolerance. Specifically, the data-collection program automatically assigns each participant a unique hash ID, which is then used to organize the data. Furthermore, the data-collection program saves the video stream into a buffer before sending it to the image-processing program. This configuration avoids the data logging process being interfered with the image-processing program, even when the image-processing program fails unexpectedly.

The image-processing program is used to detect the needle and thread motion in the video stream from Camera 1. Unlike the code in the original simulator, the image-processing program is not for displaying metric values after each suture but for concurrently calculating and logging metric values. To achieve real-time processing, multi-threading is used in the image-processing program. Specifically, Thread 1 is for detecting needle and thread in the video stream, which utilizes the OpenCV module running on GPU. In comparison, Thread 2 is for calculating metrics and extracting features such as the needle tip, which requires the OpenCV module running on CPU. These results are then saved for later analysis. Notice that Thread 1 and Thread 2 communicate via a buffer (i.e., shared memory), as the two threads run at different speeds.

The data-collection program and the image-processing program are communicated via TCP/IP protocol, which is responsible for transferring the video stream and the control signals. The control signals are represented by the following keyboard inputs:

- **w** (or **w**rite) starts the data-collection and the image-processing process.
- **m** (or **m**ove) moves the region of interest to the next suture. It is used to continue the two

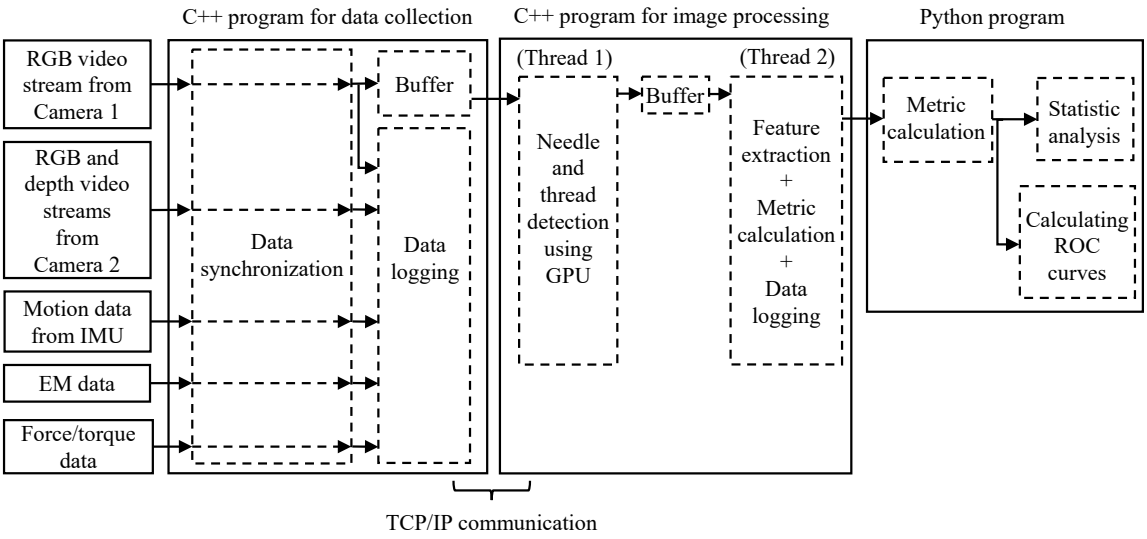


Figure 2.4: Software for SutureCoach

programs when a specific suture is incomplete.

- **s** (or **s**top) stops the two programs from processing the current suturing exercise, but the TCP/IP communication is still available, which allows the two programs to wait for the start of the next exercise.

- **q** (or **q**uit) quits the two programs and the TCP/IP communication.

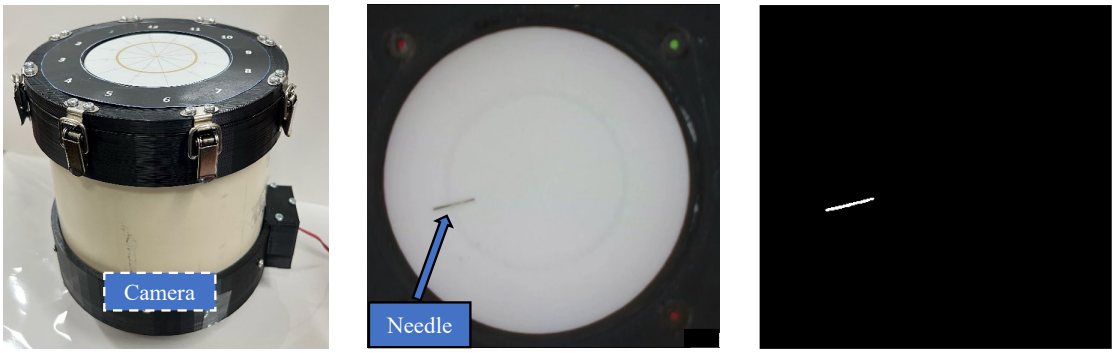
Since the control signals are shared via the TCP/IP communication, the data-collection and the image-processing program are controlled simultaneously, which makes operating SutureCoach easier than operating the original simulator.

In stage (ii), a custom Python program (shown on the right in Fig. 2.4) is used to calculate the skill-assessment metrics. The Python program is also responsible for calculating the statistics and the ROC curves. Since the Python program operates on the output files of the image-processing program, buffer or TCP/IP communication is not used between the two programs.

Chapter 3

Suturing skill assessment based on needle motions

SutureCoach and the original simulator are equipped with a camera inside the membrane housing (see Fig. 3.1(a)). The camera is for recording needle motion during the suturing exercise (see Fig. 3.1(b)). Needle motion is then detected by segmenting needle from video frame (see Fig. 3.1(c)). To assess participants' suturing skills, metrics are used to examine the properties of needle motion, such as the needle tip trajectory.



(a) (b) (c)

Figure 3.1: Detecting needle motion. (a) Membrane housing; (b) Video frame from the camera; (c) Result of needle segmentation

This work uses the metrics in [51, 49] and new metrics for suturing skill assessment. The new metrics are developed after refining the suture phase segmentation scheme in [49]. These metrics

are designed to measure if the following suturing rules have been satisfied:

- The suturing process should follow the curvature of the needle [78, 40].
- The suturing process should minimize the wrench between the tissue and the needle [41].

These rules help to minimize internal tissue stress during the suturing process, which consequently reduces additional tissue trauma [41].

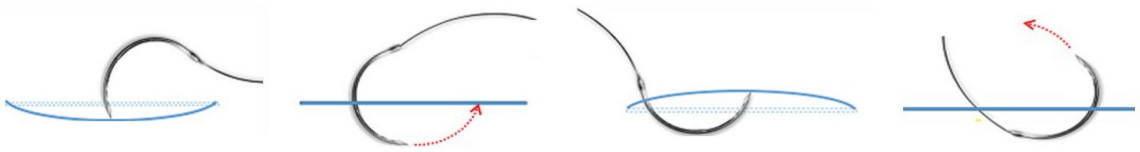
3.1 Suture phase segmentation

The general suture rules in [78, 74, 41] divide the suturing process into several essential steps, motivating us to improve the suture phase segmentation scheme used in the original simulator (see Fig. 3.2).

3.1.1 Suture phase segmentation scheme in the original simulator

In the original suturing simulator [49], a suture is decomposed into four phases:

- The entry phase denotes the period of puncturing the needle into the membrane.
- The driving phase denotes the period of driving the needle along some path under the membrane.
- The exit phase denotes the period of exiting the needle tip from the membrane.
- The pull-out phase denotes the period of pulling the needle completely from the membrane.



(a) Entry phase (b) Driving phase (c) Exit phase (d) Pull-out phase

Figure 3.2: Suture phase segmentation used in the original simulator. The needle and the membrane are denoted by the black semi-circles and the blue curves, respectively. The camera, under the membrane, is not shown for brevity. The image is from [49]

This suture phase segmentation scheme provides a guideline for the essential suture steps, but the boundaries between the four phases are undefined, which consequently hinders segmenting

sensor data for each phase. This shortcoming motivates us to improve the suture phase segmentation scheme by determining the boundaries between the suture phases.

3.1.2 The improved suture phase segmentation scheme

As shown in Fig. 3.3, the improved suture phase segmentation scheme divides a suture based on the events (i.e., tip entry time, tip exit time, and swage exit time) determined by the image-processing program and the Python program. For each suture, the tip entry time is the time of the first video frame in which the needle is detected, while the swage exit time is the time of the last video frame in which the needle is detected. The tip entry time and the swage exit time are determined by the image-processing program in the data-collection stage.

The tip exit time, determined in the data-analysis stage, is defined as when the needle tip starts piercing the membrane in an upward direction. Its calculation requires the needle exit point and the needle tip trajectory detected in the data-collection stage. By calculating the distance between the needle exit point and the points on the trajectory, the tip exit time is chosen as the timestamp corresponding to the last needle tip point that is $1mm$ away from the needle exit point.

After the segmentation based on the three events, a suture is split into 4 phases.

- The approach phase is the period before the needle punctures the membrane.
- The insertion phase is the period when a participant inserts the needle body into the membrane housing and then moves the needle tip toward the exit point.
- The removal phase is the period when a participant re-grasps the needle tip and then removes the needle body from the membrane housing.
- The pull-out phase is the period for pulling out the thread.

The improved suture phase segmentation scheme opens up the possibility of extracting sensor data at each suture phase, which helps to interpret the data and develop new metrics for suturing skill assessment.

3.2 Skill assessment metrics from previous works

Needle Tip Path Length (`TipPathLen`) is the length of the needle tip trajectory during the insertion phase. Similar to the path length of the instrument's tip [75], `TipPathLen` is designed

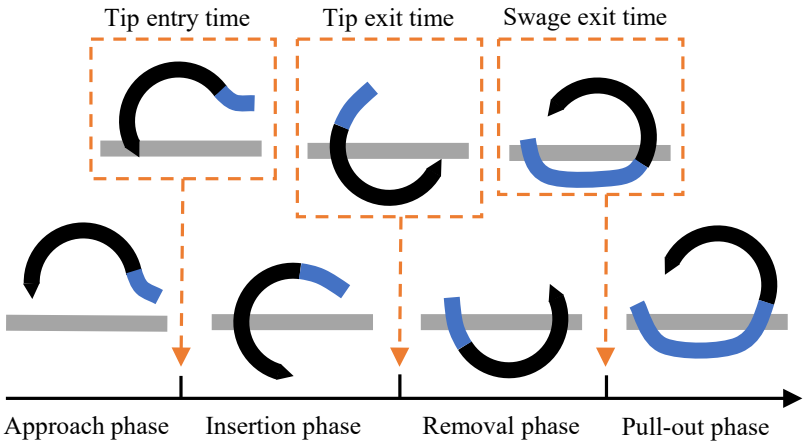


Figure 3.3: The improved suture phase segmentation. The needle, the thread, and the membrane are denoted by the black semi-circles, the blue curves, and the gray line, respectively. The camera, under the membrane, is not shown for brevity.

to measure participants' motion economy. Also, a small `TipPathLen` indicates the suturing trajectory follows the curvature of the needle, as any deviation from the ideal trajectory would increase `TipPathLen`.

To reduce the noisy data, the noise-removal techniques in [49] were adopted to filter the detected tip points. `TipPathLen` is calculated by

$$\texttt{TipPathLen} = k_{cam} \sum_{i=1}^{n-1} ||p_{i+1} - p_i||_2 \tag{3.1}$$

where k_{cam} denotes the pixel-to-mm conversion ratio (i.e., 0.21 for our simulator), p_i denotes the i^{th} filtered tip point in Cartesian coordinate system, and n denotes the number of filtered tip points.

Needle swept area (`SweptArea`) is the accumulated needle locations during the insertion and removal phase. The metric is designed to measure if the needle-driving process follows the curvature of the needle [40] as well as the wrench between the tissue and the needle during suturing. An experienced surgeon is expected to have small `SweptArea`.

To calculate `SweptArea`, the image-processing program detects the needle in each frame captured by the camera inside the membrane housing (see Fig. 3.1(a)). After each suture, `SweptArea` is the union of the detected needle locations. The unit of the result is then converted to mm^2. `SweptArea` is calculated by

Figure 3.4: An illustration of `SwayLen`. (a) Positive `SwayLen`; (b) Negative `SwayLen`. The needle, the entry point, and the exit point, `SwayLen` are denoted by the black curve, $\odot$, $\otimes$, and the red dashed line, respectively.

$$\texttt{SweptArea} = k_{cam}^2 * n \tag{3.2}$$

where n denotes the number of 'on' pixels in the binary image.

Needle sway length (`SwayLen`) measures the needle rotation about the axis from the needle entry point to the needle tip. The metric is designed to measure the wrench between the tissue and the needle during suturing. A small `SwayLen` indicates few lateral movements of the needle, which also implies high suturing skills.

To calculate `SwayLen` for image i, the needle entry point is virtually connected to the needle tip. An orthogonal line is then drawn at the midpoint of the connection. The distance, from the midpoint to the intersection of the orthogonal line and the needle body, is defined as $\texttt{SwayLen}_i$ (denoted by the red dashed lines in Fig. 3.4). Notice that the sign of `SwayLen` is determined by the direction of the unit vector with respect to the orthogonal line.

Suppose the insertion and the removal phase of a suture contains n frames in total, `SwayLen` is calculated by

$$\texttt{SwayLen} = \max_i\{\texttt{SwayLen}_{1,\dots,n}\} - \min_i\{\texttt{SwayLen}_{1,\dots,n}\} \tag{3.3}$$

Stitch length (`StitchLen`) is the distance between the needle entry point p_{en} and the needle exit point p_{ex}. This metric was introduced in [25] to distinguish experts' and novices' performance for closing a straight incision on a foam skin pad, so it is assumed that the metric can also be used for skill assessment of radial suturing. `StitchLen` is obtained by

$$\texttt{StitchLen} = k_{cam}||p_{en} - p_{ex}||_2 \tag{3.4}$$

3.3 Skill assessment metrics based on suture phases

Using the segmented suture phases, the suturing skill can be further evaluated by four metrics, namely the needle angle range (`Range(NAngle)`), the averaged needle angle (`AVG(NAngle)`), the range of orthogonal tip projection (`Range(TipOrth)`), and the duration for needle removal (`RemovalTime`). Since `Range(NAngle)`, `AVG(NAngle)`, and `Range(TipOrth)` are affected by the accuracy of the detected tip points, the filtered tip points for `TipPathLen` are used. Notice that `Range(NAngle)`, `AVG(NAngle)`, and `Range(TipOrth)` evaluate participants' performance during the insertion phase, while `RemovalTime` evaluates participants' performance during the removal phase.

`Range(NAngle)` measures the needle rotation about the entry point during the insertion phase, where the needle angle (`NAngle`) is defined as the angle between the tip-to-entry connection and the entry-to-exit connection in a single video frame (see Fig. 3.5). A participant with a small `Range(NAngle)` is expected to have high suturing skill, as the best sutures have minimum wrench between the tissue and the needle.

Equation 3.5 is for `NAngle` calculation. Specifically, v_{tip} denotes the vector pointing from the entry point to the needle tip, while $\hat{e}_{tan}$, and $\hat{e}_{orth}$ are the tangential and the orthogonal unit vector for the entry-to-exit connection, respectively.

$$\texttt{NAngle} = \text{atan2}(v_{tip} \cdot \hat{e}_{orth}, v_{tip} \cdot \hat{e}_{tan}) \tag{3.5}$$

Suppose the insertion phase of a single suture contains n frames, `Range(NAngle)` is then determined by

$$\texttt{Range(NAngle)} = \max_i\{\texttt{NAngle}_{1,\dots,n}\} - \min_i\{\texttt{NAngle}_{1,\dots,n}\} \tag{3.6}$$

`AVG(NAngle)` is the mean of absolute `NAngle` during the insertion phase. Similar to `Range(NAngle)`, `AVG(NAngle)` indicates the wrench between the tissue and the needle during suturing, so `AVG(NAngle)` is expected to be inversely proportional to the experience levels. `Range(NAngle)` is calculated by

$$\texttt{AVG(NAngle)} = \frac{\sum_{i=1}^{n} |\texttt{NAngle}_i|}{n} \tag{3.7}$$

where n denotes the number of video frames during the insertion phase.

`Range(TipOrth)` measures the range of needle tip lateral movement during the insertion

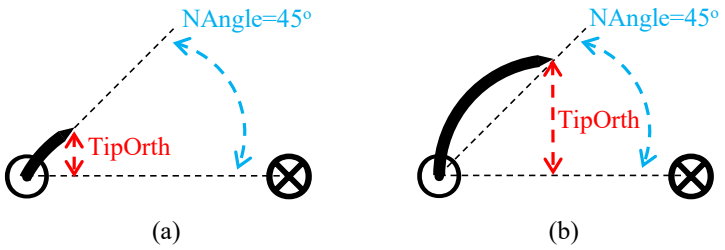


Figure 3.5: `NAngle` versus `TipOrth`. (a) At the beginning of the insertion phase; (b) At the middle of the insertion phase. Notice that (a) and (b) has identical `NAngle` but different `TipOrth`. The needle, the entry point, and the exit point are denoted by the black curve, $\odot$, and $\otimes$, respectively.

phase. The orthogonal tip projection (`TipOrth`) denotes projecting the tip-to-entry connection in a single video frame to $\hat{e}_{orth}$, where $\hat{e}_{orth}$ denotes the orthogonal unit vector for the enty-to-exit connection. A small `Range(TipOrth)` indicates a minimum wrench between the tissue and the needle, which implies high suturing skill.

Compared with `NAngle`, `TipOrth` requires higher detection accuracy of the needle tip, but it is more sensitive to needle deviation when the needle tip approaches the exit point. An illustration is shown in Fig. 3.5. It is noticeable that the needle lateral movement is significantly larger in Fig. 3.5(b) than in Fig. 3.5(a). The difference between `NAngle` in Fig. 3.5(a) and (b), however, is much smaller than the difference between `TipOrth` in the two subfigures.

Equation (3.8) is for the calculation of `TipOrth`. Specifically, v_{tip} denotes the vector from the entry point to the needle tip, while $\hat{e}_{orth}$ represents the orthogonal unit vector for the needle entry-to-exit connection.

$$\texttt{TipOrth} = v_{tip} \cdot \hat{e}_{orth} \tag{3.8}$$

Suppose the insertion phase of a single suture contains n frames, `Range(TipOrth)` is then determined by

$$\texttt{Range(TipOrth)} = \max_i\{\texttt{TipOrth}_{1,\dots,n}\} - \min_i\{\texttt{TipOrth}_{1,\dots,n}\} \tag{3.9}$$

`RemovalTime` is how long a participant spends in the removal phase. Time is a prevalent metric for suturing skill assessment [11, 6, 10], so it is expected that the duration in the removal phase can also distinguish participants with different skill levels. `RemovalTime` is calculated by

$$\texttt{RemovalTime} = T_{swage\ exit} - T_{tip\ exit} \tag{3.10}$$

where $T_{swage\ exit}$ and $T_{tip\ exit}$ denote the swage exit time and the tip exit time, respectively.

3.4 Comparing metric performance

The first statistical question examined is whether the metric values for the three groups have different means. Since the dataset contains a large number of individual sutures, this paper uses one-way analysis of variance (one-way ANOVA) and Welch's t-test to answer this question. If $p < 0.05$, the group means are considered to be different.

One-way ANOVA and Welch's t-test examine whether the metric performance of the three groups is different, but the ultimate goal of skill assessment is to classify the performance of individual participants. Receiver Operating Characteristics (ROC) curves are specifically designed to visualize the performance of two-group threshold-based classifiers and thus are useful for comparing the effectiveness of individual metrics for use in skill assessment. Specifically, an ROC curve shows how the true positive rate (TPR) and the false positive rate (FPR) vary when changing the classifier threshold value.

The metric classification performance is evaluated by the ROC curves for the suture classifier and for the trial classifier. For the suture classifier, a participant's performance is determined based on a single suture. The trial classifier, however, determines a participant's performance by the median metric value of a single suturing exercise. Compared with the suture classifier, the trial classifier is expected to have higher performance, since the decision is based on more measurements. Specifically, the median summarizes the participant's performance on the 12 sutures in an exercise. Further, the median is insensitive to extreme values that might arise due to sensor noise.

A popular way to evaluate the performance of a classifier is to measure the area under its ROC curves, known as area under curve (AUC). AUC is considered to be a measure of aggregate classification performance, as an ROC curve includes TPR and FPR determined by all possible decision thresholds [23]. For example, a random classifier has AUC equal to 0.5, since its ROC curve is the diagonal from $(0, 0)$ to $(1, 1)$. In contrast, AUC for an ideal classifier is equal to 1, as the corresponding ROC curve approaches the top-left corner of the ROC graph [31].

Chapter 4

Data collection

SutureCoach was used to collect data on campus and in several vascular conferences over a 5-month period [76, 83]. Before using the simulator, each participant was required to fill out a consent form and a questionnaire that recorded the subject's characteristics (e.g., age and height). The questionnaire is shown in Appendix A. The experimental procedures were approved by Clemson University Institutional Review Board (IRB number: IRB2020-387).

Since SutureCoach and the original simulator use the same suturing pattern, The two simulators have similar instructions for the suturing exercise. To complete the exercise on SutureCoach, a participant starts inserting the needle at the mark on Suture 1 (see Fig. 4.1). The ideal needle pull-out location is on the same radius, such that the edge of the brown circle is in the middle between the mark and the pull-out location. The participant then sutures at Suture 2, using the same procedure. During the exercise, participants are allowed to replace used needles with new ones. The exercise is complete when all 12 sutures have been finished. When collecting the dataset, all participants received identical instructions for using SutureCoach.

During data collection, participants were required to perform four exercises on SutureCoach: The first exercise is at surface condition (see Fig. 2.3(d)), followed by an exercise at depth condition (see Fig. 2.3(e)). They then repeated the two exercises in the same order. All exercises used the 3-0 Prolene sutures (26mm 1/2c SH needle).

Recall that the software of SutureCoach is controlled by keyboard inputs, so an operator is required during the data collection. When setting up SutureCoach, the software is turned on by running the executable file. Before each suturing exercise, the operator presses the **w** key to

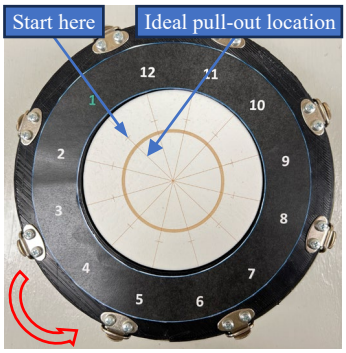


Figure 4.1: An illustration for performing the suturing exercise on SutureCoach. The red arrow denotes the suturing direction.

activate the software so the simulator starts recording and processing the sensor data. If a suture cannot be processed by the software (indicated by the terminal output), the operator is required to press the **m** key, so the data collection process can be continued. It was observed that the **m** key was used occasionally when collecting the dataset, which was due to participants' improper suturing techniques, such as failing to puncture the membrane. After an exercise is finished, the **s** key should be pressed so the software stops recording data for the current exercise. After the data collection process is completed, the **q** key should be pressed so the software is terminated appropriately.

