Imperial College London

FINAL YEAR PROJECT

DEPARTMENT OF COMPUTING

IMPERIAL COLLEGE OF SCIENCE, TECHNOLOGY AND MEDICINE

Automated Surgical Skill Assessment

Author:
James Calo - jam414 -
00943851

Supervisor:
Dr Benny Lo

June 26th, 2018

Submitted in partial fulfillment of the requirements for the M.Eng of Imperial College London

Abstract

Training of surgical students has been the same for a long time; an expert surgeon observes a trainee and assesses their skill. This has several disadvantages: The expert surgeon's time is very valuable, it would be more useful if they spent it doing procedures only they have the ability to do, especially with the increased demand on health care services. Furthermore the assessment is very subjective and not well standardized, each surgeon has their own style and preferences. We propose that the optimal solution to this problem is an autonomous surgical skill assessment system using Computer Vision and black box of Convolutional Neural Networks to assess the trainees. Using a simple camera we feed a video of a student performing surgery and, by tracking their hands, thumb and finger tips, we assess their skill level using metrics derived from the current standard assessments, such as the OSATS, effectively evaluating the trainee with no interaction from an expert surgeon, or anyone. This allows the trainee to not only get fair and objective assessment but also allows the trainee to practice multiple times without having to book time with an expert surgeon.

Acknowledgements

I would like to thank so many people for all their help and support with this project, especially.

To my supervisor Dr Benny Lo, without whom I would have no project, Dr Su-Lin Lee, who is the reason I fell in love with Computer Vision and the whole field of computer science combined with medicine and Mr Ben Short, without you Sir I never would've gotten into Imperial. Everything I have been able to learn since coming here is thanks to you.

Contents

1 Introduction

1.1 Motivation

In today's world with advances in modern medicine it is imperative to train surgeons to the highest standards. Currently the main method involves doctors in training being watched by a professional and graded. This has a number of problems: The professional must take the time to watch the students, this can be time consuming and so the professional may not be available often meaning less training for the students and more time away from other important tasks for the professional. Even if the professional records all of their students performing surgery, to mark at a later date, it still takes a considerable amount of time and feedback will be delayed. Moreover the scoring is rather subjective, every surgeon has their own style and so two professionals may well disagree on what constitutes a good procedure.

Whilst there has been advances in recognizing actions in both images and videos, there have been fewer in skill assessment and feedback reporting. The ability to measure ones abilities and provide coherent and useful feedback is very useful, trainee surgeons would be able to practice without supervision and get constructive feedback so as to reduce the amount of time a professional would need to observe. Furthermore the assessment would be objective since the system would be following a set of rules. We could even extend this to teach a machine how to follow the same actions, entering the world of medical robotics.

1.1.1 OSATS

The OSATS (objective structured assessment of technical skills), are marked using a Likert scale [1]from levels 1-5. However only levels 1, 3 and 5 actually have values and even they aren't very explicit e.g. "majority of knots placed correctly". Worse still some are not even distinctive, "Knew all important steps" vs "Familiarity with all steps".

The OSATS contain seven diverse components, for example "instrument handling' (IH) is to do with the fluidity of the movements where as "respect for tissue' (RT) focuses on the object of the operation.

All in all the OSATS do not seem to be all that objective and as shown in [1] there is a poor correlation between subjective and objective evaluations. They take a considerable amount of time and are challenging to mark given the lack of detail in the criteria. This is especially noticeable when compared with A-Level and GCSE mark schemes.

1.1.2 Currently proposed methods

There are currently several different methods out there to try to automate the OSATS and other training methods; many focus on specialized mechanical devices and measure surgical proficiency using metrics derived from these mechanical devices. There is also research looking at using different types of sensors, both on the tools and the subjects, although these can be intrusive. However cameras are a category of sensors too and this form of sensor is the area this paper will focus on.

Whilst there seems to be growing excitement in using computer vision in order to assess skill, not just in medical contexts but also in sports and photography, many require the use of either minimally invasive equipment, specially coloured equipment such as gloves or suturing thread or special markers on the standard medical gloves and tools, which can be a problem due to the sterile environment necessary for some surgical tasks.

1.2 Proposed idea

The objective of this project is to create a vision system that causes little to no change to the user, this would also allow extension into a real operating room since there is no change necessary to the surgeon. Essentially the aim is for the trainee to do the assessment exactly the same as if it was being recorded for their supervisor; however instead, at the end they would get given their results by the program, consisting of a black box of Convolutional Neural Networks, which will provide results in real or near real time. Once the student has stopped suturing the system will automatically stop recording and display the results. This could include annotated feedback for improvement and possibly a “Highlight reel” of what they did well and a negative reel of what they did poorly so they can improve. This is similar to the work in [2] which was applied to two Olympic sports, diving and figure skating. However we will focus solely on surgery which has its own set of challenges. For example surgery is rarely whole body based

[1]Typical Likert scale: Strongly disagree, Disagree, Neither agree nor disagree, Agree, Strongly agree

unlike diving and skating. Furthermore data also comes from interactions with the substitute for the patient where as diving and skating, along with other sports, predominantly focus on the performer themselves.

Initially we aim to focus on suturing and knot tying, these are both simple and essential skills all surgical students must learn. Whilst it may be possible to then extend this to incorporate other surgical procedures, it is far more important to master accurate skill assessment as this would then form the basis of a general assessment framework, on which the process of learning to assessing another procedure would become far simpler.

The most important aspect of this project will be data. Many surgical skill assessment systems have gotten accurate classification results with data sets (people performing suturing/knot tying) in the tens such as [1] which had only 18 participants but got results above 85% accurate for its classification. However since we are not only classifying but also returning feedback, a lot more data will be needed. That said expert data is exceptionally hard to come by as professionals are busy enough, the whole aim of this project is to reduce that. Fortunately suturing is a simple task and so I should be able to record myself multiple times in order to create the dataset, improving as I do so creating the different skill levels required in a dataset.

2 Background

2.1 Surgical Training

2.1.1 Overview

The core of surgical training has been the same since 1889 when Sir William Halsted introduced a residency training system based on graded responsibility at Johns Hopkins Hospital [3]. It is generally accepted that skill levels amongst surgeons varies due to teaching, training and experience and that there is a causal link between surgical skill level and clinical outcome [4, 5].

Furthermore there appears to be a general trend in the increase in the number of surgeries each year. In fact according to the U.S. National Center for Health Statistics; 40.3 million inpatient procedures were performed in the US in 1996, which was increased to 45.9 million in 2005 [6].

Moreover from this population the Agency for Health care Research and Quality discovered that in the year 2000 there were more than 32,000 surgery related deaths, costing 9 billion dollars and accounting for 2.4 million extra days in the hospital [7].

Given these facts there are, unsurprisingly, consequential financial pressures on hospitals caused by the increasing costs of malpractice insurance, surgical equipment, personnel salaries and the cost of training residents [4]. On top of all this, there are labor laws limiting a resident's work week to 80 hours [8].

Therefore surgical students must learn a growing number of techniques and technologies while having less time for hands-on experience, which when combined with the fact hospitals have fewer funds to spend on increased training, is no mean feat.

In fact surgeons exiting the UK surgical training programme are now further lacking in experience and subsequently require more guidance during the first few years of their careers [9], [10]. Furthermore [11] found that only 34% of surgical students believed they were receiving enough training in basic skills.

This becomes problematic if trainees do not have the natural potential to reach adequate competency in their specialty within a few years. In fact it has been argued that most surgeons do not ever become true experts and in order to gain expertise in surgery there is a need for deliberate practice [12]. Consequently, a surgeon's progression is contingent on a valid, but unstandardized, selection process that and is almost purely based on academic achievement, personal knowledge of the candidates and subjective referrals.

However there are many factors that determine the skill level of a surgeon such as knowledge of the task, judgment / decision making and manual dexterity. Whilst the first two are currently taught in a classroom and evaluated using written and oral exams [4], methods to standardize technical skill level for teaching and assessment has been difficult [13]. Many surgical training schemes are based upon Sir William Halsted's training method of "see one, do one, teach one" where students perform whilst being observed by expert surgeons [4], this however is time consuming and is by no means a standardized approach to of assessing surgical skill [14].