The dataset includes 97 participants, which are then separated into three groups:

- The expert group consists of 35 participants who are fellows or attending surgeons.
- The intermediate group consists of 32 resident surgeons whose postgraduate year (PGY) is from one to five.
- The novice group consists of 30 participants who are medical students or graduate students.

Fig. 4.2 shows the distribution for the years of experience in intermediate and expert groups. In Fig. 4.2, a one-year fellow is considered to have six years of experience, while a one-year attending surgeon is considered to have eight years of experience. The red dashed line in Fig. 4.2 denotes the boundary between intermediate and expert groups. Notice that the students in the novice group are considered to have zero years of experience, so the novice group is not shown in Fig. 4.2.

For our dataset, the expert group, the intermediate group, and the novice groups have 1695, 1509, and 1500 detected sutures, respectively. Notice that some of the suture data (less than 0.5%) are removed from the dataset, as participants finished the sutures without puncturing the

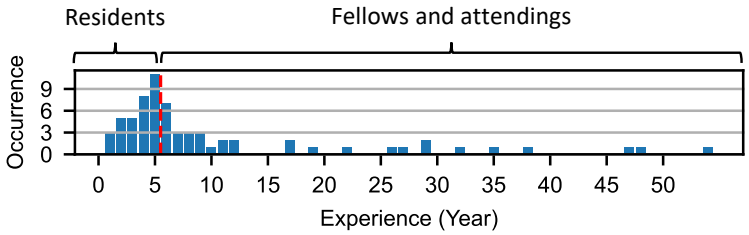


Figure 4.2: Distribution for the year of experience in intermediate and expert groups. The intermediate group is on the left of the red dashed line, while the expert group is on the right of the red dashed line.

membrane, which consequently caused the image-processing program to have inaccurate needle and thread detection.

Chapter 5

Results and discussion for suturing skill assessment via needle motion

5.1 Results

For the eight metrics, one-way ANOVA shows that the means of the three groups are significantly different at both surface and depth conditions. Also, Levene's test shows that the metrics have different variances among the three groups at the two conditions.

The confidence intervals calculated by Welch's t-test are shown in Fig. 5.1, where each subfigure shows the comparisons of experts versus intermediates (E-I), experts versus novices (E-N), and intermediates versus novices (I-N) for a specific metric. The comparisons are made separately for surface and depth conditions. The confidence interval denotes the range of likely differences between the two group means. Therefore, if a confidence interval does not cross zero, indicated by the red dashed line, then the null hypothesis is rejected. In other words, if zero is not in a confidence interval, there is a statistically significant difference between the mean of the two groups. Detailed statistical analysis for the eight metrics are summarized in Table 1 in Appendix B.

Boxplots for the eight metrics are shown in Fig. 5.2, where the metric values for the three groups are presented at surface and depth conditions separately. Fig. 5.2 also includes the mean metric values for each group. Extreme outliers are not shown in Fig. 5.2 to avoid cluttering the figures.

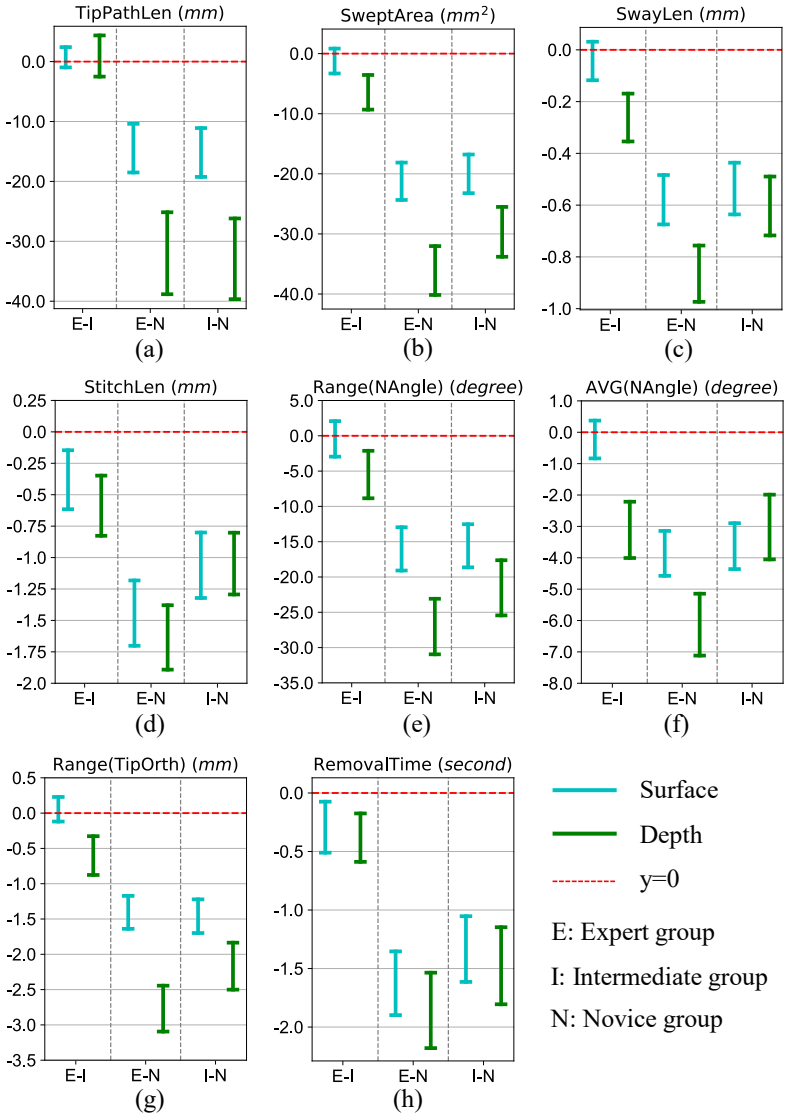


Figure 5.1: Confidence intervals for the eight image-based metrics showing pairwise comparisons of experts versus intermediates (E-I), experts versus novices (E-N), and intermediates versus novices (I-N). For each pairwise comparison, the surface and the depth condition are shown separately.

Fig. 5.3 and Fig. 5.4 present ROC curves for the eight metrics at the depth condition. Specifically, Fig. 5.3 shows the ROC curves for the suture classifier, while Fig. 5.4 shows the ROC curves for the trial classifier. In the two figures, the group with more experience is considered to be positive.

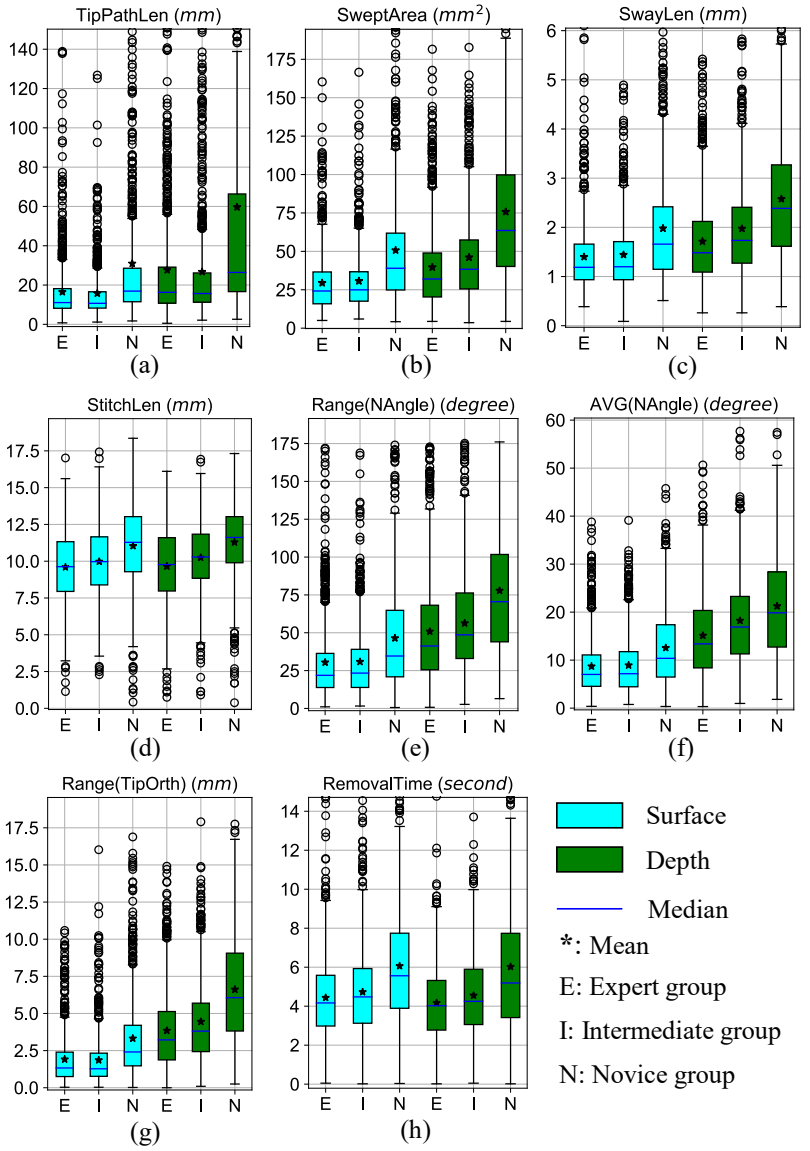


Figure 5.2: Boxplots of the eight image-based metrics. Each sub-figure presents the outputs of a single metric, separated into the three groups, at surface and depth conditions. The extreme outliers are not shown in the boxplots.

5.2 Discussion

Fig. 5.1 shows that the metrics are effective at distinguishing between students' and surgeons' performance. Moreover, the depth constraint enhances the ability to differentiate performance between resident surgeons and attending surgeons. More complex movements are used to suture at the depth condition, which is beneficial for distinguishing participants' suturing skill levels. This

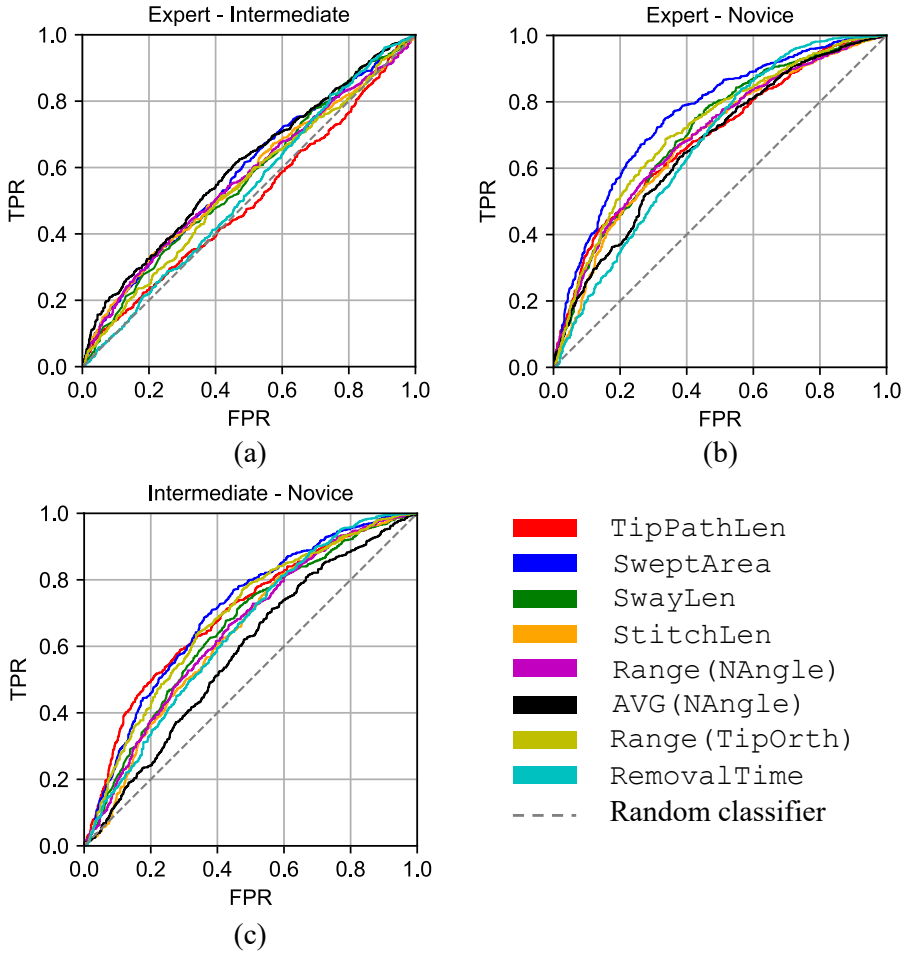


Figure 5.3: ROC curves for the suture classifier. Each subfigure shows the classification between two groups at the depth condition.

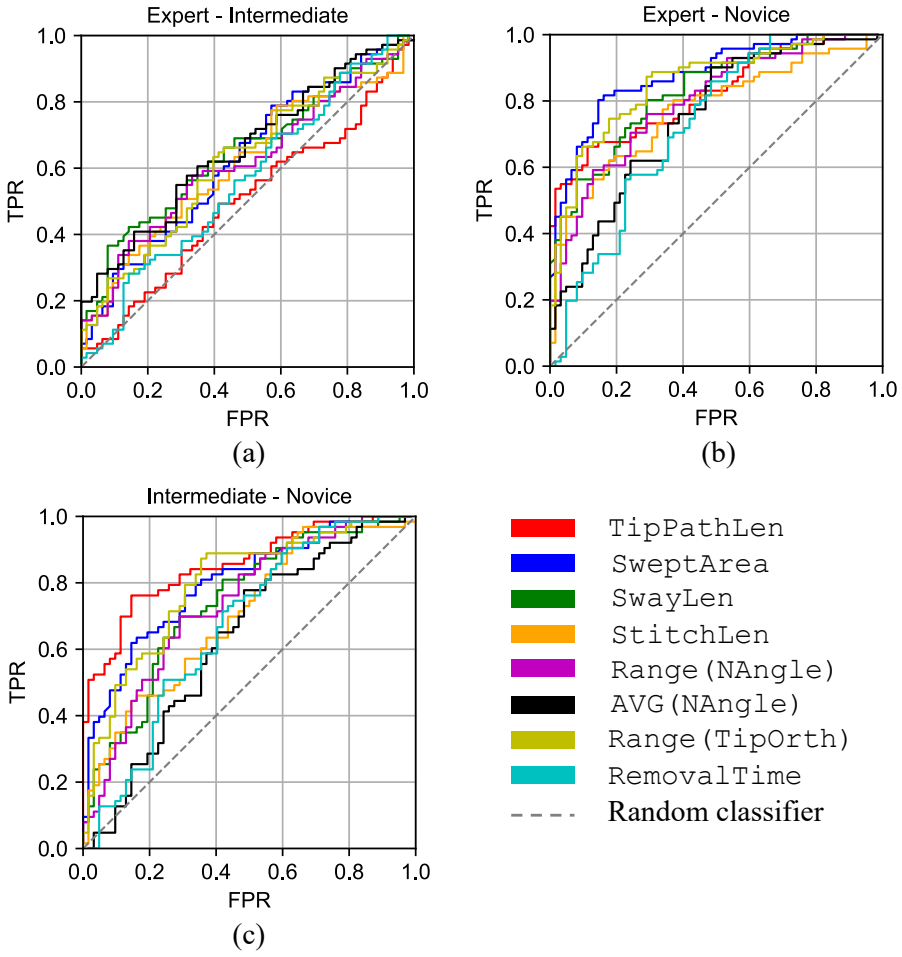


Figure 5.4: ROC curves for the trial classifier. Each subfigure shows the classification between two groups at the depth condition.

hypothesis is further supported by Fig. 5.2, which shows that each group tends to have larger metric values at the depth condition than at the surface condition.

`StitchLen` is one of the metrics that have significantly different means for the three groups at the surface condition; however this result might be an artifact of the instructions to participants. During data collection, participants were asked to make sutures symmetric about the incision line on the membrane (shown by the brown circle in Fig. 2.3(b)). The geometry of the specified entry locations and incision line used in this study made the ideal `StitchLen` to be 13mm, which is considered uncomfortably long from surgeons' feedback. As shown in Fig. 5.2(d), attending surgeons tended to make shorter stitches than the resident surgeons, suggesting that the experienced participants were more likely to disregard the instructions.

Fig. 5.2 shows that the expert group achieves the smallest range of metric values at the depth condition for six out of eight metrics, suggesting that attending surgeons have more consistent performance as compared to resident surgeons and students. Fig. 5.2 also shows that the expert group has the smallest median and mean at the depth condition for metrics `SweptArea`, `SwayLen`, `Range(NAngle)`, `AVG(NAngle)`, and `Range(TipOrth)`, indicating the attending surgeons have less lateral needle movement than the other participants. Furthermore, the time measurement metric (i.e., `RemovalTime`) suggests the attending surgeons are more efficient than the other two groups in the removal phase at the depth condition (see Fig. 5.2(h)).

Like [49], `SweptArea`, `SwayLen`, and `StitchLen` are capable of distinguishing the suturing skill between resident and attending surgeons at the depth condition in our dataset. Although `TipPathLen` is able to distinguish the resident surgeon group and the attending surgeon group in [49], it does not show significant differences in the current dataset. In the current simulator, the needle occasionally reflects the LED light, causing the detected needle tip position to change between the needle midpoint and the actual needle tip. The `TipPathLen` metric is particularly sensitive to this noise in tip detection, which may explain the decreased effectiveness of the `TipPathLen` metric in the present study.

As shown in Fig. 5.1, `Range(NAngle)`, `AVG(NAngle)`, and `Range(TipOrth)` have significantly different means for the three groups at the depth condition. These three metrics outperform `TipPathLen`, in part because they are less likely to be affected by noisy needle tip detection.

Thus, Fig. 5.1 and 5.2 indicate that the metrics reveal the different performances among the three groups, which are in various experience levels.

Comparing Fig. 5.1 and 5.3, notice that even if two groups have statistically different means for some metrics, that does not guarantee high classification performance. For example, even though `RemovalTime` has $p < 0.05$ for the E-I comparison at the depth condition (see Fig. 5.1(h)), the corresponding ROC curve indicates `RemovalTime` has similar classification performance as a random classifier (see Fig. 5.3(a)). In contrast, `SweptArea` not only has significant mean differences for E-N and I-N comparison (see Fig. 5.1(b)) but also has the largest AUC among the eight metrics when classifying students from surgeons at the depth condition (see Fig. 5.3(b) and (c)).

Fig. 5.3 demonstrates the ROC curves for the suture classifier. ROC curves are examined for the depth condition only, since several metrics showed statistically significant differences between experts and intermediates only at depth, but not at surface. From Fig. 5.3(b) and (c), note that `SweptArea` has the largest AUC between the student and the surgeon groups, followed by `Range(TipOrth)`. In contrast, the ROC curve of `AVG(NAngle)` has the smallest AUC for the student-surgeon classification. Fig. 5.3(b) and (c) also show that `Range(TipOrth)` outperforms `Range(NAngle)` and `AVG(NAngle)` for E-N and I-N classification, but `AVG(NAngle)` has better classification performance than `Range(TipOrth)` and `Range(NAngle)` for E-I classification (see Fig. 5.3(a)).

The ROC curves for the trial classifier are shown in Fig. 5.4. The resolution of these ROC curves (Fig. 5.4) is lower than the resolution of the suture classifier ROC curves (Fig. 5.3), since there are twelve times as many sutures as trials. In Fig. 5.4(c), `TipPathLen` has significantly larger AUC than in Fig. 5.3(c). One reason is the noisy tip detection decreases the performance of `TipPathLen` in the suture classifier.

By comparing the AUC in Fig. 5.3 and 5.4, it is noticeable that the trial classifier outperforms the suture classifier, indicating the trial classifier should be used to distinguish participants' performance. Specifically, the trial classifier with `AVG(NAngle)` has the best performance for E-I classification (see Fig. 5.4(a)), while the trial classifier with `TipPathLen` has the best performance I-N classification (see Fig. 5.4(c)).

Thus, Fig. 5.3 and 5.4 indicate that the metrics can classify participants' suturing skill levels.

In the present work, clinical standing (i.e., attending surgeons and fellows; resident surgeons; and students) is used as a proxy for suturing skill, but suturing skill is not necessarily directly determined by clinical standing. A more objective evaluation of the suturing metrics could be

obtained if an improved ground-truth assessment of suturing skill was available. For example, the suturing skill could be independently assessed using expert ratings from several experienced surgeon educators.

Fig. 4.2 shows that the threshold for splitting the intermediate and the expert group is close to the peak in the year-of-experience distribution. Since years of experience is only an approximation of suturing skill, the cluster of participants close to the intermediate-expert boundary may lead to increased misclassification, even if the suturing metrics correlate well with suturing skill. This observation is supported by the results in Section 5.1, both the statistical results and the ROC curves show that the eight metrics are capable of distinguishing students from surgeons, but their performances decrease when comparing the resident surgeons with the attending surgeons. It is expected that this problem will be alleviated by using other techniques to evaluate participants' actual suturing skill.

Chapter 6

Deep learning algorithms for hand roll estimation

6.1 Introduction

6.1.1 Motivation

To complete the suturing exercise on SutureCoach, participants are required to drive the needle by rotating their wrists, such that the palm is facing upward/downward. An example is shown in Fig. 6.1. Since the ideal pull-out location is in the brown circle, the participant needs to rotate the wrist to make the palm face downward, so the needle can approach the ideal pull-out location. This motion can be considered as the hand rotation about the roll axis (see Fig. 6.1), so the motion is abbreviated as hand roll hereafter.

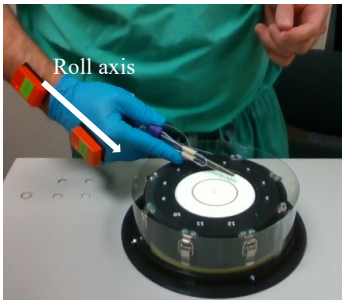


Figure 6.1: A demonstration of the suturing exercise on SutureCoach. The figure is adapted from [76]

6.1.2 Prior work

Hand roll has proven to be an effective measurement of participants' suturing skills, as hand roll angle and roll velocity can be used to calculate hand-roll metrics for suturing skill assessment [21, 76]. Dubrowski et al. [21] use an electromagnetic motion analysis system to estimate participants' hand roll angles when they suture on an artificial artery model. The results show that the junior resident surgeons and the faculty surgeons have significantly different roll during the exercise, where roll refers to the difference between the minimum and the maximum of hand roll angle. Shayan et al. [76] use inertial measurement units (IMU) to measure participants' hand roll velocity when they practice on SutureCoach. The results show that metrics based on hand roll velocity have significantly different outcomes among students, resident surgeons, and attending surgeons. These studies, however, require participants to wear a specific sensor during data collection, which would interfere with their suturing performance.

6.1.3 Hand roll estimation and suturing skill assessment via video analysis

To reduce potential interference from wearing sensors, this work proposes using a video analysis program for hand roll estimation and suturing skill assessment. The pipeline for the program is shown in Fig. 6.2. After receiving the videos with hand motion, the program first crops the hand region by a hand detection algorithm. The cropped images are then processed by an algorithm for hand roll angle and velocity estimation, such that the estimated roll angle and roll velocity are similar to the IMU measurement. The estimated hand roll angle or roll velocity is then used to calculate hand-roll metrics in [76] for suturing skill assessment.

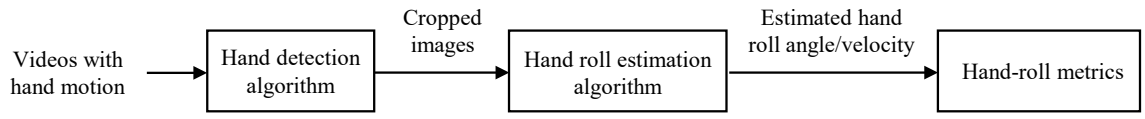


Figure 6.2: Pipeline for the video-analysis program for hand roll estimation and suturing skill assessment.

This chapter is organized as follows: Section 6.2 describes the data used for the two algorithms and data for calculating hand-roll metrics. Section 6.3 describes the hand detection algorithm. Section 6.4 describes the hand roll estimation algorithms. Section 6.5 describes the hand-roll met-

rics in [76], which are for examining if the algorithm output can be used for surgical suturing skill assessment.

6.2 Data description

Recall that SutureCoach has been used to collect data at different locations over a 5-month period [76, 83]. Fig. 6.3 shows the video frames from Camera 2 (see Fig. 2.3(a)) at the 5 data collection locations, where Location A, B-D, and E, were at a hospital, vascular conferences, and a graduate student office, respectively.

Figure 6.3: Video frames captured at the 5 data collection locations.

Table 6.1 shows the total number of sutures used in this study, which involves sutures for the hand detection dataset and sutures for the hand roll estimation dataset. The two datasets are used to train the algorithms shown in Fig. 6.2. After training the algorithms, the total number of sutures is used to calculate the hand-roll metrics for suturing skill assessment. The number of sutures in Table 6.1 is smaller than the number of sutures used for the image-based metrics (see Chapter 4), as some sutures are removed due to underexposed videos from Camera 2.

Table 6.1: Number of sutures used for training algorithms and suturing skill assessment

Total sutures			
Condition	Expert	Intermediate	Novice
Surface	676	615	594
Depth	646	619	548
Sutures for the hand detection dataset			
Condition	Expert	Intermediate	Novice
Surface	84	48	48
Depth	72	24	72
Sutures for the hand roll estimation dataset			
Condition	Expert	Intermediate	Novice
Surface	60	144	48
Depth	59	180	36

A custom Python program (see Fig. 6.4) is used to average roll angle and roll velocity

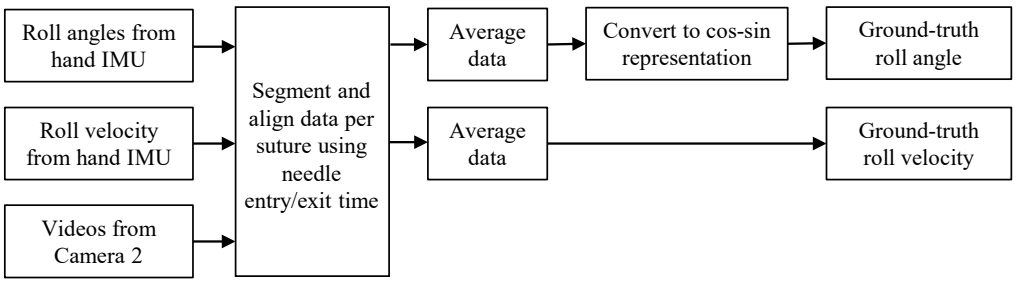
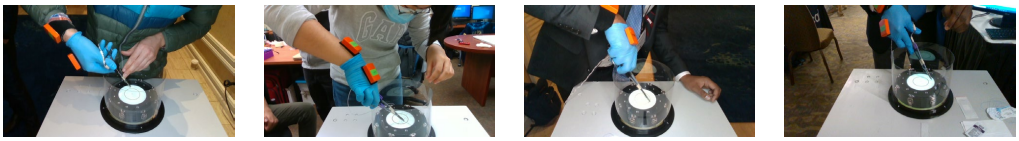


Figure 6.4: The custom Python program for averaging roll angles and roll velocity.

measured by IMU, as IMU records data at 120 Hz but Camera 2 records videos at 30 fps. To average the data, the Python program first uses needle entry/exit time per suture to segment roll measurement from IMU (see Fig. 2.3(d)) and videos from Camera 2 (see Fig. 2.3(a)). After aligning IMU measurement with video frames via timestamps, the roll angle and the roll velocity for a single video frame are then represented by the mean of its closest IMU measurement and its three preceding IMU measurements. For simplicity, the average roll angle is denoted by the ground-truth roll angle, and the average roll velocity is denoted by the ground-truth roll velocity hereafter. The Python program also converts the roll angles from Euler angle representation to cos-sin representation, which guarantees the hand roll estimation algorithm estimating continuous signals. A detailed discussion of Euler and cos-sin representations is in Section 6.4.1.2.

6.2.1 Hand detection dataset



(a) (b) (c) (d)

Figure 6.5: Video frames from the 4 videos that record blue gloves and other blue objects. Those 4 videos are included in the hand detection dataset.

The hand detection algorithm aims to detect the participant's blue glove in different environments, so the hand detection dataset should involve images with different backgrounds and ambient light. Thus, each data collection location in Fig. 6.3 has been randomly chosen 5 videos to construct the hand detection dataset. For the 25 videos, each of them records a suturing task at the surface condition (see Fig. 2.3(d)) or at the depth condition (see Fig. 2.3(e)). To reduce the false

detection of the algorithm, 4 videos with other blue objects are also added to the hand detection dataset. Example video frames from the 4 videos are shown in Fig. 6.5. Since each of the videos records 12 sutures, the number of sutures in the hand detection dataset is summarized in Table 6.1.

For the 25 videos, the ground-truth bounding boxes of the blue glove are determined by color thresholding, as there is only a single blue object (i.e., the blue glove) in each video frame. To determine a bounding box, a Python program first converts video frames from RGB to HSV color space. The blue glove is then detected by the pre-defined color threshold. The smallest rectangle that encloses the detection result is defined as the ground-truth bounding box of the blue glove.

For the 4 videos including other blue objects, the ground-truth bounding boxes are manually labeled via a graphical annotation tool named LabelImg [56]. Since YOLO object detection algorithms require the bounding box to be normalized by image width w_i and image height h_i, each bounding box is denoted by $(\hat{x}, \hat{y}, \hat{w}, \hat{h})$.

$$\hat{x} = \frac{x}{w_i}; \quad \hat{y} = \frac{y}{h_i}; \quad \hat{w} = \frac{w}{w_i}; \quad \hat{h} = \frac{h}{h_i} \tag{6.1}$$

where (x, y) is the center, w is the width, and h is the height of the original bounding box.

The normalized bounding boxes, along with the video frames, are used to construct the hand detection dataset. when training the hand detection algorithm, the dataset is separated into training, validation, and test sets, which involve 48k, 16k, and 16k images, respectively.

6.2.2 Hand roll estimation dataset

The hand roll estimation algorithms aim to estimate participants' hand roll angles or roll velocity based on cropped images. Since a major portion of the cropped images is the blue glove, the hand roll estimation algorithms are assumed to be less sensitive to image backgrounds than the hand detection algorithm. Thus, the hand roll estimation dataset is constructed from 44 videos collected at a single location (i.e., Location D in Fig. 6.3). The 44 videos record suturing tasks at the surface condition (see Fig. 2.3(d)) and at the depth condition (see Fig. 2.3(e)). Since each video records 12 sutures, the number of sutures in the hand roll estimation dataset is summarized in Table 6.1.

The hand roll estimation dataset involves the average roll angle, average roll velocity, and cropped images of the blue gloves. The average roll angle and average roll velocity are calculated by the Python program shown in Fig. 6.4, while the cropped images are generated based on the

bounding box from the hand detection algorithm. To crop the blue glove from a video frame, the bounding boxes from the hand detection algorithm are expanded to squares. The square bounding box is centered at the blue glove unless the bounding box exceeds the boundary of the video frame. Based on the bounding box location, the glove portion was then cropped from the video frame. When training the hand roll estimation algorithms, the hand roll dataset is separated into training, validation, and test sets, which involve 100k, 33k, and 33k images, respectively. The training set is used to update the parameters such as weights, while the validation set is used to determine the hyper-parameters such as the learning rate [70]. After the training process is completed, the test set is used to provide an independent evaluation of the algorithm's performance [72].

6.3 Hand detection algorithm

6.3.1 Motivation

Since deep-learning object detection algorithms support detecting different types of objects in different environments, they have been used for surgical tool detection and hand detection [13, 29]. Deepika et al. [13] use mask-RCNN [32] to segment surgical tools from videos of neurosurgery. Their algorithm has high accuracy, but the processing speed is comparatively low. Specifically, mask-RCNN can only process 5 frames per second [32]. In comparison, Goldbraikh [29] et al. use YOLOv3 [66] to localize participants' hands and surgical tools from videos of suturing exercises. Their algorithm archives decent detection accuracy and near real-time processing speed (35 frames per second). This work motivates us to detect participants' hands via deep-learning object detection algorithms.

6.3.2 Hand detection using YOLOv7

For our video analysis program (see Fig. 6.2), the hand-detection algorithm is expected to run in real time, so the subsequent hand roll estimation algorithm can provide instantaneous output. Thus, the proposed hand detection algorithm will be developed based on YOLOv7 [94], which is the recent version in the YOLO family.

Studies show that YOLOv7 surpasses previous YOLO algorithms in both speed and accuracy [94]. Compared with previous YOLO algorithms, YOLOv7 introduces an efficient training method,

a robust loss function, and an efficient label assignment method [94]. To achieve efficient training, YOLOv7 uses RepConv [17] without identity connection to design a planned re-parameterization model, where re-parameterization is a technique that uses a large model at the training stage and then merges some model components to create a smaller model at testing stage [17]. To improve the loss function, YOLOv7 adds auxiliary heads in the middle layers of the network, so the shallow network can be guided by the assistant loss from the auxiliary heads. Research shows that using auxiliary heads improves the algorithm convergence speed and reduces testing errors [54]. To improve the label assignment method, YOLOv7 provides fine labels to the lead head, but coarse labels to the auxiliary heads. Compared with fine labels, coarse labels are less accurate, but coarse labels are more suitable for auxiliary hands that have limited learning ability [94].

6.3.3 Data pre-processing and hyperparameters

Since YOLOv7 has been trained on MS COCO dataset [55] and our hand detection task is comparatively simple, the YOLOv7 model [93] has been fine-tuned using the hand detection dataset. During the training process, the default hyperparameters on [93] are used. The input images are re-sized to 640×640, and the number of epochs is set to 20, chosen empirically. During testing, the input images are resized to 640×640, and the confidence threshold is set to 0.558. The confidence threshold is chosen based on the F-1 curve generated during the training process.

6.3.4 Evaluating algorithm performance

6.3.4.1 Mean average precision (mAP)

mAP is a common metric to evaluate the performance of object detection algorithms. The process of mAP calculation is as follows.

When YOLOv7 receives an image, the algorithm outputs the confidence and the bounding box of the detected objects. The accuracy of the predicted bounding box is evaluated by intersection-over-union (IoU), which is a ratio of the predicted bounding box ($\widehat{box}$) and the ground-truth bounding box (box).

$$IoU = \frac{\widehat{box} \cap box}{\widehat{box} \cup box} \tag{6.2}$$

If IoU of a detected bounding box is above the pre-defined IoU threshold, it is considered as a correct detection.

Based on the pre-defined confidence threshold and IoU threshold, a confusion matrix is generated. The matrix includes 4 categories.

- True positive (TP): The algorithm predicts a bounding box of the hand at a specific location (positive), and it is correct (true).
- False positive (FP): The algorithm predicts a bounding box of the hand at a specific location (positive), but it is wrong (false). It indicates the algorithm falsely determines other objects as the hand.
- False negative (FN): The algorithm does not predict a bounding box of the hand at a specific location (negative), but it is wrong (false). It indicates the algorithm ignores the hand.
- True negative (TN): The algorithm does not predict a bounding box of the hand at a specific location (negative), and it is correct (true). This corresponds to the image background.

Precision(P) and recall(R) are calculated based on the confusion matrix.

$$P = \frac{TP}{TP + FP} \tag{6.3}$$

$$R = \frac{TP}{TP + FN} \tag{6.4}$$

If the pre-defined confidence threshold varies, the values of P and R would change [4]. The various P and R can be plotted as a precision-recall curve, and the area under the curve is defined as average precision (AP) [9].

$$AP = \int_0^1 P(R)dR \tag{6.5}$$

mAP is the average of AP among different classes.

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{6.6}$$

where N denotes the number of classes. For the hand detection algorithm, there is only one class (i.e., hand), so $N = 1$.

6.3.4.2 Evaluation metrics used by YOLOv7

YOLOv7 uses P, R, mAP@0.5, and mAP@0.95 to as evaluation metrics. P and R are calculated using (6.4) by setting the confidence threshold to be 0.558 and the IoU threshold to be 0.65, where the confidence threshold is chosen based on F1-curve, and the IoU threshold is the default value. mAP@0.5 denotes mAP with IoU threshold being 0.5. mAP@0.95 denotes the average of mAP over different IoU thresholds, and the thresholds are from 0.5 to 0.95 with a step size of 0.05 [100].

6.4 Hand roll estimation algorithms

6.4.1 Comparing orientation representations

6.4.1.1 The 3D orientation representations used by IMUs

The IMUs on SutureCoach measure the object orientation in 3-dimensional space. The IMUs have three orientation representations:

- **Euler angles**. It describes orientation using the X (roll), Y (pitch), and Z (yaw) axis, so it is easy to interpret. The representation, however, has the gimbal lock. Specifically, the values at the roll and the yaw axes are undefined when the IMU points upward/downward (i.e., pitch=$\pm 90°$).
- **Unit quaternions**. It describes orientation using a 4-element unit vector. Unit quaternions are free from gimbal lock.
- **Rotation matrix**. It describes orientation using a 3×3 matrix. Rotation matrices are free from gimbal lock.

The three orientation representations, however, are unsuitable for training the deep-learning algorithms [107]. Specifically, Euler angles and unit quaternions suffer from discontinuity, while rotation matrices are an excessive representation. To train deep-learning algorithms for 3D orientation estimation, the study [107] recommends letting the algorithms estimate the first two columns (i.e., 6 values) in rotation matrices, which guarantees a continuous orientation representation and avoids the redundant information in the rotation matrices. Suppose the algorithm outputs (i.e., 6 values)

are separated into two 3×1 vectors a_1 and a_2, the estimated rotation matrix M can be calculated by

$$M = [b_1, b_2, b_3]^T \tag{6.7}$$

where

$$\begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} N(a_1) \\ N(a_2 - b_1 \cdot a_2 \cdot b_1) \\ b_1 \times b_2 \end{bmatrix} \tag{6.8}$$

In (6.8), $N(\cdot)$, $\cdot$, and $\times$ denote a normalization function, dot product, and cross product, respectively.

The algorithms in this chapter, however, are not for estimating the 3D orientation of participants' hands. Since SutureCoach was used to collect data at different locations at which the IMU initial orientation is unknown, it is challenging to calibrate the IMU to a canonical orientation (e.g., let IMU point to the magnetic north). Also, hand roll has been widely used for surgical suturing skill assessment [21, 76], and the rotation around the roll axis is independent of the IMU initial orientation. Thus, this work focuses on training deep-learning algorithms to estimate hand rotation around the roll axis.

6.4.1.2 Orientation representations at roll axis

Recall that the Euler angle representation provides an intuitive way to interpret the rotation around the roll axis, so the Euler angle at the roll axis can be directly used to represent the hand roll. The discontinuity of the Euler angle representation, however, makes it unsuitable for training deep-learning algorithms. Since the range of Euler angle is in $[-\pi, \pi]$ but the hand roll angle can exceed the range, a tiny hand roll can cause the switch from $-\pi$ to π, in which the magnitude is much larger than the actual hand roll. This discontinuity hinders the convergence of deep-learning algorithms, as studies show that deep-learning algorithms' approximation speed is positively proportional to the smoothness of the function to be approximated [102, 103]. Worse, the discontinuity causes Euler angle representation to have misleading error measurements when training the algorithms. An example is shown in Fig. 6.6(a). It is noticeable that the error between the IMU measurement and the estimation is significantly large, but $-\pi + \epsilon$ and $\pi - \epsilon$ actually represent similar orientations at the roll axis, indicating the algorithm should have a small penalty.

The cos-sin representation is an alternative to represent orientation at the roll axis. Different from Euler angles, the cos-sin representation uses a 2×1 vector to represent orientation. Research proves that the cos-sin representation is a continuous representation of 2D rotation, as it is the first column of the rotation matrix in 2D space [107]. Also, the training errors measured by the cos-sin representation reflect the actual errors of orientation. As shown in Fig. 6.6, the error between the estimation and the IMU measurement is much smaller in the cos-sin representation (see Fig. 6.6(b)) than in the Euler angle representation (see Fig. 6.6(a)). These observations suggest that the cos-sin representation should be used when training the deep-learning algorithms.

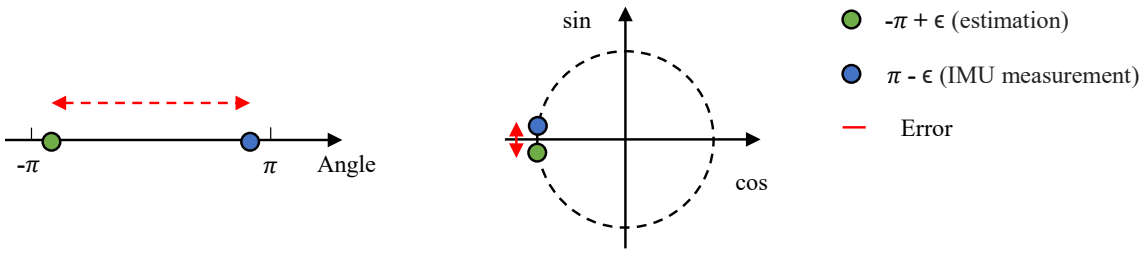
Suppose the hand roll angle θ is in Euler angle representation. Its corresponding cos-sin representation (α, β) is calculated by

$$\alpha = \cos(\theta) \tag{6.9}$$

$$\beta = \sin(\theta) \tag{6.10}$$

Equation (6.11) is used to convert the cos-sin representation (α, β) back to the Euler angle representation.

$$\theta = \text{atan2}(\beta, \alpha) \tag{6.11}$$



(a) (b)

Figure 6.6: An example of algorithm training error measured by (a) Euler angle representation and (b) cos-sin representation.

Since the cos-sin representation is beneficial to training deep-learning algorithms and the Euler angle representation provides an intuitive interpretation, the cos-sin representation is used when training the hand roll estimation algorithms, while the Euler angle representation is used when reporting the roll angle estimation error. To avoid misleading error measurements caused by

Euler angle representation, the errors are constrained in $[-pi, pi]$ by

$$Error = \begin{cases} \hat{\theta} - \theta + 2\pi, & \text{if } \hat{\theta} - \theta <= -\pi \\ \hat{\theta} - \theta - 2\pi, & \text{if } \hat{\theta} - \theta >= \pi \end{cases} \tag{6.12}$$

where $\hat{\theta}$ denotes the algorithm output and θ denotes the ground-truth roll angle.

6.4.2 Algorithm structures

Three algorithms are developed for the roll angle and roll velocity estimation (see Fig. 6.7). Since roll angle describes the magnitude of rotation and can be estimated by a single image, it can be obtained from the algorithm for image analysis (see Fig. 6.7(a)) and the algorithms for image sequence analysis (see Fig. 6.7(b) and (c)). In comparison, roll velocity is the amount of change in rotation in a given amount of time, which should be calculated based on a sequence of images. Thus, roll velocity can only be obtained from the algorithms for image sequence analysis (see Fig. 6.7(c)). These algorithms are written in Python with Pytorch deep-learning library.

6.4.2.1 Backbone algorithm

The algorithms in Fig. 6.7 use ResNet-50 as a backbone. ResNet-50 is a version of the residual convolutional neural network [33] that has 50 weighted layers. Compared with the ordinary feed-forward neural network such as VGG-16 [80], residual networks have a shortcut connection between the input and the output of each building block (see Fig. 6.8). Since the shortcut connection provides an identity mapping between the input and the output, the layers only need to learn the residual mapping [33]. Experiment results in [33] show that the residual networks outperform the counterpart networks that are without the shortcut connections, which indicates optimizing the residual mapping is easier than directly optimizing the mapping between the input and the output.