The Intercollegiate Surgical Curriculum Programme (ISCP) attempts to standardize surgical training by dividing the assessment into four domains: knowledge, judgment, technique and professionalism [15]. They specify that all assessment methods during training should meet the following criteria; they are to be valid (the complexity of the the given tasks increases in proportion with the system), reliable (multiple measurements lead to the same outcome), feasible (the results should be available within 5 to 10 minutes), cost effective, be open to feedback and have an impact on the student's learning.

If it is possible to combine surgical training with a valid, reliable and objective assessment system we could then ensure that surgeon's in training are fairly evaluated in a standardized manner ensuring that hospitals can be confident that each trainee surgeon finish training with basic competence which will result in a reduction of critical medical errors.

It was found that 85% of students were interested in innovative surgical training systems, however it was noted that there was a lack of time or resources for a training facility. Surprisingly there have been few studies to addressed feedback in training methodologies, this has caused difficulties for improvement of surgical training and performance [16]. Whilst opportunities for learning through work with patients has diminished, there has been an increase in interest in laboratories with a formal curriculum, specifically designed to teach surgical skills. This model of surgical education focuses on teaching basic surgical skills using models and simulators, some of which are discussed later, with the aim of better preparing students for the operating room [5], [17, 18, 19, 20, 21].

The Fitts-Posner Three-stage Theory of Motor Skills Acquisition.*			
Stage	Goal	Activity	Performance
Cognition	Understanding the task	Explanation, demonstration	Erratic, distinct steps
Integration	Knowledge becomes motor skill	Deliberate practice, feedback	Increased fluidity, less interruptions
Automation	Perform quickly, smoothly and precisely	Performance with little cognitive input, refine performance	Continuous, fluid, adaptive

Table 2.1: *Adapted from [5]

These training techniques are based upon the way in which motor skills are built and expertise is developed. Fitts and Posner's three-stage theory of motor skill acquisition is widely accepted in both the motor skills literature and the surgical literature [5, 22, 23]. In the cognitive stage, the student learns the theory of the task; performance is erratic, and the procedure is carried out in clearly distinct steps. Using the process of surgical knot tying as an example, in the cognitive stage the learner must understand the mechanics of knot tying, how to hold the tie, how to place the throws, and how to move the hands. With practice and feedback, the student will reach the integrative stage, in which the knowledge of the procedure is translated into motor behavior. The student still has to actively thinks about how to move the hands and hold the tie but is able to execute the task more fluidly, with fewer interruptions. Finally is the autonomous stage, in which practice gradually results in a smooth performance. The student no longer consciously thinks about how to perform the task and can concentrate on other aspects of the procedure. However this still doesn't aid in the task of objective skill assessment, only in improving the quality of training.

2.2 Grading Systems

2.2.1 Objective Structured Assessment Of Technical Skill (OSATS)

The Objective Structured Assessment of Technical Skill (OSATS) is widely agreed upon as the gold standard for evaluating surgical skill and has been successfully adapted to many surgical specialities [24], [25, 26, 27, 28, 29]. The OSAT requires students to perform a series of standardized surgical tasks on inanimate models under the direct observation of an expert [5]. Examiners score students using two methods. The first is a task specific check-list consisting of

10 to 30 specific surgical manoeuvres that have been deemed essential elements of the procedure. The second is a global rating form, which includes five to eight surgical behaviours, e.g. Respect for Tissues and Economy of Motion.

There has been research already into the reliability and validity of the OSATS. Two different forms of the OSATS were evaluated, one using live anaesthetized animals and the other using bench model simulations. The evaluation found that bench models have the same level of difficulty and are equivalent to using live animals in the testing of a trainee.

Unfortunately, even though the OSATS are widely accepted as the gold standard for assessment, there are many prohibitive limitations. The examinations are expensive and still there is the issue of the subjectivity of performance measures [30], [31], [32]. The high cost of the examination comes from multiple factors including the cost of personnel, such as the professional surgeon and the operators (either for the operating room, the simulator or whatever system is being used) and the materials used.

This does not even begin to take into account the cost to the surgeon examining the trainees; whilst examining trainee surgeons they are unavailable to perform or instruct in possibly more complicated and/or vital surgeries. When the aim is to create a fully integrated and regular assessment system, to evaluate a surgeon's training, it is not feasible to require a large number of expert surgeons for such a long period of time, even shorter periods are inadvisable given the limited number of truly expert surgeons and the already high demand for their time elsewhere.

2.2.2 Procedure Based Assessment (PBA)

Procedure Based Assessment (PBA) is a modification of the OSATS which is currently being used by the ISCP for training in all surgical specialities[15]. This system uses a surgical consultant or someone of similar skills to assesses the professionalism and functional skills of the trainee. Then a decision is made as to whether the trainee has performed to a satisfactory standard. However the majority of the time the PBA is assessed by the trainee's clinical supervisor; this causes the assessment to firstly remain subjective and secondly, since the examiner has prior knowledge of the trainees style and in fact trained the trainee themselves, can cause the assessment not to be based completely of the examination itself.

2.2.3 Global Rating Scale (GRS)

The Global Rating Scale (GRS) was created in order to assess surgical skill in a laboratory environment. It has been shown to reliably assess performance on arthroscopic tasks [33], [34]. The GRS system consists of a blinded task-specific check-list and is assessed by an experienced clinician. Paper methods, which encompass skill assessment systems such as this (GRS) and check-list bases systems, have been widely used in real operating theatres [35, 36, 37, 38, 39, 40].

2.2.4 Fundamentals of Laparoscopic Surgery (FLS)

The Fundamentals of Laparoscopic Surgery (FLS) system is aimed at teaching and assessing laparoscopic skills, it measures both the trainee's cognitive and technical skill as well as providing an opportunity for trainees to practice basic laparoscopic skills [41]. In the US it is even considered the gold standard for both teaching and assessment of a trainee. FLS involves 5 standard tasks, each of which are evaluated by two metrics, time and accuracy [42].

2.3 Automated Skill Assessment Systems (Motion Analysis)

Currently there are a number of systems being used to try and autonomously assess trainee surgeon's skill level. These systems are autonomous to varying degrees, they all obtain some data about how the trainee performed the surgical task, however some still require an assessor to look at the data.

Many of these systems fall under the "Motion analysis" category. These systems attempt to objectively measure the trainee's technical dexterity and monitor the learning curves for various surgical procedures [43], [24], [44, 45, 46]. These systems clearly have an advantage over grading systems that require the over watch of a trained professional such as the OSATS and the FLS systems: They don't require a supervisor at all! This means that expert surgeons can spend less time assessing the trainees and focus on teaching and performing complicated surgeries only they can do. Furthermore the automated assessment process is more objective, it returns raw data only, and as such is a fairer way to assess surgeons [24].

The need to have expert surgeons observe and assess trainees, giving up time that there may be a greater need for elsewhere, is a huge hurdle in the hunt for an improved way to assess trainees. Moreover the need for real time assessment, or close to (ISCP states that results should be available in 5 to 10 minutes) has been recognized [47].

It has been suggested that instead of the expert surgeons watching a trainee perform the assessment in person they can instead watch a recording of the trainee performing the same task. This has many advantages, for example the expert surgeon can assess multiple trainees without the need to wait for the assessment to be set up each time; they can watch the recordings back to back, allowing them to assess more trainees in a certain time period. They can also re-watch any elements of the recordings that may need closer analysis. Finally the assessment process is also essentially blinded, the recordings can be taken without showing the identity of the trainee and the assessor need not know the identity of the trainee, as is the case with almost all written examinations. However, despite all these advantages, it is still a time consuming process and prone to subjectivity, each surgeon has their own style and preferences in styles.

Systems that analyse dexterity from the trainee's hand movements are quite popular, alone they can perform solid analysis of requirements that are common to many grading schemes; for example three of the OSATS categories (Time and Motion, Instrument Handling and Flow of Operation) are straight forward to assess with such a system. However they cannot assess other important requirements, again using the OSATS as an example, the following categories cannot be easily assessed using hand motion analysis systems alone: Respect for Tissue, Knowledge of Instruments, Use of Assistants and Knowledge of Specific Procedure.

A combination however, of motion analysis followed by a performance check-list assessed by a qualified professional, though not necessarily an expert surgeon, is a possible solution that would have the benefits of both the recorded assessment system described previously and the motion analysis, resulting in a more objective manner that is also less time and resource, the expert surgeon, consuming.