ResNet-50 consists of 16 building blocks. Their general structure is shown in Fig. 6.8, which includes a shortcut connection and three convolutional layers. Specifically, the first 1×1 convolutional layer is for reducing the number of channels in the feature map (i.e., $\alpha > \beta$ in Fig. 6.8), while the second 1×1 convolutional network is for increasing the number of channels in the feature map (i.e., $\gamma > \beta$ in Fig. 6.8). The stack of 1×1 convolutional layers, known as the bottleneck design, is used to decrease the computational cost. In comparison, the 3×3 convolutional layer is for

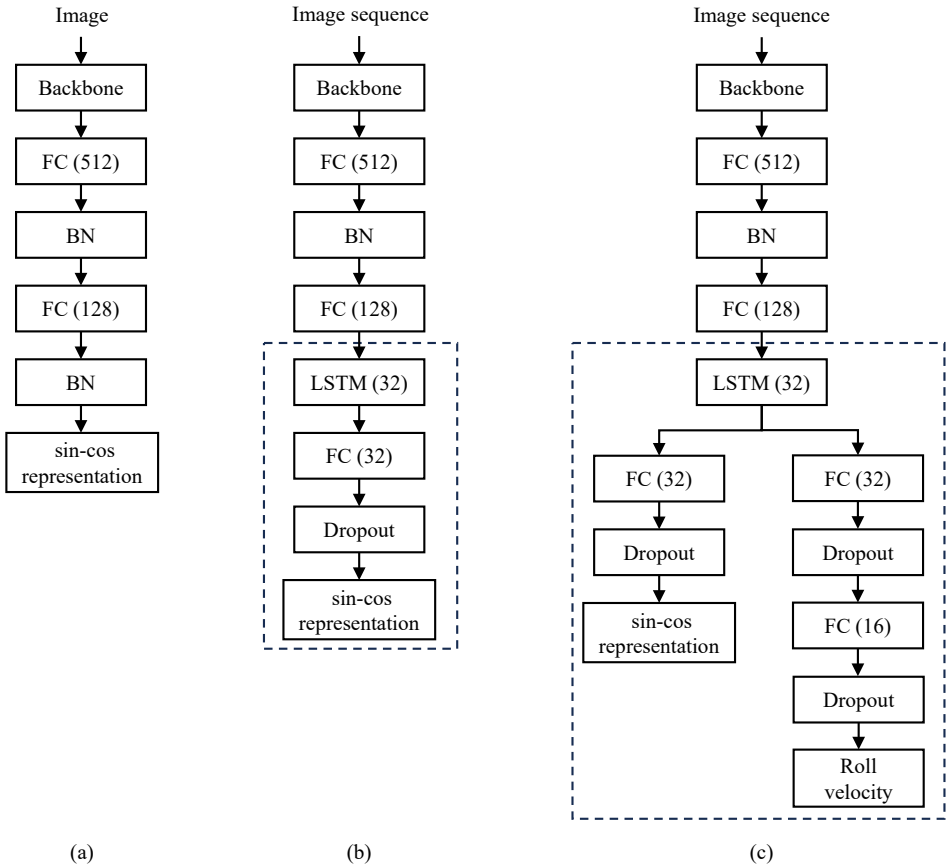


Figure 6.7: Algorithms for roll angle and roll velocity estimation. (a) CNN; (b) CRNN(angle); (c) CRNN(angle & vel). For (b) and (c), the trainable layers are in the dashed rectangle. BN denotes batch normalization, while FC denotes a fully connected layer. The number in parentheses denotes the number of neurons in a layer. The ReLU activation function is not shown in the figure for brevity.

learning the translation-invariant image features [30], so the algorithm can determine if the image includes a specific object without knowing the object location. Notice that the ReLU activation function [61] is used after each convolutional layer.

Like other convolutional neural networks, ResNet-50 gradually increases the number of channels in the feature maps. Thus, there are two types of shortcut connections in ResNet-50. If the input and the output of a building block have the same number of channels (i.e., α=γ in Fig. 6.8), the shortcut connection is an identity shortcut.

$$y = f(x, W_i) + x \tag{6.13}$$

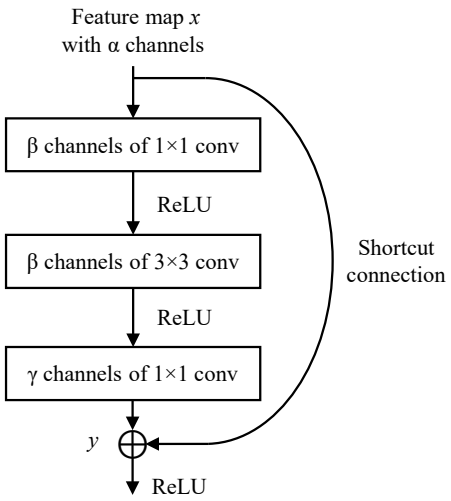


Figure 6.8: Building block for ResNet-50, where conv denotes convolution and $\oplus$ denotes element-wise addition.

where $f(x, W_i)$ denotes the residual mapping learned by the three convolutional layers in a building block. Notice that the second term in (6.13) indicates the identity shortcut preserves the input x.

On the other hand, if the input and the output have different numbers of channels (i.e., $\alpha < \gamma$ in Fig. 6.8), the shortcut connection is a projection shortcut.

$$y = f(x, W_i) + W_s x \tag{6.14}$$

where W_s denotes the matrix for linear projection. Note that $f(x, W_i)$ and $W_s x$ in (6.14) have the same dimension, so they can be summed up by element-wise addition.

The backbone algorithm excludes the output layer of ResNet-50. Since ResNet-50 is originally designed for the image classification task, its output layer contains the softmax activation function that converts the network output into a vector that contains the probability distribution of image categories. In other words, ResNet-50 outputs a vector with each element in [0,1], which is smaller than the codomain of the *sin* or *cos* function (i.e., [-1,1]).

6.4.2.2 CNN for estimating roll angle

The CNN estimates the hand roll angle by a single image, where the roll angle is in the cos-sin representation. Fig. 6.7(a) shows the algorithm structure, which includes the backbone algorithm and three fully connected (FC) layers. The FC layers are used to gradually decrease

the feature dimension, as the backbone algorithm outputs a 2048×1 feature vector but the cos-sin representation is a 2×1 vector. Specifically, the first FC layer decreases the feature dimension to 512×1, while the second FC layer decreases the feature dimension to 128×1. The two FC layers use the ReLU activation function [61]. In comparison, the last FC layer, known as the output layer, decreases the feature dimension to 2×1 but without any activation function.

The two FC layers use batch normalization (BN) [36] before the ReLU activation function. BN is a prevalent technique for accelerating the training process of deep-learning algorithms. It is well-known that training a deep-learning algorithm is challenging because the distribution of each layer's input changes during training [36]. BN tackles the problem by normalizing the linear output of the layers, along the dimension of batch size, so the input to the next layer has a mean approximated to 0 and a variance approximated to 1. Specifically, batch size denotes the number of input samples for a single forward pass. Studies show that using BN allows algorithms to be trained with a high learning rate, which significantly decreases the training time [36].

BN can be used in FC layers and convolutional layers. This work focuses on the implementation of BN on FC layers. Let the input to BN is $\boldsymbol{Z} \in \mathbb{R}^{B\times N}$, or $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, ..., \boldsymbol{z}_B]$, where B denotes the batch size, and N denotes the length of features. The implementation of BN is as follows.

When training the deep-learning algorithms, BN firstly calculates the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$.

$$\boldsymbol{\mu} = \frac{1}{B}\sum_{i=1}^{B} \boldsymbol{z}_i \tag{6.15}$$

$$\boldsymbol{\sigma}^2 = \frac{1}{B}\sum_{i=1}^{B} (\boldsymbol{z}_i - \boldsymbol{\mu})^2 \tag{6.16}$$

Notice that the dimension of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ are $\mathbb{R}^{N\times 1}$. Also, BN requires $B > 1$ during the training process.

Each feature is then normalized by the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$, so the features have a mean that approximates 0 and a variance that approximates 1.

$$\hat{z}_i = \frac{\boldsymbol{z}_i - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2 + \epsilon}} \tag{6.17}$$

where $i \in [1, B]$ and ϵ denotes an arbitrary small number.

The normalized features are then scaled by γ and shifted by β, where γ and β are updated during backpropagation. This operation allows BN to recover the original distribution, as normalization is unnecessary at times.

$$\boldsymbol{y}_i = \gamma \hat{\boldsymbol{z}}_i + \beta \tag{6.18}$$

During the training process, the mean $\boldsymbol{\mu}_m$ and variance $\boldsymbol{\sigma}_m^2$ of the entire dataset are also calculated via exponential moving average. These values, along γ and β, are saved when the training process is complete.

$$\boldsymbol{\mu}_m = \alpha \boldsymbol{\mu}_m + (1 - \alpha)\boldsymbol{\mu} \tag{6.19}$$

$$\boldsymbol{\sigma}_m^2 = \alpha \boldsymbol{\sigma}_m^2 + (1 - \alpha)\boldsymbol{\sigma}^2 \tag{6.20}$$

where α is set to 0.9 in our implementation.

When testing the deep-learning algorithms, the input $\boldsymbol{Z}$ is reduced to $\boldsymbol{Z} \in \mathbb{R}^{1 \times N}$, or $\boldsymbol{Z} = [\boldsymbol{z}_1]$. By using the saved $\boldsymbol{\mu}_m$, $\boldsymbol{\sigma}_m^2$, γ, and β, $\boldsymbol{Z}$ is then normalized via

$$\boldsymbol{y}_1 = \gamma \frac{\boldsymbol{z}_1 - \boldsymbol{\mu}_m}{\sqrt{\boldsymbol{\sigma}_m^2 + \epsilon}} + \beta \tag{6.21}$$

The CNN parameters are configured using the transfer learning (TL) technique, which is for adapting a pre-trained model to accomplish a task with a comparatively small dataset [15]. Since ResNet-50 has been optimized for the large-scale ImageNet dataset [71], it is assumed that its pre-trained parameters have learned the general features of images and can be directly used for our image analysis task. By using the pre-trained ResNet-50 as the backbone algorithm, TL avoids training the hand roll estimation CNNs from scratch, which significantly simplifies the training process.

In this report, the CNN parameters have three configurations, resulting in three CNN models.

- **CNN(FC)**: Only the parameters in the three FC layers are updated during training.
- **CNN(conv+FC)**: In addition to the three FC layers, the training process also updates the parameters in the `conv4` and `conv5` portions in ResNet-50 [33].
- **CNN(step)**: The model is trained as CNN(FC) at the first 5 epochs. The model is then

trained as CNN(conv+FC) at the remaining epochs.

Among the three models, CNN(FC) is used as the baseline for the roll angle estimation accuracy, while CNN(conv+FC) is designed based on the fact that the algorithm performance can be improved by increasing the number of trainable convolutional blocks [86]. In comparison, CNN(step) is designed based on the assumption that training a shallow network is easier than training a deep network [30], so we let the FC layers converge before simultaneously training the FC layers and the convolutional blocks. Notice that the three CNNs update the parameters in the later layers but preserve the parameters in the earlier layers during the training process, as the parameters in earlier layers contain generic features that can be shared among different datasets [15, 86].

6.4.2.3 CRNN for estimating roll angle

The CRNN estimates the hand roll angle based on a sequence of images, where the roll angle is in cos-sin representation. The algorithm structure is shown in Fig. 6.7(b), which includes an LSTM layer and the top portion of the CNN in Fig. 6.7(a). The LSTM layer allows the algorithm to use the previous and the current images to determine the current roll angle, so the algorithm performance is less likely to be affected by problematic images. For example, the estimated roll angles from the CNN have larger fluctuations than those from the CRNN when the suturing hand is out of the camera view. Notice that the CRNN in Fig. 6.7(b) only estimates roll angles, so it is abbreviated as CRNN(angle) hereafter.

The dashed rectangle in Fig. 6.7(b) shows that dropout [85] is used between the FC layer and the output layer. Dropout is a technique to avoid overfitting, which means an algorithm has high performance on the training data but not on the new data. To implement dropout, the neurons in the FC layer are randomly "turned off" with probability p during training, where $p = 0.5$ in our settings. At test time, however, all neurons in the FC layer are used with the weight scaled as $W_{test} = pW_{train}$. Since our result shows that CRNN(angle) with dropout outperforms CRNN(angle) with BN, and it is suggested not to use dropout and BN simultaneously [36], BN is not used between the FC layer and the output layer in CRNN(angle).

Fig. 6.9 compares the structure of the standard recurrent neural network (RNN) neuron (see Fig. 6.9(a)) and the structure of the long short-term memory (LSTM) neuron (see Fig. 6.9(b)). Both standard RNN neurons and LSTM neurons can construct recurrent neural networks for processing

time-series data, but LSTM neurons overcome the long-term dependency problem, which refers to the vanished gradient when there is a large gap between related information [7]. To tackle the problem, the LSTM neuron uses the cell state c_t (see Fig. 6.9(b)) rather than the hidden state h_t (see Fig. 6.9(a)) to transfer temporal information.

An LSTM neuron includes 3 gates: the forget gate, the input gate, and the output gate. The forget gate determines what information should be removed from c_t.

$$f_t = \sigma(W_{fh}h_{t-1} + W_{fx}x_t + b_f) \tag{6.22}$$

where σ, W, and b_f denote the sigmoid function, the weight matrices, and the bias, respectively.

The input gate determines what information should be added to c_t.

$$i_t = \sigma(W_{ih}h_{t-1} + W_{ix}x_t + b_i) \tag{6.23}$$

$$\tilde{c}_t = \tanh(W_{\tilde{c}h}h_{t-1} + W_{\tilde{c}x}x_t + b_{\tilde{c}}) \tag{6.24}$$

After the forget gate and the input gate, the cell state c_t becomes

$$c_t = f_t \cdot c_{t-1} + i_t \cdot \tilde{c}_t \tag{6.25}$$

where the operator $\cdot$ denotes the element-wise multiplication.

Finally, the output gate determines what information should be output based on c_t.

$$o_t = \sigma(W_{oh}h_{t-1} + W_{ox}x_t + b_o) \tag{6.26}$$

$$h_t = o_t \cdot \tanh(c_t) \tag{6.27}$$

By comparing Fig. 6.9 (a) and (b), it is noticeable that the configuration of the LSTM neuron avoids the temporal information being directly modified by input $x(t)$, so the temporal information can be transferred for a long period.

As can be seen from Fig. 6.9(c), time step T denotes the maximum duration of the cell state and the hidden state. In other words, the LSTM neuron at T cannot transfer the cell state and hidden state to the LSTM neuron at $T + 1$. Notice that the time step is synonymous with the length of the input sequence when implementing LSTM in Pytorch.

Recall that CRNN requires the input to be a sequence of images, so the sliding window approach is used to generate the CRNN input sequence. Specifically, the first input sequence is the first T video frames (i.e., $X = [x_1, x_2, ..., x_T]$), while the second input sequence is from the second to the $T + 1$ video frame (i.e., $X = [x_2, x_3, ..., x_{T+1}]$). A special case occurs when transferring from one suture to the next one. Recall that the needle entry and exit time are used when generating the dataset, so the dataset excludes the video frames between the adjacent sutures. Thus, the sequence generated by the sliding window approach is discontinuous when going across sutures. To tackle the problem, the video frames from the previous suture are replaced by zero matrices if the input sequence includes images from two adjacent sutures. For instance, if the input sequence includes $T - 1$ frames for the first suture and the first frame of the second suture, the input sequence will become $X = [\mathbf{0}, \mathbf{0}, ..., x_1]$.

CRNN (angle) has two configurations, namely the many-to-many CRNN and the many-to-one CRNN. The many-to-many CRNN [27] receives a sequence of video frames and then estimates the hand roll angle per video frame. In comparison, the many-to-one CRNN [27] receives a sequence of video frames but only estimates the hand roll angle at the last video frame. To accelerate the training process, CRNN(angle) uses the parameters of CNN(conv+FC) and only updates the parameters of the top layers (see the dashed rectangle in Fig. 6.7(b)) during training.

6.4.2.4 CRNN for estimating roll angle and roll velocity

Shayan et al. [76] distinguish participants' surgical suturing skills by analyzing the hand roll velocity measured by IMU, which motivates us to design a video analysis algorithm to estimate hand roll velocity. Also, studies show that relevant information can be simultaneously estimated by a single deep-learning algorithm [68, 66]. Since CRNN(angle) supports estimating roll angles and the first-order derivative of roll angle is roll velocity, a new branch is added to CRNN(angle), so the modified algorithm can simultaneously estimate roll angle and roll velocity (see Fig. 6.7(c)). Since the algorithm estimates both roll angle and roll velocity, it is abbreviated as CRNN(angle & vel) hereafter.

As shown in Fig. 6.7(c), the two output branches of CRNN(angle & vel) have similar configurations, but the branch estimating roll velocity has one more FC layer than the branch estimating roll angle. The extra FC layer is used to guarantee the gradual decrease of the feature dimension for roll velocity estimation, as the output dimension of the velocity-estimation branch

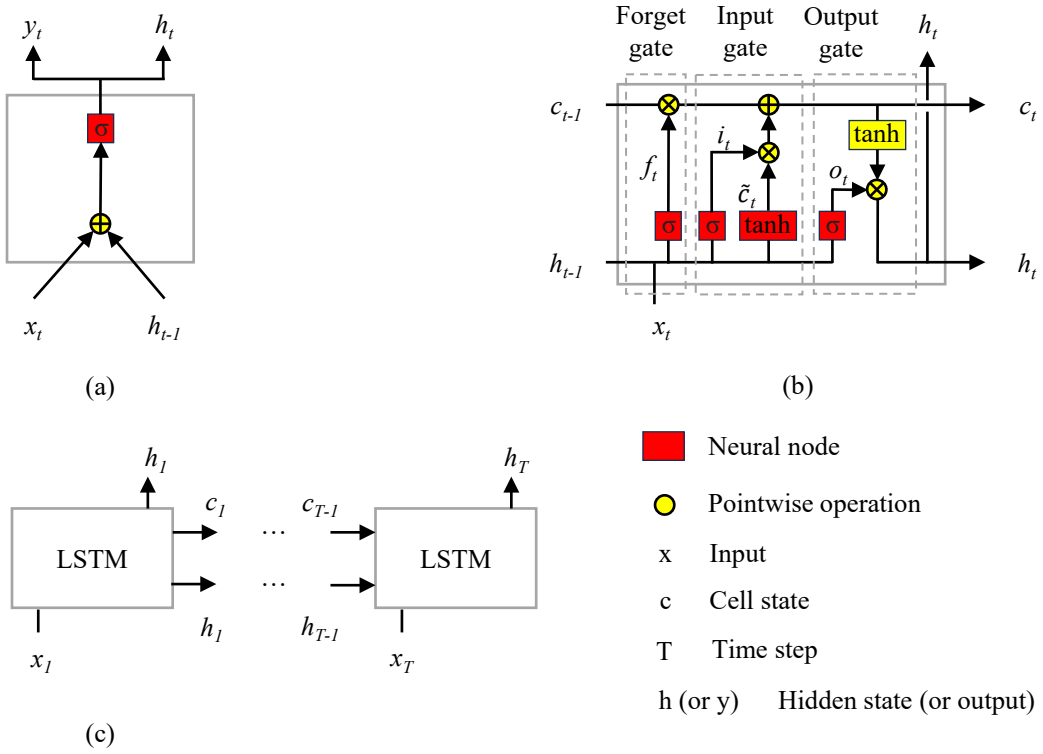


Figure 6.9: Comparing the standard RNN neuron and the LSTM neuron. (a) A standard RNN neuron; (b) An LSTM neuron; (c) An illustration of T time steps. The sub-figures (a) and (b) are from [105].

(i.e., a scalar) is smaller than the output dimension of the angle-estimation branch (i.e., a 2×1 vector).

Like CRNN(angle), CRNN(angle & vel) has the many-to-many and the many-to-one configuration. Also, CRNN(angle & vel) uses the parameters of CNN(conv+FC) and only updates the parameters of the top layers (see the dashed rectangle in Fig. 6.7(c)) during training.

6.4.3 Data pre-processing and hyperparameters

The training images are pre-processed by the following data augmentation techniques, which aim to increase variation in the training data.

- Random shift in brightness in range [0.7, 1.2].
- Random shift in contrast in range [0.7, 1.2].
- Random shift in saturation in range [0.8, 1.3].

- Random cropping the original image. The cropped image is 70-100% as large as the original one.

After data augmentation, each image is processed based on the requirements of Pytorch pre-trained ResNet-50. That is, the images are scaled such that both the height and the width are equal to 224. Also, the RGB channels are normalized by mean=$[0.485, 0.456, 0.406]$ and standard deviation=$[0.229, 0.224, 0.225]$, where the mean and standard deviation (STD) are calculated based on the ImageNet dataset [71].

The normalized images are shuffled before the training process, which aims to prevent the hand roll estimation algorithms from overfitting to specific patterns. Specifically, the order of images is shuffled after each epoch when training the CNNs. For training the CRNNs, however, the order of image sequences is shuffled but the order within image sequences is preserved after each epoch, as the LSTM requires the temporal information within the image sequences.

Also, the ground-truth roll velocity is normalized before training the hand roll estimation algorithms. Our experimental results show that the normalized roll velocity is beneficial to improving the roll velocity estimation accuracy. The normalization process uses mean=0.071 and STD=1.302, which are calculated from the hand roll estimation dataset.

The following hyperparameters are used during the training process. The learning rate was set to $1e^{-4}$ for the Adam optimizer [52], while the number of epochs was set to 20, chosen empirically. For CRNN, time step was set to 5. When training each algorithm, we only saved the parameters corresponding to the smallest errors in the validation set, which is calculated by mean absolute error(MAE).

6.4.4 Evaluating algorithm performance

After training the hand roll estimation algorithms, their performance is examined by the test set and the 12 videos excluded from the hand roll estimation dataset. The test set provides an independent evaluation of the algorithm [72], but we argue that videos provide a better estimation of the algorithm's actual performance. Since the test set is a portion of the hand roll estimation dataset, the algorithms have seen images similar to the test set during training, so the algorithms might have smaller errors on the test set than on the videos. Also, the test set only allows us to calculate MAE, but the videos allow us to calculate MAE and analyze estimation errors for specific

hand motion.

The 12 videos are collected by 2 SutureCoach in the SCVS conference. Specifically, 6 videos are from Location D (i.e., the location collecting the hand roll estimation dataset), while the remaining 6 videos are from Location C (i.e., the location excluded from the hand roll estimation dataset). Compared with test videos (Location D), test videos (Location C) are assumed to be more challenging for the hand roll estimation algorithms, as the video frames in test videos (Location C) have different brightness from the hand roll estimation dataset. Also, some participants are in both the hand roll estimation dataset and test videos (Location D), but the participants in the hand roll estimation dataset are excluded from test videos (Location C). Since the hand roll estimation algorithms have seen the participants' motion patterns during training, it is assumed that the algorithms have smaller estimation errors on test videos (Location D) than on test videos (Location C). Thus, test videos (Location C) help to estimate the robustness of the algorithms.

To visualize the instantaneous errors of the videos, a custom Python program is used to synchronize the video frames, the estimated values, and the ground-truth (GT) values. The synchronized data is beneficial to revealing which hand motion contributes to large estimation errors. Fig. 6.10 demonstrates the interface for inspecting instantaneous errors of CRNN(angle & vel)(many-to-one) (i.e., MO) and CRNN(angle & vel)(many-to-many) (i.e., MM). Specifically, the blue dashed rectangle includes the current video frame, in which the corresponding values are denoted by the magenta vertical line in the right plots. The blue dashed rectangle also includes a green bounding box that denotes the cropped image received by the hand roll estimation algorithms. The right plots show the ground-truth and the estimated values for a single suture. The average errors for that suture are shown in the red dashed rectangle.