The following is a sample of the main systems for skill assessment that use motion analysis:

2.3.1 Imperial College Surgical Assessment Device (ICSAD)

Hand motion analysis performed by the Imperial College Surgical Assessment Device (ICSAD) [48] has been found to be an effective measure of of technical skill and surgical dexterity for both laparoscopic [49] and open surgical simulation [45] [50, 51] and has been shown to have good concordance with OSATS scores [5] [52].

This system uses an electromagnetic tracking system (Isotrak II)and consists of an electromagnetic field generator and two 10-mm electromagnetic trackers, attached to the dorsum of each hand at the midshaft of the third metacarpal, which then are further secured to each hand by the latex gloves worn by the user of the system. The system obtains the Cartesian coordinate information from each tracker a 20Hz frequency and 1mm resolution. Individual movements of the hands are found by a change in velocity and positional data is converted into parameters, including the number of hand movements, the amount each hand moved and the time taken to complete the procedure, and is used to measure performance [47], [45].

Unfortunately this system is limited to ex vivo benchtop models, described in the next section, due to the fact neither the extraneous wires or markers can be used on the surface of the gloves in live surgery [4]. Moreover this system is logistically difficult to implement in the operating room [30], mainly because the electromagnetic field is prone to disturbance by ferrous materials.

2.3.2 Advanced Dundee Endoscopic Psychomotor Tester (ADEPT)

The Advanced Dundee Endoscopic Psychomotor Tester (ADEPT) [4] system has a potential advantage over the ICSAD since it actively compares the performance of expert and novice surgeons and as such performs an aptitude test instead of just returning evaluation metrics [52]. It performs motion tracking by reflecting infra-red light off sensors attached to the surgeon's arm. The system then applies trajectory analysis in order to obtain the positional data of these sensors.

However, there are many limitations to this system; there is the issue of line of sight, as is a problem with all optical tracking systems, so it cannot be used outside of simulations. Moreover simultaneous limbs cannot be tracked due to overlapping signals [4] and the difficulty of the tasks that can be assessed is limited.

Although the ADEPT system is perceived to be a valid system for assessing psychomotor skills [53], participants in the study did not hold the system in high regard; Furthermore the endoscopic setting used by this system is not similar laparoscopic environment used in practice.

2.3.3 Sensor Gloves

There is some development into a wireless sensor glove with the aim to assess surgical skill by analysing hand movement made while performing laparoscopy. It uses hidden Markov models (HMM's), described later, in order to place the sensors in the optimal locations for a given set of gestures [32]. However the current size of the sensors is an issue.

HMM's are well suited to coping with spatiotemporal features [32] and can find patterns based of the location of the sensors in order to learn the optimal placement of the sensors to improve the hand gesture classification.

2.3.4 ProMIS

The ProMIS simulator [4, 54] uses a passive tracking system. The instrument's movements are obtained from three separate cameras that capture the video of the internal movement of the laparoscopic instrument from three different angles. This design allows for measurement of the motions in 3D space. Standard laparoscopic instruments are covered with two strips of yellow tape which act as markers for the camera tracking system. However the tracking system is situated in a large mannequin and is therefore not easily portable. Whilst it comes with both a real and a virtual environment, it cannot be used in an actual operation, however as discussed above, in the training of junior surgical residence this is not such an issue. It also provides force feedback.

2.3.5 Zebris Ultrasound System

The Zebris Ultrasound System [54] uses 3D ultrasound measurements in order to track the 3D coordinates as well as the rotation of miniature ultrasound transmitters attached to the surgical instruments. It achieves this my measuring the relative location of these transmitters according to three microphones that have a fixed location. Since the ultrasound transmitters can be sterilized this system can be used in the operating room as well as with both box trainers (physical models) and virtual reality trainers. Due to the use of real surgical tools natural physical feedback is present and since the system relies on these sensors and microphones alone, along with the tools to attach them to, it is portable.

2.3.6 Hardware interfaces

Laparoscopic Surgical Workstation, Virtual Laparoscopic Interface and Laparoscopic Impulse Engine are three well known interfaces for use with laproscopic virtual simulations [54]. These are fully instrumented tools which allow realistic controls for use with the virtual environment. Since virtual environments are cost effective compared to higher fidelity systems and lower fidelity systems do not hinder teaching as much in junior residence, the use of realistic tools further enhances the usefulness of this type of training.

2.3.7 Combinations of motion analysis with alternative analysis systems

Combining other sensing systems with motion analysis could have an increased effect on the accuracy of automated surgical skill assessments and further reduce such assessment from subjectivity. For example by combining eye tracking with motion analysis one study has shown that, surgical skill assessment accuracy was increased by 13.2% and 5.3% for expert and novice skill respectively [55]. Using both types of analysis lead to important information about hand-eye coordination being found. This enabled the system to evaluate OSAT metrics, such as Knowledge of Instruments, which would not be possible by motion analysis alone. This increase in the number of categories that can be autonomously determined by the system represents a step further in removing the subjectiveness of skill assessment, when assessed by another person, and decreases the overall time which requires an expert to assess the trainee.

2.3.8 Evaluation Metrics Used

The following table shows the evaluation metrics used by some of the systems described above as well as the OSATS for comparison:

Types Of Simulators.*			
Simulation	Advantages	Disadvantages	Best Use
Bench models	Cheap, portable, reusable, minimal risks	Acceptance by trainees; low fidelity; basic tasks, not operations	Basic skills for novice learners, discrete skills
Live animals	High fidelity, availability, can practice hemostasis and entire operations	Cost, special facilities and personnel required, ethical concerns, single use, anatomical differences	Advanced procedural knowledge, procedures in which blood flow is important, dissection skills.
Cadavers	High fidelity, only "true" anatomy simulator currently, can practice entire operations	Cost, availability, single use, compliance of tissue, infection risk	Advanced procedural knowledge, dissection, continuing medical education.
Human performance simulators	Reusable, high fidelity, data capture, interactivity	Cost, maintenance, and downtime; limited "technical" applications	Team training, crisis management.
Virtual reality surgical simulators	Reusable, data capture, minimal setup time	Cost, maintenance, and downtime; acceptance by trainees; three dimensions not well simulated	Basic laparoscopic skills, endoscopic and transcutaneous procedural skills.

Table 2.2: *Table from [5]

2.4 Models

There are a variety of models to use when training surgeons. For the simulation of living human tissue and anatomy, inanimate models, virtual reality, live animals and human cadavers are used. For critical-incident and team training high-performance patient simulators can also be used. Although human cadavers most closely approximate reality, their cost, limited availability and the poor compliance of cadaveric tissue limits their use. Using live animals is also difficult due to the ethical concerns, high costs and the need of specialized facilities. In contrast, inanimate models are safe, reproducible, portable, readily available and generally more cost-effective than animals and cadavers.

Advances in virtual reality technology have a huge potential for enhancing surgical skills training, and many virtual reality systems are now commercially available. Virtual reality provides the opportunity for very detailed feedback and may allow for more subtle measurement of the student's performance than would be possible to measure by an expert observer [51]. Furthermore measures of precision and accuracy as well as error rates can be calculated easily. Two

Types Of Simulators.*			
Simulation	Advantages	Disadvantages	Best Use
Bench models	Cheap, portable, reusable, minimal risks	Acceptance by trainees; low fidelity; basic tasks, not operations	Basic skills for novice learners, discrete skills
Live animals	High fidelity, availability, can practice hemostasis and entire operations	Cost, special facilities and personnel required, ethical concerns, single use, anatomical differences	Advanced procedural knowledge, procedures in which blood flow is important, dissection skills.
Cadavers	High fidelity, only "true" anatomy simulator currently, can practice entire operations	Cost, availability, single use, compliance of tissue, infection risk	Advanced procedural knowledge, dissection, continuing medical education.
Human performance simulators	Reusable, high fidelity, data capture, interactivity	Cost, maintenance, and downtime; limited "technical" applications	Team training, crisis management.
Virtual reality surgical simulators	Reusable, data capture, minimal setup time	Cost, maintenance, and downtime; acceptance by trainees; three dimensions not well simulated	Basic laparoscopic skills, endoscopic and transcutaneous procedural skills.