6.5 Hand-roll metrics for surgical suturing skill assessment

Shayan et. al. [76] showed that hand-roll metrics named *Number of rolls*, *Average roll angle*, *Log dimensionless jerk*, *Spectral arc length*, and *Median roll velocity* had significantly different means among participants with different surgical suturing skill levels, where the metrics were calculated by integrating the IMU roll velocity. Since the integration of roll velocity is closely related to roll angle, it is hypothesized that those metrics based on roll angles also have different means among participants with different skill levels. For simplicity, *Number of rolls*, *Average roll angle*, *Log dimensionless jerk*,

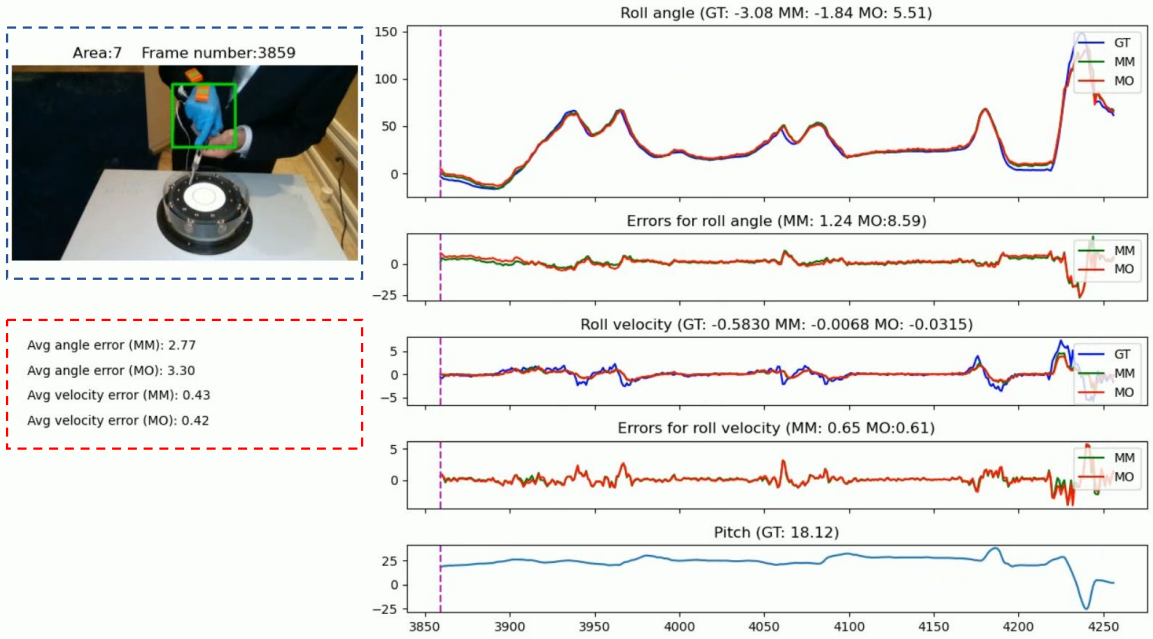


Figure 6.10: Interface for inspecting instantaneous errors. GT, MO, and MM denote ground-truth data, CRNN(angle & vel)(many-to-one), and CRNN(angle & vel)(many-to-many), respectively.

Spectral arc length, and *Median roll velocity* are abbreviated as `Number of rolls`, `Average roll`, `LDLJ`, `SPARC`, and `Median vel`, respectively.

The hand-roll metrics can be classified into 2 categories.

- Roll angle metrics: `Number of rolls` and `Average roll`.
- Roll velocity metrics: `LDLJ`, `SPARC`, and `Median vel`.

The roll angle metrics are free from the derivative calculation (see Fig. 6.11(a)), while the roll velocity metrics require taking derivatives of roll angles to obtain roll velocity (see Fig. 6.11(b)).

To examine if the algorithm output can be used for surgical suturing skill assessment, the hand roll angle estimation algorithm that has the smallest errors on test videos (Location C) is used to process the data collected by SutureCoach. The algorithm output is then used to calculate the hand-roll metrics. It is expected that the hand-roll metrics based on algorithm output have similar statistics as the hand-roll metrics based on ground-truth roll angles (see Fig. 6.4) if the algorithm output and ground-truth roll angles have similar patterns.

Since the ground-truth roll angles and the algorithm output contain different amounts of noise, Savitzky-Golay filters with various settings are used to pre-process the roll angles before

metric calculation (see Fig. 6.11). Specifically, Fig. 6.11(a) is the pre-processing technique for the roll angle metrics. Notice that the Savitzky-Golay filter is only applied to the algorithm output. In comparison, Fig. 6.11(b) is the pre-processing technique for the roll velocity metrics. After data pre-processing, ground-truth roll angles and the algorithm output use identical formulas for metric calculation.

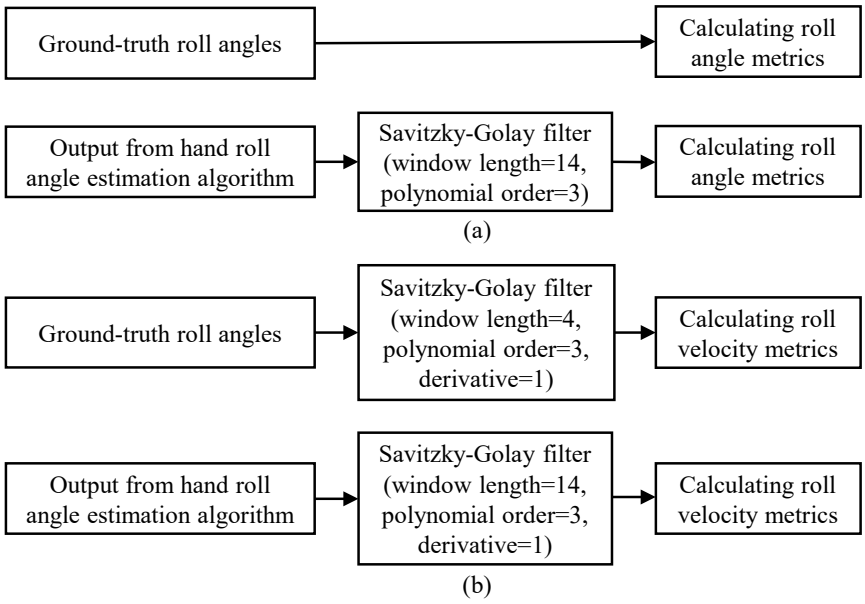


Figure 6.11: Data pre-processing for (a) roll angle metrics and (b) roll velocity metrics.

6.5.1 Roll angle metrics

`Number of rolls` is for counting the number of rotational motions in the hand roll direction during a single suture. It is assumed that `Number of rolls` decreases as suturing skill increases, as competent surgeons have fewer unnecessary hand rotations than medical students [76]. To calculate `Number of rolls`, the Scipy [92] *find_peaks* function with *prominence=1* and *width=2* is used to detect local minima and local maxima in the time-series roll angle signal. The resulting number of minima and maxima is `Number of rolls`.

`Average roll` is the mean of local minima and local maxima in the time-series roll angle signal during a single suture. Since the radial suturing task is dominated by rotational motion and a skillful participant has large magnitude per rotational motion, it is assumed that `Average roll` is proportional to participants' suturing skill levels [76]. To calculate `Average roll`, the Scipy

[92] *find_peaks* function with *prominence=1* and *width=2* is used to detect local minima and local maxima in the time-series roll angle signal. The local minima and local maxima are then averaged by

$$\texttt{Average roll} = \frac{\sum |\theta_{min}| + \sum |\theta_{max}|}{N} \tag{6.28}$$

where θ_{min} are the local minima and θ_{max} are the local maxima in the time-series roll angle signal for a single suture. N denotes the number of local minima and local maxima in that suture.

6.5.2 Roll velocity metrics

The roll velocity metrics involve the derivative of roll angles, which is calculated using the Scipy [92] *savgol_filter* function (see Fig. 6.11(b)). Since the padding used by the *savgol_filter* function causes artifacts at the start and end of the derived roll velocity, the first and last *window length* of the velocity are removed before calculating the roll velocity metrics.

`LDLJ` measures the intermittency in motion, which can be caused by the lack of controlled movement [82]. Thus, `LDLJ` quantifies motion smoothness. The core of `LDLJ` is the third derivative of roll angles (i.e., jerk), which is an indication of motion intermittency. Since jerk is integrated in `LDLJ` calculation, the metric is sensitive to the duration and amplitude of motion. Thus, the integration of jerk is multiplied by a scalar (i.e., $\frac{D^3}{v_{peak}^2}$) to make the result dimensionless [34]. Finally, the natural log is to eliminate the wide variability of the results [5]. Since medical students have more motion intermittency than surgeons and results in a small `LDLJ` value, `LDLJ` is assumed to be proportional to the suturing skills. `LDLJ` is calculated by

$$\texttt{LDLJ} = -ln\left[\frac{D^3}{v_{peak}^2}\int_{t_1}^{t_2} \dddot{\theta}(t)^2 dt\right] \tag{6.29}$$

where $D = t_2 - t_1$ is duration, v_{peak} is peak velocity, and θ is roll angles of a suture [34, 76].

`SPARC` is another metric measuring motion smoothness. The metric calculates the arc length, from 0 to the cut-off frequency ω_c, in the Fourier magnitude spectrum of $v(t)$ [5]. Since unsmooth motion contributes to large magnitudes in the Fourier magnitude spectrum and results in a small

SPARC value, SPARC is assumed to be proportional to the suturing skills. SPARC is calculated by

$$\texttt{SPARC} = -\int_0^{\omega_c} \left[\left(\frac{1}{\omega_c} \right)^2 + \left(\frac{d\hat{V}(\omega)}{d\omega} \right)^2 \right]^{\frac{1}{2}} d\omega; \quad \hat{V}(\omega) = \frac{V(\omega)}{V(0)} \tag{6.30}$$
$$\omega_c = min \left\{ \omega_c^{max},\ min \left\{ \omega,\ \hat{V}(r) < \overline{V}\ \forall\ r > \omega \right\} \right\}$$

where $V(\omega)$ is the Fourier magnitude spectrum of $v(t)$. $V(0)$ is the DC value of the spectrum. $v(t)$ is the first-order derivative of the time-series roll angle for a single suture. It is recommended using $\overline{V} = 0.05$ and $\omega_c^{max} = 20\pi$ [5].

Median vel examines if experienced surgeons have high angular velocity or use slow controlled roll motion during suturing [76]. Median vel is calculated by

$$\texttt{Median vel} = median(|v(t)|) \tag{6.31}$$

where $v(t)$ is the first-order derivative of the time-series roll angle for a single suture.

Chapter 7

Results and discussion for suturing skill assessment via hand motion

7.1 Results of hand detection

7.1.1 Evaluation metrics for hand detection

Table 7.1 shows the precision, recall, mAP@0.5, and mAP@0.95 of the hand detection algorithm at the test set.

Precision	Recall	mAP@0.5	mAP@0.95
1.00	1.00	1.00	0.99

Table 7.1: Evaluating YOLOv7 performance

As shown in Table 7.1, the hand detection algorithm has high detection accuracy. One reason is the hand detection algorithm is based on the fine-tuned YOLOv7, which is capable of detecting multiple objects in a single image. In comparison, the hand detection task is comparatively simple, as there is only one target per video frame.

7.1.2 Visualizing hand detection results

Fig. 7.1 shows the hand detection results of the fine-tuned YOLOv7, where the results are highlighted by blue bounding boxes overlaid with the object name (i.e., dominant hand) and the

confidence.

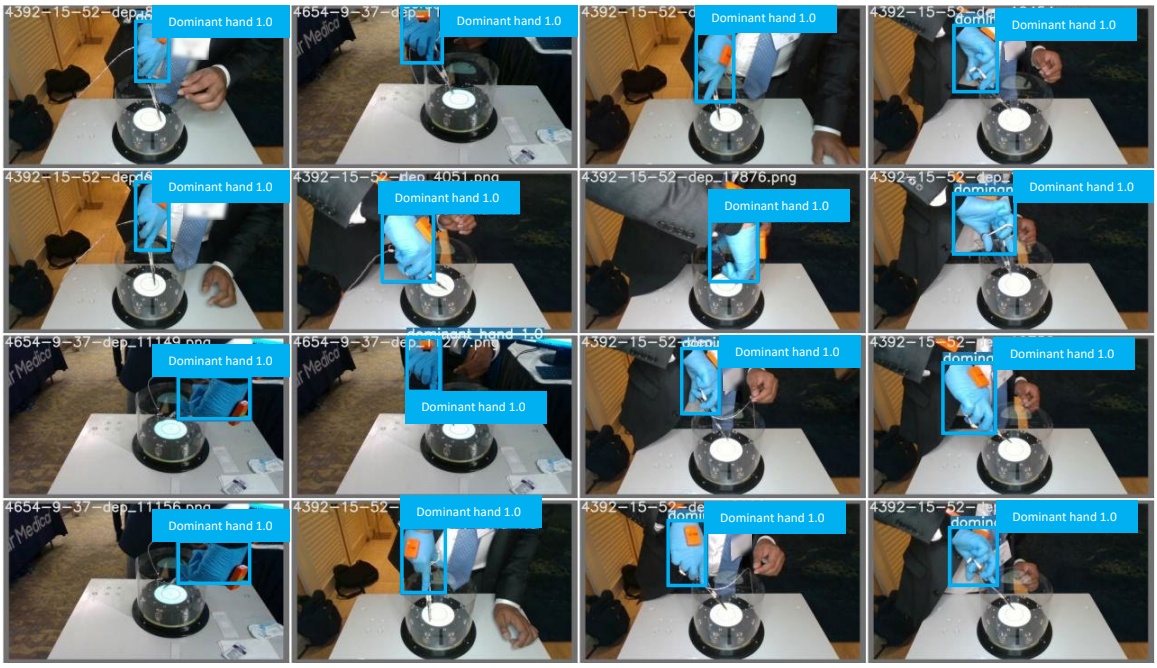


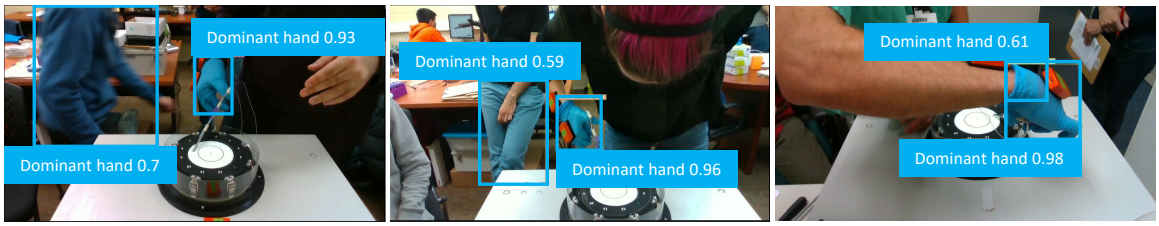
Figure 7.1: Output of the hand detection algorithm. The results are denoted by the blue bounding boxes, where each bounding box is overlaid with the object name (i.e., dominant hand) and the confidence.

From Fig. 7.1, we can notice that the hand detection algorithm can precisely detect participants' dominant hands in images with different backgrounds. Also, it shows that the detection accuracy is insensitive to the hand orientation.

7.1.3 Visualizing false detection of hands

Fig. 7.2 shows some false detections of the fine-tuned YOLOv7, which involves blue objects irrelevant to the blue gloves (see Fig. 7.2(a) and (b)) and overlapped detections (see Fig. 7.2(c)). The detections are highlighted by blue bounding boxes overlaid with the object name (i.e., dominant hand) and the confidence.

As shown in Fig. 7.2, the fine-tuned YOLOv7 algorithms might falsely determine blue hoodies (see Fig. 7.2(a)) or blue jeans (see Fig. 7.2(b)) as the dominant hand. One reason is those objects have similar colors to the blue glove. Also, the folds on the clothes look similar to the fingers on the glove. Differ from 7.2(a) and (b), Fig. 7.2(c) shows that some detections are overlapped. By comparing the confidence of the detections, we can notice that participants' dominant hands have a



(a) (b) (c)

Figure 7.2: False detection from the hand detection algorithm. The detection results are denoted by the blue bounding boxes, where each bounding box is overlaid with the object name (i.e., dominant hand) and confidence.

larger confidence value than other blue objects or the overlapped detections.

After processing the videos for total sutures (see Table 6.1), there are 1.1% images that have false or overlapped detections. Under the circumstances, the detection with the largest confidence will be considered as the dominant hand.

7.2 Results of hand roll angle and velocity estimation

7.2.1 Comparing the sizes of algorithms

Table 7.2 shows the number of parameters in the hand roll estimation algorithms. Since the algorithms are trained via the transfer learning techniques, only a portion of the parameters (i.e., the trainable parameters) are updated during the training process.

	CNN (FC)	CNN (conv+FC)	CNN (step)	CRNN (angle) (many-to-one)	CRNN (angle) (many-to-many)	CRNN (angle & vel) (many-to-one)	CRNN (angle & vel) (many-to-many)
parameters	24,624K	24,624K	24,624K	24,646K	24,646K	24,647K	24,647K
trainable parameters	1,116K	23,179K	23,179K	22K	22K	23K	23K

Table 7.2: Comparing the size of algorithms

As shown in Table 7.2, the seven hand roll estimation algorithms have similar numbers of parameters because the backbone network is the major portion of those algorithms. The number of trainable parameters, however, is significantly larger in the CNNs than in the CRNNs. Recall that the CRNNs include the parameters in CNN(conv+FC), which are non-trainable when training the CRNNs.

Notice that CRNN (many-to-one) and CRNN (many-to-many) have the same number of

parameters even though CRNN (many-to-one) has fewer outputs than CRNN (many-to-many). Since CRNN (many-to-many) uses identical output branches at each time step, increasing the number of outputs does not increase the parameters of CRNN (many-to-many).

7.2.2 Results and discussion of hand roll angle estimation on test data

7.2.2.1 Performance on the test set

Table 7.3 shows the mean absolute errors (MAE) of the hand roll estimation algorithms when processing the test set. Specifically, the second row in Table 7.3 is the MAE in the cos-sin representation, which is the original algorithm output. In comparison, the third row in Table 7.3 is the MAE in the Euler angle representation, which is converted from the original algorithm output and constrained in $[-pi, pi]$ (see (6.12) in Subsection 6.4.1.2). Since the Euler angle representation provides an intuitive interpretation of roll angle estimation accuracy, the roll angle estimation error is reported in the Euler angle representation hereafter. Table 7.3 also shows the roll velocity estimation error of CRNN (angle & vel) when processing the test set.

	CNN (FC)	CNN (conv+FC)	CNN (step)	CRNN (angle) (many-to-one)	CRNN (angle) (many-to-many)	CRNN (angle & vel) (many-to-one)	CRNN (angle & vel) (many-to-many)
Test error (Roll angle) (cos-sin)	0.07	**0.02**	0.02	0.04	0.04	0.05	0.05
Test error (Roll angle) (Unit: degree)	5.89	**1.97**	2.15	2.86	3.08	3.63	3.28
Test error (Roll velocity) (Unit: rad/s)	N/A	N/A	N/A	N/A	N/A	0.54	0.52

Table 7.3: Algorithm performance on the test set. The smallest roll angle estimation error is highlighted in bold

Table 7.3 shows that CNNs have smaller errors than CRNNs when estimating the roll angle in the test set. Among the three CNNs, CNN (conv+FC) has the smallest roll angle estimation error. For the four CRNNs, CRNN(angle)(many-to-one) has the smallest roll angle estimation error. Notice that increasing the number of outputs from many-to-one to many-to-many does not have a noticeable effect on the CRNN test errors. Also, Table 7.3 shows that CRNN(angle & vel)(many-to-one) and CRNN(angle & vel)(many-to-many) have similar errors for roll velocity estimation.

7.2.2.2 Performance on the test videos (Location D)

Table 7.4 summarizes the MAE of roll angle estimation for the six videos, which are from the location collecting the hand roll estimation dataset (i.e., Location D). The six videos include two videos from two medical students, two videos from a resident surgeon, and two videos from an attending surgeon. For each of the two videos, one records suturing at the surface condition (see Fig. 2.3(d)), while the other records suturing at the depth condition (see Fig. 2.3(e)). Each number in Table 7.4 is the average error of the 12 sutures in a single video. Also, Table 7.4 shows the average error of the six videos in the far-right column. In each column of the table, the smallest value is highlighted in bold.

The six videos in Table 7.4 are excluded from the hand roll estimation dataset. but the participants in Table 7.4 have been included in the hand roll estimation dataset. This is because each participant has 4 videos, so a participant might have videos in the hand roll estimation dataset and have videos in Table 7.4.

	Student (surface)	Student (depth)	Resident (surface)	Resident (depth)	Attending (surface)	Attending (depth)	Average
CNN (FC)	16.7	9.37	8.24	11.59	11.51	9.01	11.07
CNN (conv+FC)	7.14	4.18	4.9	5.19	7.52	6.15	5.85
CNN (step)	**7.05**	4.27	**4.6**	**5.1**	7.16	5.2	**5.56**
CRNN (angle) (many-to-one)	7.56	4.26	4.8	6.17	5.72	4.82	**5.56**
CRNN (angle) (many-to-many)	7.82	4.24	6.16	6.99	5.51	**4.43**	5.86
CRNN (angle & vel) (many-to-one)	7.94	**3.95**	6.09	7.39	5.58	4.74	5.95
CRNN (angle & vel) (many-to-many)	8.11	4.21	4.87	7	**5.39**	4.83	5.74

Table 7.4: MAE for roll angle estimation (Unit: degree). The six videos are from Location D, which is for collecting the hand roll estimation dataset. For each column, the smallest number is highlighted in bold.

By comparing the errors in the test videos from Location D (Table 7.4) and the errors in the

test set (Table 7.3), it is noticeable that the algorithms have larger errors in the test videos. Since the test set and the training set are generated from the same videos, the test set and the training set have similar images, which consequently makes the algorithms have high performance on the test set. From the far-right column at Table 7.4, it is noticeable that CNN(step) and CRNN(angle)(many-to-one) have the smallest roll angle estimation errors, followed by CRNN(angle & vel)(many-to-many). Also, Table 7.4 shows that all algorithms have the largest error when processing the video of the medical student suturing at the surface condition, as the participant was unfamiliar with the suturing procedure and had abnormal hand motion during the suturing task.

7.2.2.3 Performance on the test videos (Location C)

Table 7.5 shows the MAE of roll angle estimation for the six videos from Location C, which is not for collecting the hand roll estimation dataset. The six videos include two videos from a medical student, two videos from a resident surgeon, and two videos from an attending surgeon. For each of the two videos, one records suturing at the surface condition (see Fig. 2.3(d)), while the other records suturing at the depth condition (see Fig. 2.3(e)). Notice that Table 7.5 and Table 7.4 arrange the rows and columns in the same order. Table 7.5 also averages the errors of 12 sutures in each video and highlights the smallest value in each column in bold.