Table 2.3: *Table from [5]

prospective trials have demonstrated that students who have been trained on low-fidelity, not very lifelike, virtual reality models, e.g. laparoscopic box trainers, make fewer intraoperative errors when performing a laparoscopic cholecystectomy than students who have not had simulation training [56, 57]. High-fidelity (lifelike) virtual reality models are also available for training in procedures such as colonoscopy and carotid artery stenting [58]. An FDA (Food and Drug) administration panel recommended the use of virtual reality simulation as an integral component of a training package for carotid artery stenting [59]. Unfortunately the higher the fidelity, i.e. the more realistic the model, the more expensive and so there are many studies needed to determine the worthwhile of using such systems. Fidelity may be less important at relatively junior levels of training however. For example, when a group of medical students was trained using a high-fidelity-video endoscopic urology system and another with the use of a simple bench model, the two groups showed the same improvement in performance and both showed more improvement than the control group (given didactic training) [60]. Furthermore, among first-year surgical residents, improvement in a variety of open procedures has been shown to be the same regardless of whether low-fidelity bench models or cadavers have been used [61].

2.5 Machine Learning Algorithms

There are many famous Machine learning (ML) algorithms for use with vision systems. The following is a basic overview of some of the most common ones.

2.5.1 HMMs

Hidden Markov Model's are statistical Markov models that assumes the Markov property, that future states of the model depend only upon the current state and not of those of past state, that is to say it is a memoryless (stochastic) process. For a system to be a HMM it needs to be autonomous and have the state be only partially observable, this often manifests as not being able to directly observe the state changes but can observe their affect. For example say there are three possible state, {Go to the pub, Go clubbing, Go home} which depend upon a coin toss with a trick coin. If the initial ”start” probabilities of the coin toss are known (since there is no previous state a beginning state must have know probabilities) to be 0.6 for heads and 0.4 for tails and the probability that the previous state {Heads, Tails} will be repeated is 0.7 and change is 0.3 (the transition probability). Then, assuming knowledge of the probability of each state, {Go to the pub, Go clubbing, Go home} given the outcome of the coin toss (the emission probability), by observing this state one can use this data (Modeled as an HMM since the actual coin toss is unobserved / hidden) to statistically find the result of the coin toss.

2.5.2 BoF/ BoW

Bag of Features/Bag of Words for computer vision tasks comes from Bag of Words for natural language processing. In the original context, text was in an unordered collection and each unique word would have a count of occurrences. This then forms a histogram which is used to generate features for learning.

For BoF/BoW for images there are these main steps: Firstly to extract the features, this can be done by feature detection algorithms such as Harris corner detection, Shi-Tomasi corner detection or scale-invariant feature transform (SIFT). Given these features we then need to represent them, if we use SIFT we are left with a 128 dimensional vector. Given these features it is then necessary to convert them to ”words” so we can form a histogram as with the original method. One popular way to do this is use the K-means clustering algorithm, discussed below. Each "word" used is defined to be cluster centre and by applying k-means each feature representation will be assigned the word corresponding to their cluster. This word is also known as a codevector. The set of "words" (also called the vocabulary) is known as a codebook. The image, or frame of a video, can then be represented as a histogram of these "words"/codevectors. It is vital to choose the correct vocabulary size since if this is too small there will not be enough words to be representative of all the features. However too many words will cause overfitting, a big issue in Machine Learning where the model fits the training data so well but performs poorly on data not used in training.

3 Convolution Neural Network: Hand Tracking

In this chapter we will cover the creation of the Convolutional Neural Network (CNN) that is used in this project in order to track the hands of the trainee surgeon during assessment. We wanted to actually contribute to the field of hand tracking in general and as such have moved away from more common approaches. The input to the CNN is a two dimensional grayscale image, as opposed to the majority of current systems out that use coloured images, some of which also rely on three dimensional images.

Please note that a lot of the Tensorflow references in this report apply to the lower level counterparts of the api's we used and so the parameters may appear different. In general we used features from the tf.layers module but often the technical details are documented in the equivalent tf.nn module.