The six videos in Table 7.5 are excluded from the hand roll estimation dataset. Also, participants for the six videos are different from participants in the hand roll estimation dataset.

Compared with the errors in the test set (Table 7.3) and the errors in the test videos from Location D (Table 7.4), the algorithms have the largest estimation errors in the test videos at Location C (Table 7.5). Since the videos in Table 7.5 are not from the location for collecting the hand roll estimation dataset, the results in Table 7.5 indicate the robustness of the algorithms. Specifically, the far-right column in Table 7.5 shows that CNN(conv+FC) has the smallest roll angle estimation errors, followed by CNN(step) and CRNN(angle)(many-to-one). Also, it should be noticed that 6 out of the 7 algorithms have the largest errors when processing the video of the resident surgeon suturing at the depth condition. In the video, the resident surgeon's hand is at the vertical direction (see Fig. 7.5) multiple times, causing the IMU to have gimbal lock and to have inaccurate roll angle measurement. This phenomenon will be explained in detail later. Like Table 7.3 and Table 7.4, the roll estimation errors in Table 7.5 increase slightly when adding the roll velocity output branch to the CRNNs.

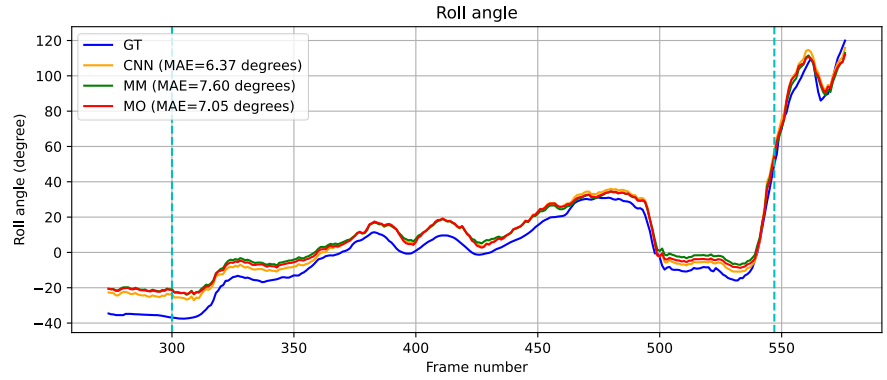
	Student (surface)	Student (depth)	Resident (surface)	Resident (depth)	Attending (surface)	Attending (depth)	Average
CNN (FC)	10.79	9.79	17.47	16.79	9.43	15.75	13.34
CNN (conv+FC)	4.92	3.7	8.06	**10.42**	5.98	5.43	**6.42**
CNN (step)	5.16	**3.59**	**7.96**	10.51	6.55	5.35	6.52
CRNN (angle) (many-to-one)	4.98	3.88	8.43	12.12	**5.58**	5.27	6.71
CRNN (angle) (many-to-many)	**4.79**	4.12	8.19	12.41	5.8	**4.96**	6.71
CRNN (angle & vel) (many-to-one)	5.21	4.1	9.51	13.23	6.56	6.07	7.45
CRNN (angle & vel) (many-to-many)	4.96	3.94	9.57	13.69	5.91	5.48	7.26

Table 7.5: MAE for roll angle estimation (Unit: degree). The six videos are from Location C, which is NOT for collecting the hand roll estimation dataset. For each column, the smallest value is highlighted in bold.

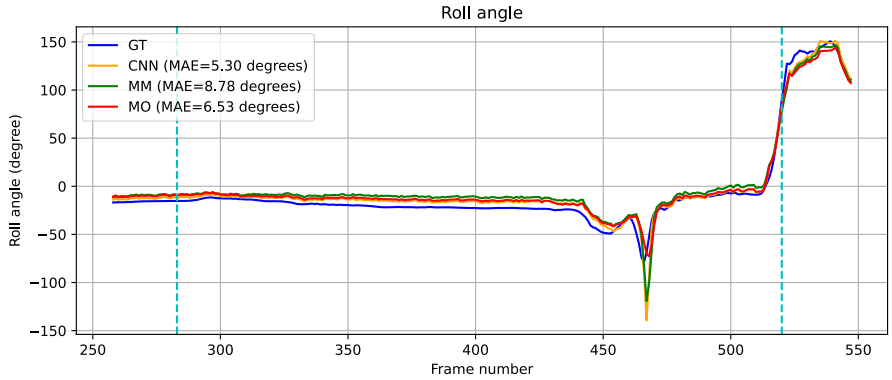
7.2.2.4 Visualizing roll angle estimation accuracy

Fig. 7.3 compares the ground-truth (GT) roll angles and the algorithm output for two sutures in a frame-by-frame manner. In Fig. 7.3, CNN, MM, and MO denote the outputs of CNN(conv+FC), CRNN(angle)(many-to-many), and CRNN(angle)(many-to-one), respectively. Also, the legend shows the MAE of the algorithms at the suture, while the cyan vertical lines refer to the needle entry/exit time at the suture. Notice that Fig. 7.3(a) is for the attending surgeon's first suture at the surface condition. The corresponding video is used in the sixth column in Table 7.5. In comparison, Fig. 7.3(b) is for the resident surgeon's first suture at the depth condition. The corresponding video is used in the fifth column in Table 7.5.

As shown in Fig. 7.3, the ground-truth roll angles have the same pattern as the outputs of the three algorithms, indicating that the hand roll angle estimation algorithms are capable of estimating the hand roll angle. Compared with CRNN(angle)(many-to-many) and CRNN(angle)(many-to-one), CNN(conv+FC) has less estimation error in Fig. 7.3. It should also be noted that there exists a gap between the algorithm outputs and the ground-truth data, and the magnitude of the gap



(a)



(b)

Figure 7.3: Hand roll angle estimation for (a) a suture at the surface condition and (b) a suture at the depth condition. GT, CNN, MM, and MO refer to ground-truth data, CNN(conv+FC), CRNN(angle)(many-to-many), and CRNN(angle)(many-to-one), respectively. MAE for each algorithm is shown in the legend, while the cyan vertical lines denote the frame numbers corresponding to the needle entry/exit time.

changes over time. This bias, rather than frame-to-frame noise, is the majority of the hand roll angle estimation error. One reason for bias is the camera orientation varied slightly between trials and locations, which limits the algorithms from learning precise hand roll angles. For future data collection, the camera orientation will be rigidly fixed.

7.2.2.5 Visualizing hand motion causing large roll angle estimation errors

Fig. 7.4 and 7.5 demonstrate two situations causing the algorithms to have large roll angle estimation errors. Specifically, Fig. 7.4 shows the estimation results when the participant's suturing hand is partially out of the camera view, which happens at the third suture of the attending surgeon's surface video (i.e., the video for the sixth column in Table 7.5). In comparison, Fig. 7.5 shows

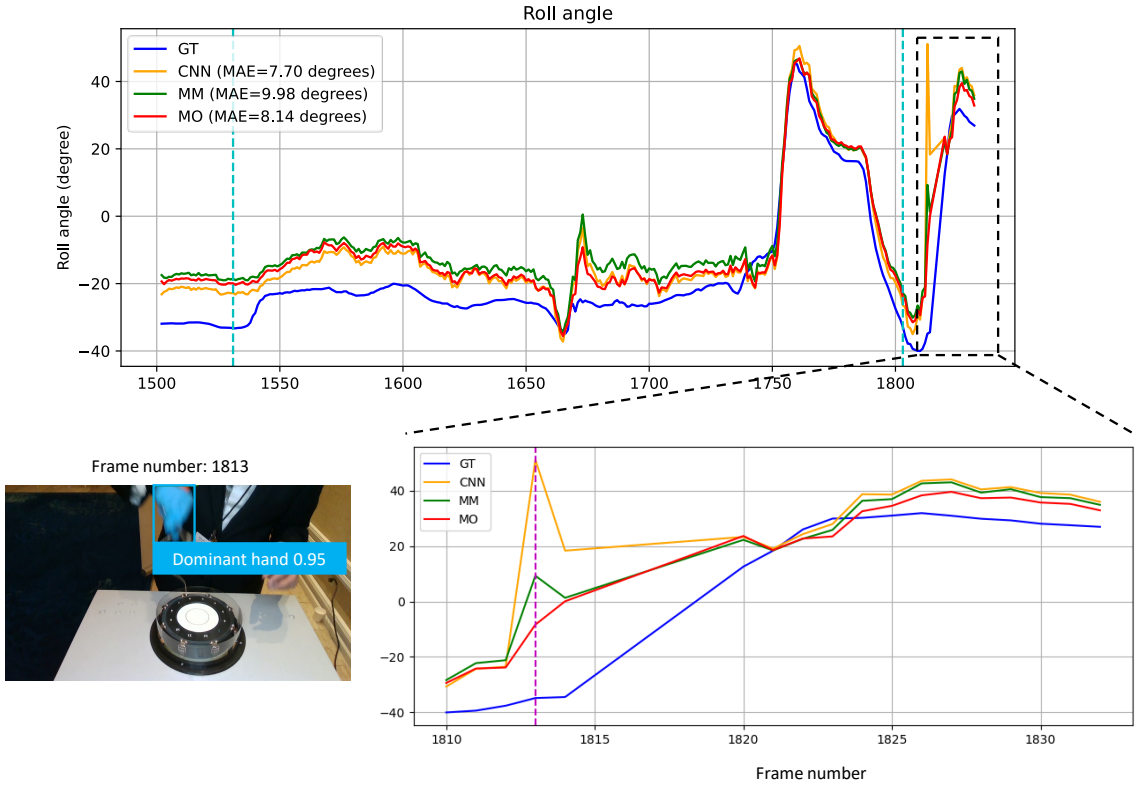


Figure 7.4: Hand roll angle estimation when the suturing hand is partially out of the camera view. The magenta vertical line denotes the frame number for the video frame shown on the left, while the cyan vertical lines denote the needle entry/exit time. The hand detection result of YOLOv7 is denoted by a blue rectangle in the video frame.

estimation results when gimbal lock occurs, which happens at the eleventh suture of the resident surgeon's depth video (i.e., the video for the fifth column in Table 7.5). To verify the reason for large estimation errors, Fig. 7.4 and 7.5 show YOLOv7 hand detection results by a blue rectangle in the video frame. The estimated roll angles corresponding to the video frame are highlighted by the magenta vertical line. Note that Fig. 7.4 and 7.5 use the same notations as in Fig. 7.3.

As shown in Fig. 7.4, CRNN(angle)(many-to-many) and CRNN(angle)(many-to-one) have smaller estimation errors than CNN (conv+FC) when the video frame only includes a portion of the dominant hand. Thanks to the LSTM layer, CRNNs estimate the current roll angle based on the current and the previous video frames, so they are less likely to be affected by problematic images. In contrast, CNNs estimate the current roll angle based on the current video frame only, so they are sensitive to the image quality. The out-of-view images, however, are negligible when calculating the motion-based metrics for suturing skill assessment [76], as the metric calculation ends when the

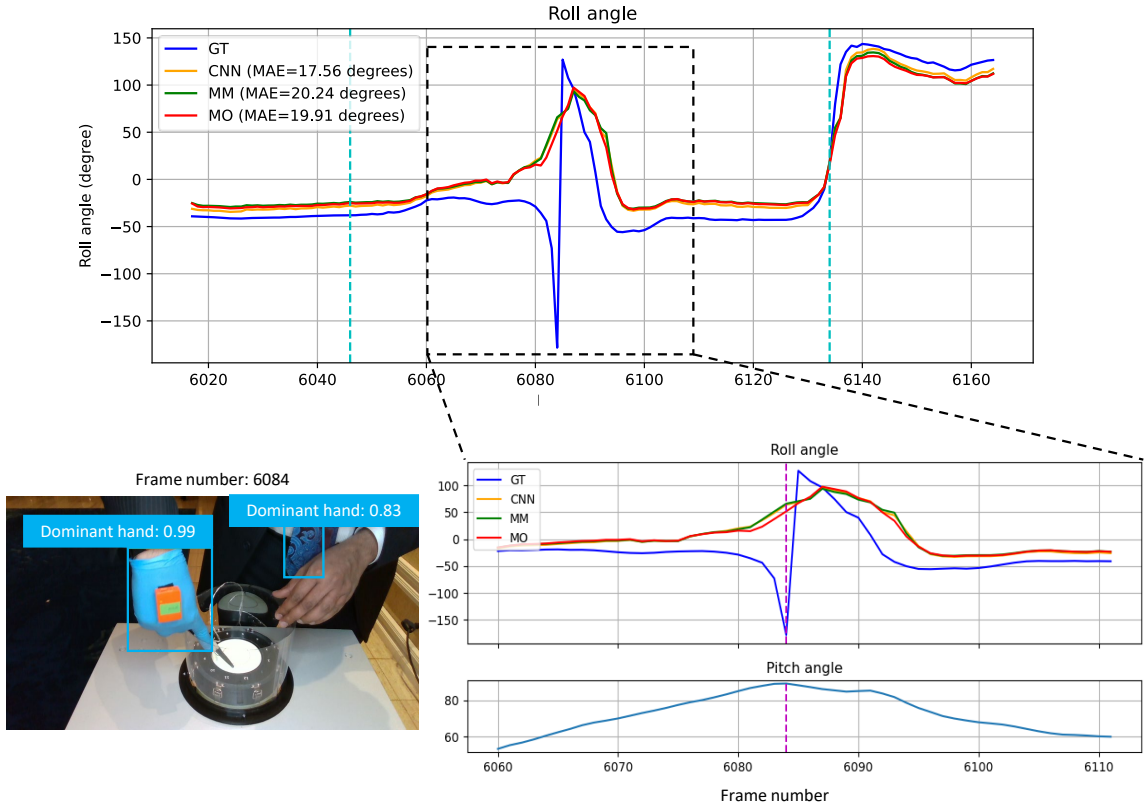


Figure 7.5: Hand roll angle estimation when gimbal lock occurs. The magenta vertical line denotes the frame number for the video frame shown on the left, while the cyan vertical lines denote the needle entry/exit time. The hand detection result of YOLOv7 is denoted by blue rectangles in the video frame.

needle leaves the membrane. By inspecting Fig. 7.4, it is noticeable that the out-of-view image happens after the cyan vertical line, which denotes the instant of needle leaving the membrane.

Fig. 7.5 compares the ground-truth and the estimated hand roll angle when gimbal lock occurs. In the IMU configuration, gimbal lock occurs when roll axis aligns with yaw axis, or pitch $\approx 90°$. From the video frame, it is noticeable that the participant's dominant hand is pointing downward (i.e., pitch $\approx 90°$ shown in the pitch angle figure). Since the IMU roll and yaw axes are aligned at gimbal lock, the roll and yaw angles are coupled. Specifically, at the time indicated by the magenta line, the IMU roll angle is approximately $-150°$, but $-150°$ represents the participant's palm facing upward when there is no gimbal lock. Therefore, it is surmised that the large estimated error at gimbal lock is due to the coupled roll and yaw angles.

7.2.3 Results and discussion of hand roll velocity estimation on test videos

	Student (surface)	Student (depth)	Resident (surface)	Resident (depth)	Attending (surface)	Attending (depth)	Average
CRNN (angle & vel) (many-to-one)	0.54	0.39	0.8	0.71	0.57	0.38	0.57
CRNN (angle & vel) (many-to-many)	0.52	0.39	0.77	0.72	0.54	0.37	0.55

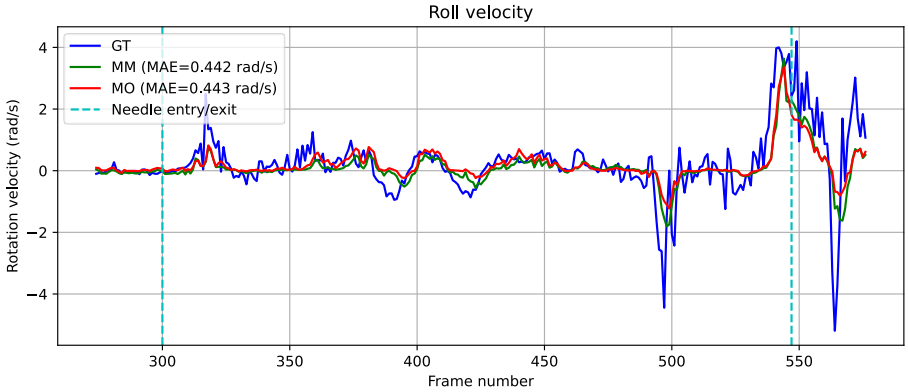
Table 7.6: MAE for roll velocity estimation (Unit: rad/s). The six videos are from Location D, which is for collecting the hand roll estimation dataset.

	Student (surface)	Student (depth)	Resident (surface)	Resident (depth)	Attending (surface)	Attending (depth)	Average
CRNN (angle & vel) (many-to-one)	0.63	0.49	0.77	0.7	0.63	0.72	0.66
CRNN (angle & vel) (many-to-many)	0.62	0.47	0.76	0.69	0.63	0.71	0.65

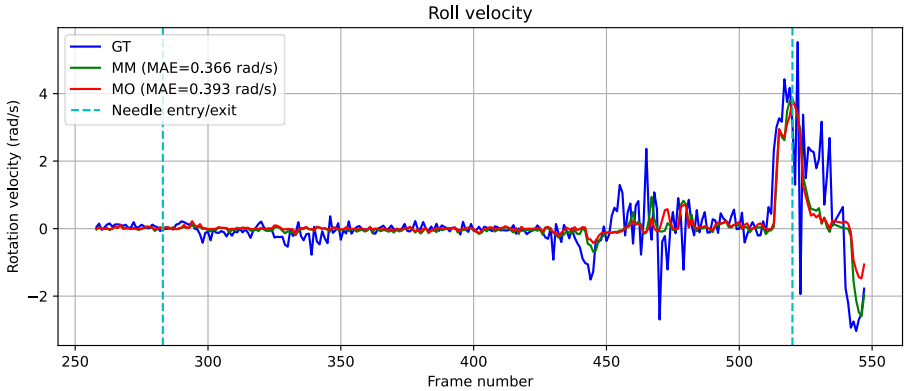
Table 7.7: MAE for roll velocity estimation (Unit: rad/s). The six videos are from Location C, which is NOT for collecting the hand roll estimation dataset.

Table 7.6 and 7.7 show the MAE of roll velocity estimated by CRNN(angle & vel). Specifically, Table 7.6 is for the six videos collected at Location D, which is for collecting the hand roll estimation dataset. In comparison, Table 7.7 is for the six videos collected at Location C, which is not for collecting the hand roll estimation dataset. The two tables arrange the columns in the same order as the tables for roll angle estimation errors (e.g., Table 7.4). Each number in the two tables is the average error of the 12 sutures in a single video. For simplicity, CRNN(angle & vel)(many-to-one) is denoted by MO, and CRNN(angle & vel)(many-to-many) is denoted by MM in this section.

As shown in the table for estimation errors on the test set (Table 7.3) and the tables for roll velocity estimation errors on the test videos (Table 7.6 and 7.7), MO and MM have similar performance for roll velocity estimation. By comparing Table 7.6 and 7.7, it is noticeable that the CRNNs have slightly better performance on the videos at Location D than at Location C. One



(a)



(b)

Figure 7.6: Hand roll velocity estimation for (a) a suture at the surface condition and (b) a suture at the depth condition. GT, MM, and MO refer to ground-truth data, CRNN(angle & vel)(many-to-many), and CRNN(angle & vel)(many-to-one), respectively. MAE for each algorithm is shown in the legend, while the cyan vertical lines denote the frame numbers corresponding to the needle entry/exit time.

reason is the videos in Table 7.6 are more similar to the hand roll estimation dataset than the videos in Table 7.7. Note that this phenomenon also exists when examining the hand roll angle estimation algorithms.

Fig. 7.6 visualizes the ground-truth (GT), MO, and MM roll velocity for two sutures in a frame-by-frame manner. Specifically, Fig. 7.6(a) corresponds to the suture in Fig. 7.3(a), which is the attending surgeon's first suture at the surface condition. In comparison, Fig. 7.6(b) corresponds to the suture in Fig. 7.3(b), which is the resident surgeon's first suture at the depth condition. The MAE of the algorithms is shown in the legend, while the cyan vertical lines refer to the needle entry/exit time for each suture.

From Fig. 7.6, it is noticeable that the ground-truth, MO, and MM roll velocity have similar

patterns, but MO and MM roll velocity are inaccurate. Specifically, the ground-truth data does not align with the MO or MM output when the magnitude of the ground-truth velocity is small (see the region from Frame 370 to Frame 430 in Fig. 7.6(a)). Worse, the sign of the MO and MM output is opposite to the sign of the ground-truth values at times (see the region from Frame 460 to Frame 480 in Fig. 7.6(b)).

7.2.4 Results and discussion of roll angle estimation errors on the entire dataset

Recall that only a portion of data at Location D is used to train and test the hand roll angle estimation algorithms. Based on the results shown in Table 7.5, it is noticeable that CNN(conv+FC) has the smallest roll angle estimation errors in the 6 test videos. Thus, CNN(conv+FC) is then used to estimate hand roll angles at the 5 locations. For simplicity, the CNN(conv+FC) estimated angle is abbreviated as CNN angle in this section. The MAE for CNN angle per location are shown in Table 7.8.

	# sutures	# images	CNN(conv+FC) (surface)	CNN(conv+FC) (depth)
Location A	331	75k	10.02°	10.66°
Location B	1009	235k	7.18°	12.97°
Location C	937	200k	6.48°	8.03°
Location D	389	57k	7.07°	11.35°
Location E	505	189k	11.39°	22.43°

Table 7.8: MAE for CNN angle at the 5 locations

As shown in Table 7.8, CNN(conv+FC) has similar roll angle estimation errors at Locations A-D. Since only Location D was used to generate the hand roll estimation dataset, these results indicate CNN(conv+FC) performance is robust to environmental variations. Among the 5 locations, Location E has the largest errors for CNN angle. One reason is the camera orientation is noticeably different at Location E compared with other locations (see Fig. 6.3), which leads to bias in the roll estimation. Another reason is the participants at Location E were novices who had little suturing experience, so they had suturing behavior and hand movements that rarely occurred in the hand roll estimation dataset. Also, Table 7.8 shows that the errors of CNN angle are larger at depth than at surface conditions. One reason is the participants' hands are more likely to have a large pitch

angle when suturing at depth, causing the IMU to experience gimbal lock and to have undesirable roll measurements (see Section 7.2.2.5).

7.3 Results and discussion of hand-roll metrics

For the metrics calculated from ground-truth roll angles and those calculated from CNN (conv+FC) estimated roll angles, one-way ANOVA shows that the expert, intermediate, and novice groups have significantly different means at both surface and depth conditions. Also, Levene's test shows that the metrics have different variances among the three groups at the two conditions. For simplicity, the ground-truth roll angle is abbreviated as GT angle, and the CNN(conv+FC) estimated roll angle is abbreviated as CNN angle in this section.

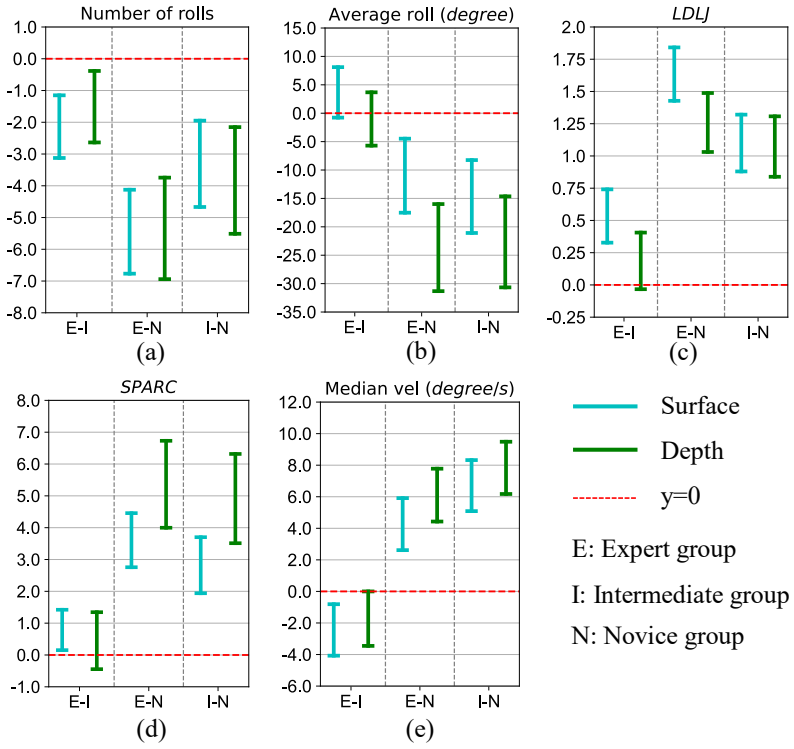


Figure 7.7: Confidence intervals for the five hand-roll metrics calculated using GT angles. Each sub-figure shows pairwise comparisons of experts versus intermediates (E-I), experts versus novices (E-N), and intermediates versus novices (I-N). For each pairwise comparison, surface and depth conditions are shown separately.

Fig. 7.7 is the confidence interval of the metrics calculated from GT angles, while Fig. 7.8 is the confidence interval of the metrics calculated from CNN angles. In Fig. 7.7 and 7.8, each sub-

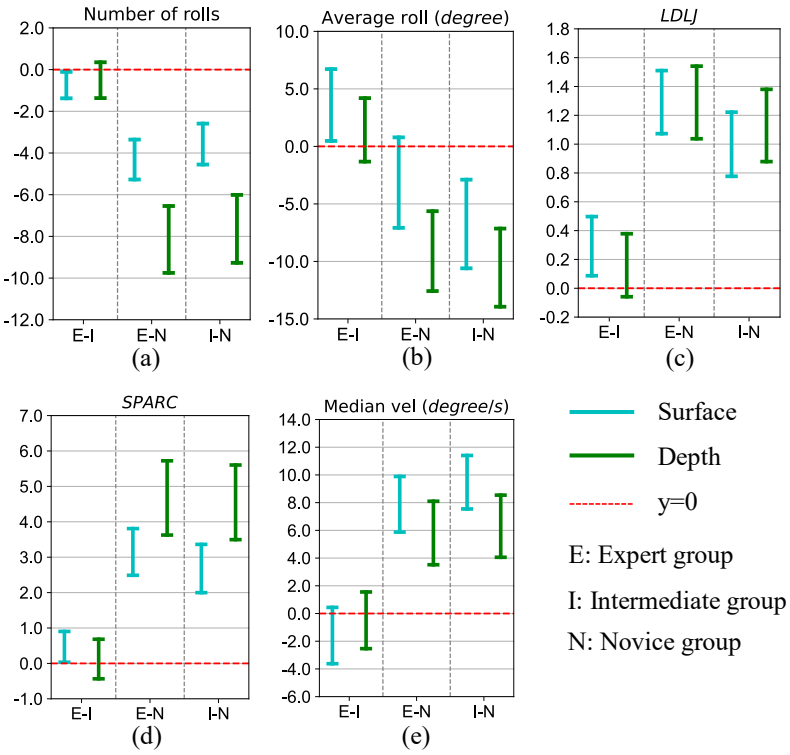


Figure 7.8: Confidence intervals for the five hand-roll metrics calculated using CNN angles. Each sub-figure shows pairwise comparisons of experts versus intermediates (E-I), experts versus novices (E-N), and intermediates versus novices (I-N). For each pairwise comparison, surface and depth conditions are shown separately.

figure shows the comparisons of experts versus intermediates (E-I), experts versus novices (E-N), and intermediates versus novices (I-N) for a specific metric. The comparisons are made separately for surface and depth conditions. The confidence interval denotes the range of likely differences between the two group means. Therefore, if a confidence interval does not cross zero, indicated by the red dashed line, then the null hypothesis is rejected. In other words, if zero is not in a confidence interval, there is a statistically significant difference between the mean of the two groups. Detailed statistical analyses for the metrics of GT angles are summarized in Table 2 in Appendix C, while the statistical analyses for the metrics of CNN angles are summarized in Table 3 in Appendix C.

Fig. 7.7 and 7.8 show that the novice group has significantly different means from the other two groups when the metrics are calculated from GT angles or from CNN angles. Moreover, Fig. 7.7 and 7.8 show that intermediate and expert groups have significant mean differences at the surface condition for `Number of rolls`, `LDLJ`, and `SPARC` calculated from GT angles and from CNN angles.

It indicates CNN angles have similar performance as GT angles for surgical suturing skill assessment.

Compared with the results in [76], Fig. 7.7 shows that `Average roll` and `Median vel` calculated from GT angles have decreased effectiveness. One reason is that `Average roll` and `Median vel` in Fig. 7.7 are calculated based on IMU roll angles, but the metrics in [76] are calculated from IMU roll velocity that is free from gimbal lock. Also, Section 7.2.2.5 shows that gimbal lock hinders CNN(conv+FC) from learning proper hand roll angles, causing `Average roll` and `Median vel` calculated from CNN angles become less effective than the results in [76].

Fig. 7.9 shows the boxplots for the metrics calculated from GT angles, while Fig. 7.10 shows the boxplots for the metrics calculated from CNN angles. For the boxplots, values for the three groups are presented at surface and depth conditions separately. The boxplots also include the mean metric values for each group. Extreme outliers are not shown in the boxplots to avoid cluttering the figures.

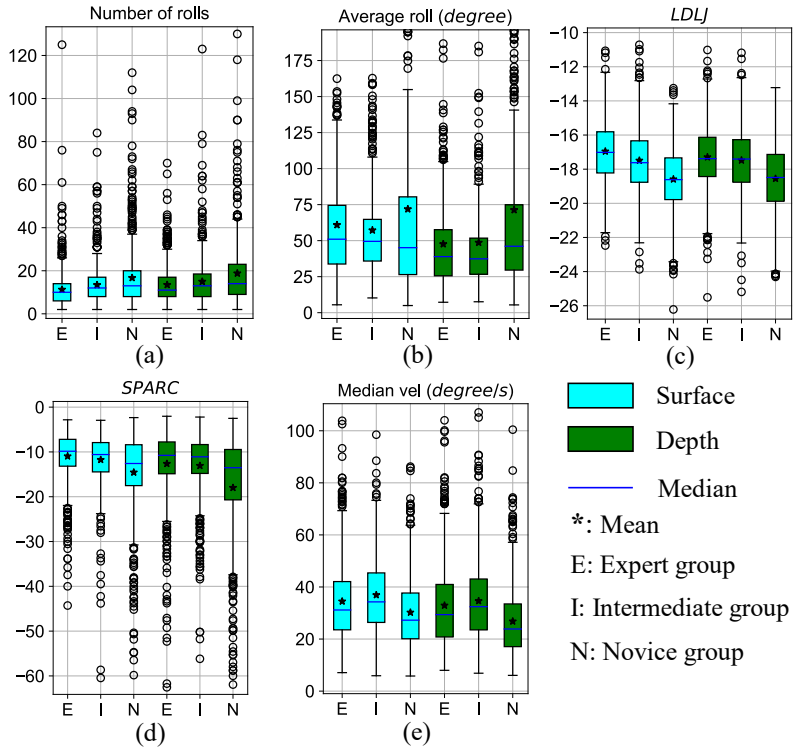


Figure 7.9: Boxplots of the five hand-roll metrics calculated using GT angles. Each sub-figure presents the outputs of a single metric, separated into three groups, at surface and depth conditions. The extreme outliers are not shown in the boxplots.

Fig. 7.9 and 7.10 show that `Number of rolls` calculated from GT angles and `Number of`

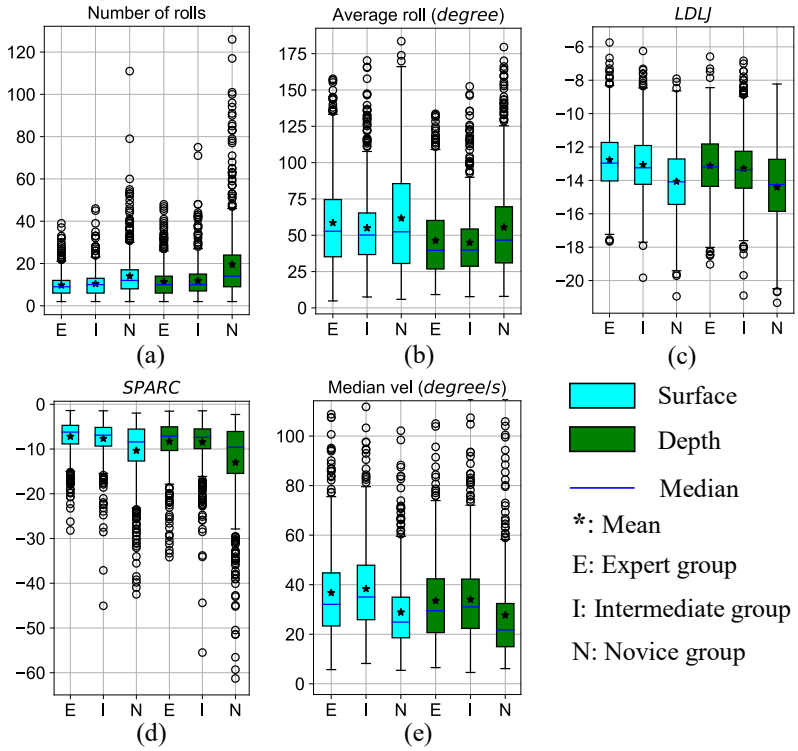


Figure 7.10: Boxplots of the five hand-roll metrics calculated using CNN angles. Each sub-figure presents the outputs of a single metric, separated into three groups, at surface and depth conditions. The extreme outliers are not shown in the boxplots.

`rolls` calculated from CNN angles have the smallest mean in the expert group, followed by the intermediate group and the novice group at the surface condition, which supports the hypothesis that `Number of rolls` decreases as participants' suturing skills increase. Also, Fig. 7.9 and 7.10 show that `LDLJ` and `SPARC` of GT angles and CNN angles have the largest mean in the expert group, followed by the intermediate group and the novice group at the surface condition. This observation supports the hypothesis that the values of `LDLJ` and `SPARC` are proportional to participants' suturing skills.

Chapter 8

Conclusion and future work

8.0.1 Conclusion

This thesis presents the software and hardware improvement of a surgical suturing simulator, which is developed based on the FVS clock-face model. The improved simulator allows the collection of reliable data at multiple locations. The dataset, including 97 participants, is constructed by sensor measurement from 2 cameras, 2 IMUs, a force sensor, and an electromagnetic motion tracker. For the two cameras, one records videos with needle motion, the other records videos with hand motion.

To examine if the videos with needle motion can be used for suturing skill assessment, eight image-based metrics are used to analyze the videos. Statistical analysis shows that all the image-based metrics have statistical mean differences between novice and surgeon groups. Moreover, using the depth constraint, 7 out of the 8 metrics have significantly different means between resident surgeon and attending surgeon groups. This suggests that the eight image-based metrics are capable of suturing skill evaluation. For the 8 metrics, the ones measuring needle lateral motion during a suture (`SweptArea`, `SwayLen`, `Range(NAngle)`, `AVG(NAngle)`, and `Range(TipOrth))` outperform `TipPathLen`, which measures the needle trajectory during a suture. Also, the improved suture phase segmentation allows segmenting data at a specific suture phase, which enables to create new metrics for suturing skill assessment. Specifically, `RemovalTime`, calculated based on the suture phase segmentation, has significant mean differences among the three groups, whereas the duration of sutures fails to show significantly different means among the three groups. Further, this dissertation introduces ROC curves to examine the classification performance of the metrics. Results show

that `SweptArea` has the highest classification accuracy between novice and expert groups, while `TipPathLen` has the highest classification accuracy between novice and intermediate groups.

To examine if the videos with hand motion can be used for suturing skill assessment, a hand detection algorithm is used to localize participants' hands in video frames. The cropped hand images are then processed by a hand roll estimation algorithm. The result shows that the hand roll estimation algorithm has consistent performance in different environments. Also, the hand-roll metrics calculated based on the algorithm output have similar statistical results as the hand-roll metrics calculated based on the IMU measurement. Specifically, 4 out of the 5 metrics have significant mean differences between novice and surgeon groups. Moreover, 3 out of the 5 metrics have significantly different means between resident surgeon and attending surgeon groups at the surface condition. This suggests that metrics calculated from neural network estimates of hand roll can be used for suturing skill assessment. The proposed hand detection and hand roll estimation algorithms enable rotational motion analysis of the hand without the need to attach sensors to clinicians and therefore eliminates any physical interference with their performance. Subsequently, this method of hand motion analysis paves the way for intra-operative surgical skill assessment, as visual tracking is non-contact and circumvents typical sterility issues.

8.0.2 Future work

SutureCoach still has some limitations. First, participants' actual suturing skills are approximated by their clinical standing. Furthermore, the membrane material caused the image-processing program to have several problematic detections. In the future, participants' ground-truth suturing skills will be determined by several objective skill assessments. Also, other materials will be investigated for making the membrane. Moreover, we will evaluate participants' skill levels by fitting multiple metrics into a linear model, which helps to discover the optimal combination of metrics for surgical suturing skill assessment. We hope that these improvements will allow SutureCoach to be used in medical schools to assist with surgical suturing education.

For the hand roll estimation algorithm, one limitation is that the estimated errors increased when gimbal lock occurred or the hand was out of camera view. Another limitation pertained to how we collected the data for this study, as inconsistent camera orientation adversely affected our results. To overcome these limitations, the camera's field of view will be extended in the future, and the camera location will be secured on SutureCoach. Also, the IMU will be set to use rotation matrices

rather than Euler angles while recording and outputting 3D orientation, as rotation matrices are free from gimbal lock.

The ultimate goal of the deep-learning algorithms is to allow surgical suturing skill assessment in operating rooms. To achieve the goal, the hand detection and the hand roll estimation algorithm will be integrated into SutureCoach to examine if the two algorithms can be used for real-time hand roll estimation. Also, the two algorithms will be retrained by a new dataset that has the original camera orientation and without the appearance of the orange IMU. Since the original camera orientation can be used as a reference, this dataset will enable the hand roll estimation algorithm to estimate the 3D orientation of the hand. The dataset also eliminates the visual hint due to the color and shape of the IMU.

Appendices

Appendix A Questionnaire for data collection

ID: _____
Date: _____

Suturing Study Questionnaire

Age: _____

Sex (circle one): **Male** **Female** **Prefer not to answer**

Height: _____

Dominant hand (circle one): **Left** **Right** **Either**

Which of the following best describes you (circle one):

Attending Surgeon: **Year**_____ **Specialty (Vascular, General, etc…)** _____

Fellow: **Year**_____ **Specialty**_____

Resident: **Year**_____ **Planned Specialty**_____

Medical student: **Year**_____

Intern or Undergraduate student: **Year**_____ **Major**_____

Please list the **total hours** spent on **suturing training** in your training (labs, workshops, models used [e.g., pigs feet, synthetic], etc.)

Suturing on synthetic models	***None***	***< 5hrs.***	***5-15 hrs.***	***15-30 hrs.***	***30-60 hrs.***	***>60 hrs.***
Suturing on *ex-vivo* animal tissue (pigs feet, etc.)	***None***	***< 5hrs.***	***5-15 hrs.***	***15-30 hrs.***	***30-60 hrs.***	***>60 hrs.***
Suturing in-vivo during a procedure	***None***	***< 5hrs.***	***5-15 hrs.***	***15-30 hrs.***	***30-60 hrs.***	***>60 hrs.***
Suturing on surgical simulators	***None***	***< 5hrs.***	***5-15 hrs.***	***15-30 hrs.***	***30-60 hrs.***	***>60 hrs.***

Do you currently have any problems with your hands, arms, or vision that may impair your suturing performance? **Yes** **No**

If yes, please describe: _____

_____

- Approximately how many years have you been practicing surgical procedures involving suturing?

  _____ **years**

- Approximately how many surgical procedures involving suturing have you performed **in the last two years**?

  1-25 **25-100** **101-500** **501-1,000** **1,001-1,500** **>1,500** **N/A**

ID: _____
Date: _____

- Approximately how many **Non-Robotic Minimally Invasive Surgeries** involving suturing have you performed?

  1-25 25-100 101-500 501-1,000 1,001-1,500 >1,500 N/A

- Approximately how many **Robotic Minimally Invasive Surgeries** involving suturing have you performed?

  1-25 25-100 101-500 501-1,000 1,001-1,500 >1,500 N/A

- On a scale of 1-10, how would you rate your surgical suturing skill ***(1=poor, 10=world-class)***

  _____

Post-test feedback

On a scale of 1-10, how would you rate the ease/comfort of using the suturing device? (1=worst; 10=best):

_____

What are the strengths of the suturing device?

_____

What are the weaknesses of the suturing device?

_____

In your opinion, what features would the ideal suturing training device have?

_____

What would you suggest to improve this system so that it may successfully aid in the suturing learning process?

_____

Are there any additional comments you would like to add?

_____

Appendix B Statistical analysis for metrics for needle motion

Table 1 shows the mean difference, standard error, and p-value for the 8 metrics at surface and depth conditions. The pairwise comparison with $p < 0.05$ is highlighted in bold.

Table 1: Pairwise comparisons among the three groups (mean difference (standard error, p-value))

	conditions	E vs. I	E vs. N	I vs. N
TipPathLen	Surface	0.72 (0.9, 0.4)	**-14.4 (2.1, <0.001)**	**-15.2 (2.1, <0.001)**
	Depth	0.93 (1.8, 0.6)	**-32 (3.5, <0.001)**	**-32.9 (3.4, <0.001)**
SweptArea	Surface	-1.23 (1.1, 0.24)	**-21.2 (1.6, <0.001)**	**-20 (1.64, <0.001)**
	Depth	**-6.4 (1.5, <0.001)**	**-36.1 (2.1, <0.001)**	**-29.7 (2.1, <0.001)**
SwayLen	Surface	-0.04 (0.04, 0.26)	**-0.58 (0.05, <0.001)**	**-0.54 (0.05, <0.001)**
	Depth	**-0.26 (0.05, <0.001)**	**-0.86 (0.06, <0.001)**	**-0.6 (0.06., <0.001)**
StitchLen	Surface	**-0.38 (0.12, 0.001)**	**-1.44 (0.13, <0.001)**	**-1.06 (0.13, <0.001)**
	Depth	**-0.59 (0.12, <0.001)**	**-1.64 (0.13, <0.001)**	**-1.05 (0.13, <0.001)**
Range(NAngle)	Surface	-0.44 (1.28, 0.73)	**-16 (1.56, <0.001)**	**-15.6 (1.56, <0.001)**
	Depth	**-5.49 (1.71, 0.001)**	**-27 (2, <0.001)**	**-21.5 (2, <0.001)**
AVG(NAngle)	Surface	-0.23 (0.31, 0.45)	**-3.86 (0.36, <0.001)**	**-3.63 (0.37, <0.001)**
	Depth	**-3.11 (0.46, <0.001)**	**-6.13 (0.5, <0.001)**	**-3.02 (0.53, <0.001)**
Range(TipOrth)	Surface	0.05 (0.09, 0.54)	**-1.41 (0.12, <0.001)**	**-1.46 (0.12, <0.001)**
	Depth	**-0.6 (0.14, <0.001)**	**-2.77 (0.17, <0.001)**	**-2.17 (0.17, <0.001)**
RemovalTime	Surface	**-0.29 (0.11, 0.009)**	**-1.63 (0.14, <0.001)**	**-1.33 (0.14, <0.001)**
	Depth	**-0.38 (0.11, <0.001)**	**-1.86 (0.16, <0.001)**	**-1.48 (0.17, <0.001)**

Appendix C Statistical analysis for metrics for hand motion

Table 2 shows the mean difference, standard error, and p-value for hand motion metrics calculated from IMU roll angles. The results of surface and depth conditions are shown separately. The pairwise comparison with $p < 0.05$ is highlighted in bold.

Table 2: Pairwise comparisons among the three groups (mean difference (standard error, p-value)). The results are calculated based on IMU roll angles. Results with $p < 0.05$ are highlighted in bold.

	conditions	E vs. I	E vs. N	I vs. N
`Number of rolls`	Surface	**-2.14 (0.5, <0.001)**	**-5.45 (0.67, <0.001)**	**-3.31 (0.69, <0.001)**
	Depth	**-1.51 (0.57, 0.009)**	**-5.34 (0.81, <0.001)**	**-3.83 (0.86, <0.001)**
`Average rolls`	Surface	3.67 (2.26, 0.11)	**-11 (3.32, <0.001)**	**-14.7 (3.27, <0.001)**
	Depth	-1.01 (2.39, 0.67)	**-23.66 (3.9, <0.001)**	**-22.65 (4.08, <0.001)**
`LDLJ`	Surface	**0.53 (0.11, <0.001)**	**1.63 (0.11, <0.001)**	**1.1 (0.11, <0.001)**
	Depth	0.19 (0.11, 0.1)	**1.26 (0.12, <0.001)**	**1.07 (0.12, <0.001)**
`SPARC`	Surface	**0.79 (0.32, 0.02)**	**3.61 (0.43, <0.001)**	**2.82 (0.45, <0.001)**
	Depth	0.45 (0.46, 0.32)	**5.36 (0.7, <0.001)**	**4.91 (0.71, <0.001)**
`Median vel`	Surface	**-2.44 (0.83, 0.003)**	**4.26 (0.84, <0.001)**	**6.71 (0.82, <0.001)**
	Depth	-1.73 (0.88, 0.05)	**6.1 (0.85, <0.001)**	**7.83 (0.84, <0.001)**

Table 3 shows the mean difference, standard error, and p-value for hand motion metrics calculated from CNN(conv+FC) estimated angles. The results of surface and depth conditions are shown separately. The pairwise comparison with $p < 0.05$ is highlighted in bold.

Table 3: Pairwise comparisons among the three groups (mean difference (standard error, p-value)). The results are calculated based on CNN(conv+FC) estimated angles. Results with $p < 0.05$ are highlighted in bold.

	conditions	E vs. I	E vs. N	I vs. N
`Number of rolls`	Surface	**-0.74 (0.32, 0.02)**	**-4.31 (0.49, <0.001)**	**-3.57 (0.5, <0.001)**
	Depth	-0.51 (0.44, 0.25)	**-8.15 (0.82, <0.001)**	**-7.64 (0.83, <0.001)**
`Average roll`	Surface	**3.6 (1.59, 0.024)**	-3.15 (2, 0.12)	**-6.74 (1.96, <0.001)**
	Depth	1.44 (1.41, 0.31)	**-9.1 (1.77, <0.001)**	**-10.5 (1.73, <0.001)**
`LDLJ`	Surface	**0.29 (0.1, 0.005)**	**1.29 (0.11, <0.001)**	**1 (0.11, <0.001)**
	Depth	0.16 (0.11, 0.15)	**1.29 (0.13, <0.001)**	**1.13 (0.13, <0.001)**
`SPARC`	Surface	**0.47 (0.22, 0.03)**	**3.15 (0.34, <0.001)**	**2.68 (0.35, <0.001)**
	Depth	0.12 (0.28, 0.66)	**4.67 (0.53, <0.001)**	**4.55 (0.54, <0.001)**
`Median vel`	Surface	-1.59 (1.04, 0.124)	**7.88 (1.02, <0.001)**	**9.47 (0.98, <0.001)**
	Depth	-0.49 (1.04, 0.64)	**5.81 (1.17, <0.001)**	**6.3 (1.14, <0.001)**

Bibliography

[1] Alexandre Alahi, Kratarth Goel, Vignesh Ramanathan, Alexandre Robicquet, Li Fei-Fei, and Silvio Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 961–971, 2016.

[2] Florent Altché and Arnaud de La Fortelle. An lstm network for highway trajectory prediction. In *2017 IEEE 20th international conference on intelligent transportation systems (ITSC)*, pages 353–359. IEEE, 2017.

[3] Dimitri J Anastakis, Glenn Regehr, Richard K Reznick, Michael Cusimano, John Murnaghan, Mitchell Brown, and Carol Hutchison. Assessment of technical skills transfer from the bench training model to the human model. *The American journal of surgery*, 177(2):167–170, 1999.

[4] Aqeel Anwar. What is average precision in object detection & localization algorithms and how to calculate it? `https://towardsdatascience.com/what-is-average-precision-in-object-detection-localization-algorithms-and-how-to-calculate-it-3f330efe697b`. Accessed: 2024-05-09.

[5] Sivakumar Balasubramanian, Alejandro Melendez-Calderon, and Etienne Burdet. A robust and sensitive metric for quantifying movement smoothness. *IEEE transactions on biomedical engineering*, 59(8):2126–2136, 2011.

[6] Simon D Bann, Mansoor S Khan, and Ara W Darzi. Measurement of surgical dexterity using motion analysis of simple bench tasks. *World journal of surgery*, 27:390–394, 2003.

[7] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[8] John D Birkmeyer, Jonathan F Finks, Amanda O'reilly, Mary Oerline, Arthur M Carlin, Andre R Nunn, Justin Dimick, Mousumi Banerjee, and Nancy JO Birkmeyer. Surgical skill and complication rates after bariatric surgery. *New England Journal of Medicine*, 369(15):1434–1442, 2013.

[9] Junyang Chen, Hui Liu, Yating Zhang, Daike Zhang, Hongkun Ouyang, and Xiaoyan Chen. A multiscale lightweight and efficient model based on yolov7: Applied to citrus orchard. *Plants*, 11(23):3260, 2022.

[10] Vivek Datta, Avril Chang, Sean Mackay, and Ara Darzi. The relationship between motion analysis and surgical technical assessments. *The American journal of surgery*, 184(1):70–73, 2002.

[11] Vivek Datta, Sean Mackay, Mirren Mandalia, and Ara Darzi. The use of electromagnetic motion tracking analysis to objectively measure open surgical skill in the laboratory-based model. *Journal of the American College of Surgeons*, 193(5):479–485, 2001.

[12] Quentin De Smedt. *Dynamic hand gesture recognition-From traditional handcrafted to recent deep learning approaches*. PhD thesis, Université de Lille 1, Sciences et Technologies; CRIStAL UMR 9189, 2017.

[13] Pon Deepika, Kalahasti VV Deepesh, Phani Sriram Vadali, Madhav Rao, Vikas Vazhayil, and Alok Mohan Uppar. Computer assisted objective assessment of micro-neurosurgical skills from intraoperative videos. *IEEE Open Journal of Engineering in Medicine and Biology*, 4:11–20, 2023.

[14] Xiaoming Deng, Yinda Zhang, Shuo Yang, Ping Tan, Liang Chang, Ye Yuan, and Hongan Wang. Joint hand detection and rotation estimation using cnn. *IEEE transactions on image processing*, 27(4):1888–1900, 2017.

[15] Erkan Deniz, Abdulkadir Şengür, Zehra Kadiroğlu, Yanhui Guo, Varun Bajaj, and Ümit Budak. Transfer learning based histopathologic image classification for breast cancer detection. *Health information science and systems*, 6:1–7, 2018.

[16] Anna M Derossis, Gerald M Fried, Harvey H Sigman, Jeffrey S Barkun, and Jonathan L Meakins. Development of a model for training and evaluation of laparoscopic skills. *The American journal of surgery*, 175(6):482–487, 1998.

[17] Xiaohan Ding, Xiangyu Zhang, Ningning Ma, Jungong Han, Guiguang Ding, and Jian Sun. Repvgg: Making vgg-style convnets great again. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13733–13742, 2021.

[18] Bardia Doosti. Hand pose estimation: A survey. *arXiv preprint arXiv:1903.01013*, 2019.

[19] Aristotelis Dosis, Rajesh Aggarwal, Fernando Bello, Krishna Moorthy, Yaron Munz, Duncan Gillies, and Ara Darzi. Synchronized video and motion analysis for the assessment of procedures in the operating theater. *Archives of Surgery*, 140(3):293–299, 2005.

[20] Jeffrey D Doyle, Eric M Webber, and Ravi S Sidhu. A universal global rating scale for the evaluation of technical skills in the operating room. *The American journal of surgery*, 193(5):551–555, 2007.

[21] Adam Dubrowski, Ravi Sidhu, Jason Park, and Heather Carnahan. Quantification of motion characteristics and forces applied to tissues during suturing. *The American journal of surgery*, 190(1):131–136, 2005.

[22] Tom Fawcett. An introduction to roc analysis. *Pattern recognition letters*, 27(8):861–874, 2006.

[23] Cesar Ferri, José Hernández-Orallo, and Peter A Flach. A coherent interpretation of auc as a measure of aggregated classification performance. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 657–664, 2011.

[24] Gerald M Fried, Liane S Feldman, Melina C Vassiliou, Shannon A Fraser, Donna Stanbridge, Gabriela Ghitulescu, and Christopher G Andrew. Proving the value of simulation in laparoscopic surgery. *Annals of surgery*, 240(3):518, 2004.

[25] Adam C Frischknecht, Steven J Kasten, Stanley J Hamstra, Noel C Perkins, R Brent Gillespie, Thomas J Armstrong, and Rebecca M Minter. The objective assessment of experts' and novices' suturing skills using an image analysis program. *Academic Medicine*, 88(2):260–264, 2013.

[26] Isabel Funke, Sören Torge Mees, Jürgen Weitz, and Stefanie Speidel. Video-based surgical skill assessment using 3d convolutional neural networks. *International journal of computer assisted radiology and surgery*, 14:1217–1225, 2019.

[27] Jun Gao, Luyun Gan, Fabiola Buschendorf, Liao Zhang, Hua Liu, Peixue Li, Xiaodai Dong, and Tao Lu. Lstm for scada intrusion detection. In *2019 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM)*, pages 1–5. IEEE, 2019.

[28] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1991–2000, 2017.

[29] Adam Goldbraikh, Anne-Lise D'Angelo, Carla M Pugh, and Shlomi Laufer. Video-based fully automatic assessment of open surgery suturing skills. *International Journal of Computer Assisted Radiology and Surgery*, 17(3):437–448, 2022.

[30] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[31] James A Hanley and Barbara J McNeil. The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, 143(1):29–36, 1982.

[32] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[33] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[34] Neville Hogan and Dagmar Sternad. Sensitivity of smoothness measures to movement duration, amplitude, and arrests. *Journal of motor behavior*, 41(6):529–534, 2009.

[35] Tim Horeman, Sharon P Rodrigues, Frank-Willem Jansen, Jenny Dankelman, and John J van den Dobbelsteen. Force measurement platform for training and assessment of laparoscopic skills. *Surgical endoscopy*, 24:3102–3108, 2010.

[36] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. pmlr, 2015.

[37] Gazi Islam, Kanav Kahol, John Ferrara, and Richard Gray. Development of computer vision algorithm for surgical skill assessment. In *Ambient Media and Systems: Second International ICST Conference, AMBI-SYS 2011, Porto, Portugal, March 24-25, 2011, Revised Selected Papers 2*, pages 44–51. Springer, 2011.

[38] Gazi Islam, Kanav Kahol, and Baoxin Li. Development of a computer vision application for surgical skill training and assessment. *Bio-Informatic Systems, Processing and Applications*, page 191, 2013.

[39] Gazi Islam, Kanav Kahol, Baoxin Li, Marshall Smith, and Vimla L Patel. Affordable, web-based surgical skill training and evaluation tool. *Journal of biomedical informatics*, 59:102–114, 2016.

[40] Russell C Jackson and M Cenk Çavuşoğlu. Modeling of needle-tissue interaction forces during surgical suturing. In *2012 IEEE International Conference on Robotics and Automation*, pages 4675–4680. IEEE, 2012.

[41] Russell C Jackson and M Cenk Çavuşoğlu. Needle path planning for autonomous robotic surgical suturing. In *2013 IEEE International Conference on Robotics and Automation*, pages 1669–1675. IEEE, 2013.

[42] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. *Advances in neural information processing systems*, 28, 2015.

[43] Anand Jagannathan. *Suture training device with computer vision based information acquisition.* PhD thesis, Clemson University, 2016.

[44] Tanmay Kavathekar. *Development of a suturing simulation device for synchronous acquisition of data.* PhD thesis, Clemson University, 2015.

[45] Tanmay Kavathekar, Irfan Kil, Richard E Groff, Timothy C Burg, John F Eidt, and Ravikiran B Singapogu. Towards quantifying surgical suturing skill with force, motion and image sensor data. In *2017 IEEE EMBS International Conference on Biomedical & Health Informatics (BHI)*, pages 169–172. IEEE, 2017.

[46] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE international conference on computer vision*, pages 1521–1529, 2017.

[47] Shuja Khalid, Mitchell Goldenberg, Teodor Grantcharov, Babak Taati, and Frank Rudzicz. Evaluation of deep learning models for identifying surgical actions and measuring performance. *JAMA network open*, 3(3):e201664–e201664, 2020.

[48] Mansoor S Khan, SD Bann, A Darzi, and PEM Butler. Use of suturing as a measure of technical competence. *Annals of plastic surgery*, 50(3):304–309, 2003.

[49] Irfan Kil. *Development and Preliminary Validation of Image-enabled Process Metrics for Assessment of Open Surgery Suturing Skill.* PhD thesis, Clemson University, 2019.

[50] Irfan Kil, John F Eidt, Richard E Groff, and Ravikiran B Singapogu. Assessment of open surgery suturing skill: Simulator platform, force-based, and motion-based metrics. *Frontiers in Medicine*, 9:897219, 2022.

[51] Irfan Kil, Richard E Groff, and Ravikiran B Singapogu. Surgical suturing with depth constraints: Image-based metrics to assess skill. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 4146–4149. IEEE, 2018.

[52] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[53] Deying Kong, Haoyu Ma, Yifei Chen, and Xiaohui Xie. Rotation-invariant mixed graphical model network for 2d hand pose estimation. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 1546–1555, 2020.

[54] Chen-Yu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu. Deeply-supervised nets. In *Artificial intelligence and statistics*, pages 562–570. Pmlr, 2015.

[55] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[56] TzuTa Lin. Labelimg - a graphical annotation tool. `https://github.com/HumanSignal/labelImg`. Accessed: 2024-05-09.

[57] Daochang Liu, Qiyue Li, Tingting Jiang, Yizhou Wang, Rulin Miao, Fei Shan, and Ziyu Li. Towards unified surgical skill assessment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9522–9531, 2021.

[58] Martin A Makary and Michael Daniel. Medical error—the third leading cause of death in the us. *Bmj*, 353, 2016.

[59] Erica L Mitchell, Malachi G Sheahan, and Mélanie Schwiesow. Simulation in vascular surgery. *Comprehensive Healthcare Simulation: Surgery and Surgical Subspecialties*, pages 327–347, 2019.

[60] Naren Nagarajan. *A suture training system with synchronized force, motion and video data collection*. PhD thesis, Clemson University, 2016.

[61] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[62] Hiroaki Niitsu, Naoki Hirabayashi, Masanori Yoshimitsu, Takeshi Mimura, Junya Taomoto, Yoich Sugiyama, Shigeru Murakami, Shuji Saeki, Hidenori Mukaida, and Wataru Takiyama. Using the objective structured assessment of technical skills (osats) global rating scale to evaluate the skills of surgical trainees in the operating room. *Surgery today*, 43:271–275, 2013.

[63] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks. *arXiv preprint arXiv:1511.08458*, 2015.

[64] Fernando Pérez-Escamirosa, Antonio Alarcón-Paredes, Gustavo Adolfo Alonso-Silverio, Ignacio Oropesa, Oscar Camacho-Nieto, Daniel Lorias-Espinoza, and Arturo Minor-Martínez. Objective classification of psychomotor laparoscopic skills of surgeons based on three different approaches. *International Journal of Computer Assisted Radiology and Surgery*, 15(1):27–40, 2020.

[65] Ravi Rajaram, Jeanette W Chung, Andrew T Jones, Mark E Cohen, Allison R Dahlke, Clifford Y Ko, John L Tarpley, Frank R Lewis, David B Hoyt, and Karl Y Bilimoria. Association of the 2011 acgme resident duty hour reform with general surgery patient outcomes and with resident examination performance. *Jama*, 312(22):2374–2384, 2014.

[66] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[67] Carol E Reiley, Henry C Lin, David D Yuh, and Gregory D Hager. Review of methods for objective surgical skill evaluation. *Surgical endoscopy*, 25:356–366, 2011.

[68] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[69] C Richards, J Rosen, B Hannaford, C Pellegrini, and M Sinanan. Skills evaluation in minimally invasive surgery using force/torque signatures. *Surgical endoscopy*, 14:791–798, 2000.

[70] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007.

[71] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015.

[72] Stuart J Russell. *Artificial intelligence a modern approach*. Pearson Education, Inc., 2010.

[73] Ingrid S Schmiederer, LaDonna E Kearse, James R Korndorffer Jr, Edmund Lee, Michael D Sgroi, and Jason T Lee. Validity evidence for vascular skills assessment: The feasibility of fundamentals of vascular surgery in general surgery residency. *Journal of Surgical Education*, 78(6):e201–e209, 2021.

[74] Nadine B Semer and Marthe Adler-Lavan. *Practical plastic surgery for nonsurgeons*. Hanley & Belfus Philadelphia, 2001.

[75] Yarden Sharon, Anthony M Jarc, Thomas S Lendvay, and Ilana Nisky. Rate of orientation change as a new metric for robot-assisted and open surgical skill evaluation. *IEEE Transactions on Medical Robotics and Bionics*, 3(2):414–425, 2021.

[76] Amir Mehdi Shayan, Simar Singh, Jianxin Gao, Richard E Groff, Joe Bible, John F Eidt, Malachi Sheahan, Sagar S Gandhi, Joseph V Blas, and Ravikiran Singapogu. Measuring hand movement for suturing skill assessment: A simulation-based study. *Surgery*, 2023.

[77] Malachi G Sheahan, Cassidy Duran, and Jean Bismuth. National simulation-based training of fellows: the vascular surgery example. *Surgical Clinics*, 95(4):781–790, 2015.

[78] David A Sherris and Eugene B Kern. *Essential surgical skills*. WB Saunders Company, 2004.

[79] Ravi S Sidhu, Jason Park, Ryan Brydges, Helen M MacRae, and Adam Dubrowski. Laboratory-based vascular anastomosis training: a randomized controlled trial evaluating the effects of bench model fidelity and level of training on skill acquisition. *Journal of vascular surgery*, 45(2):343–349, 2007.

[80] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[81] Ravikiran B Singapogu, Tanmay Kavathekar, John F Eidt, Richard E Groff, and Timothy C Burg. A novel platform for assessment of surgical suturing skill: Preliminary results. In *MMVR*, pages 375–378, 2016.

[82] Simar Singh, Joe Bible, Zhanhe Liu, Ziyang Zhang, and Ravikiran Singapogu. Motion smoothness metrics for cannulation skill assessment: What factors matter? *Frontiers in Robotics and AI*, 8:625003, 2021.

[83] Simar Singh, Amir Mehdi Shayan, Jianxin Gao, Joe Bible, Richard E Groff, and Ravikiran Singapogu. Objective and automated quantification of instrument handling for open surgical suturing skill assessment: A simulation-based study. unpublished, 2023.

[84] Abed Soleymani, Ali Akbar Sadat Asl, Mojtaba Yeganejou, Scott Dick, Mahdi Tavakoli, and Xingyu Li. Surgical skill evaluation from robot-assisted surgery recordings. In *2021 International Symposium on Medical Robotics (ISMR)*, pages 1–6. IEEE, 2021.

[85] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[86] Zar Nawab Khan Swati, Qinghua Zhao, Muhammad Kabir, Farman Ali, Zakir Ali, Saeed Ahmed, and Jianfeng Lu. Brain tumor classification for mr images using transfer learning and fine-tuning. *Computerized Medical Imaging and Graphics*, 75:34–46, 2019.

[87] Lucille Treil, Nicole Neumann, Nicolas Chanes, Anne Lejay, Tristan Bourcier, Jean Bismuth, Jason T Lee, Malachi Sheahan, Anne-Florence Rouby, Nabil Chakfé, et al. Objective evaluation of clock face suture using the objective structured assessment of technical skill (osats) checklist. In *EJVES Vascular Forum*, volume 57, pages 5–11. Elsevier, 2022.

[88] Ana Luisa Trejos, Rajni V Patel, Richard A Malthaner, and Christopher M Schlachta. Development of force-based metrics for skills assessment in minimally invasive surgery. *Surgical endoscopy*, 28:2106–2119, 2014.

[89] Yu-Hsuan Tsai, Yih-Cherng Lee, Jian-Jiun Ding, Ronald Y Chang, and Ming-Chen Hsu. Robust in-plane and out-of-plane face detection algorithm using frontal face detector and symmetry extension. *Image and Vision Computing*, 78:26–41, 2018.

[90] Munenori Uemura, Morimasa Tomikawa, Tiejun Miao, Ryota Souzaki, Satoshi Ieiri, Tomohiko Akahoshi, Alan K Lefor, and Makoto Hashizume. Feasibility of an ai-based measure of the hand motions of expert and novice surgeons. *Computational and mathematical methods in medicine*, 2018, 2018.

[91] Melina C Vassiliou, Brian J Dunkin, Gerald M Fried, John D Mellinger, Thadeus Trus, Pepa Kaneva, Calvin Lyons, James R Korndorffer, Michael Ujiki, Vic Velanovich, et al. Fundamentals of endoscopic surgery: creation and validation of the hands-on test. *Surgical endoscopy*, 28:704–711, 2014.

[92] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[93] Chien-Yao Wang. Implementation of paper - yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. `https://github.com/WongKinYiu/yolov7`. Accessed: 2024-05-09.

[94] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023.

[95] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6d: Self-supervised monocular 6d object pose estimation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part I 16*, pages 108–125. Springer, 2020.

[96] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16611–16621, 2021.

[97] Ruishuang Wang, Zhao Li, Jian Cao, Tong Chen, and Lei Wang. Convolutional recurrent neural networks for text classification. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2019.

[98] Yihao Wang, Jessica Dai, Tara N Morgan, Mohamed Elsaied, Alaina Garbens, Xingming Qu, Ryan Steinberg, Jeffrey Gahan, and Eric C Larson. Evaluating robotic-assisted surgery training videos with multi-task convolutional neural networks. *Journal of robotic surgery*, pages 1–9, 2021.

[99] Zuxuan Wu, Xi Wang, Yu-Gang Jiang, Hao Ye, and Xiangyang Xue. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *Proceedings of the 23rd ACM international conference on Multimedia*, pages 461–470, 2015.

[100] Yi Xia, Minh Nguyen, and Wei Qi Yan. A real-time kiwifruit detection based on improved yolov7. In *International Conference on Image and Vision Computing New Zealand*, pages 48–61. Springer, 2022.

[101] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[102] Zong-Ben Xu and Fei-Long Cao. Simultaneous lp-approximation order for neural networks. *Neural Networks*, 18(7):914–923, 2005.

[103] Zongben Xu and Feilong Cao. The essential order of approximation for neural networks. *Science in China Series F: Information Sciences*, 47:97–112, 2004.

[104] Shohei Yamaguchi, Daisuke Yoshida, Hajime Kenmotsu, Takefumi Yasunaga, Kozo Konishi, Satoshi Ieiri, Hideaki Nakashima, Kazuo Tanoue, and Makoto Hashizume. Objective assessment of laparoscopic suturing skills using a motion-tracking system. *Surgical endoscopy*, 25:771–775, 2011.

[105] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.

[106] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4694–4702, 2015.

[107] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

[108] Aneeq Zia, Yachna Sharma, Vinay Bettadapura, Eric L Sarin, Mark A Clements, and Irfan Essa. Automated assessment of surgical skills using frequency analysis. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part I 18*, pages 430–438. Springer, 2015.

[109] Christian Zimmermann and Thomas Brox. Learning to estimate 3d hand pose from single rgb images. In *Proceedings of the IEEE international conference on computer vision*, pages 4903–4911, 2017.