3.1 CNN Inputs

3.1.1 Dataset Generation

The final dataset comprises of 8,174 images, 8,000 for training and 174 for evaluation. Originally there were 11,000 images, 10,000 for training and 1,000 for evaluation. This dataset comprised images of no hands, one hand (with and without a glove) and both hands (performing suturing and gloved); however such a varying dataset with multiple noisy backgrounds cause terrible detection in unseen examples (overfitting). For a more detailed explanation see the Evaluation Chapter 7. Therefore the final dataset contained images solely of the suturing and knot tying procedure which occasionally included hands going in and out of shot, meaning that the background was always the same and hands naturally went in and out of focus thereby generating data with both, one or no hands. We generated the dataset using a vision system to locate the hands in each image and manually assessed the validity of the segmentation, we reviewed approximately 16,000 images one by one to generate both datasets (the initial one of 11,000 and the final dataset of 8,174 images). The vision system used to generate the dataset is documented in chapter 4

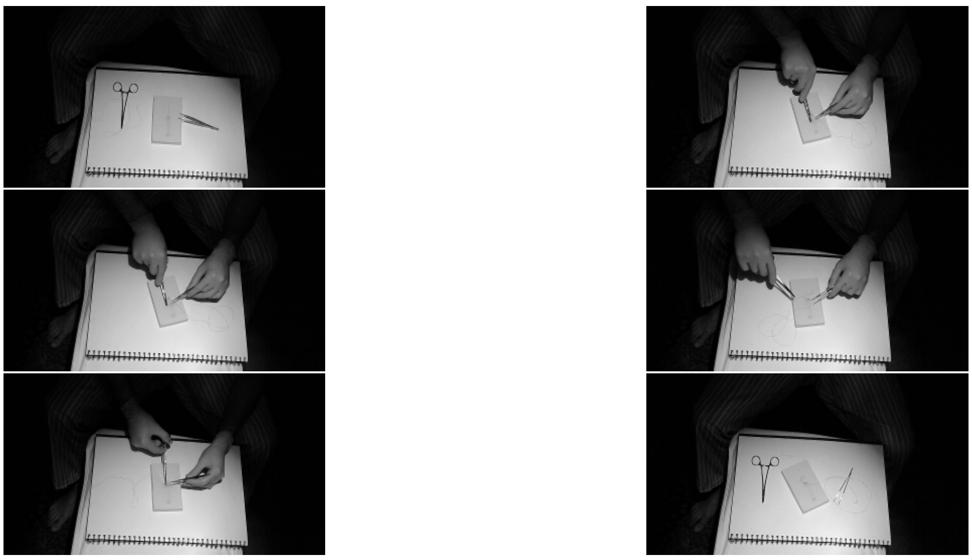


Figure 3.1: Input Images For The Hand Detection CNN

3.1.2 Feature/Input Set

CNN's, when used to learn vision based tasks, typically take each pixel channel in the image as an input. This results in the input space being equal to $ImageHeight * ImageWidth * NumberofChannels$ where the width and height are in pixels and the channels is usually either three or one, three channels is typically for colour images (i.e. RGB, HSV etc) and one channel is for grayscale (four channels is common with three dimensional images for example, where the fourth dimension is depth of the pixel). This limits the size of the image that can be used to train a CNN since large images would necessitate far too many input neurons for most computers. Typical image sizes are 32x32 pixels and 28x28, for example CIFAR-10 and MNIST respectively, however both those examples are for classification and as such only return a category, either a plain integer or a string category name identified by an integer. Therefore the input image size is not as important; the correct classification is not dependent on the image size itself, a picture of a dolphin is a picture of a dolphin whether it's at 4K resolution or 32x32 pixels. Where as for the style of regression used for this task image size was crucial. Whilst hand tracking as a classification task was more appealing, since we could get information about the probability of a hand's location at a certain pixel, it was more natural to treat it as a regression problem, returning the top left coordinates in Cartesian space and the width and height of the rectangular bounding box surrounding the hand. Therefore the image size was important since up-scaling can cause the hand locations to get re-mapped incorrectly.

As stated above this system uses gray scale images, which makes it more useful than methods which require colour since it is now colour invariant and as such trainee's using the system need not be concerned with the colour of the gloves being used. However there is an additional advantage for the CNN, only having one channel reduces the size of the input neurons resulting in faster training and allowing for bigger input image sizes. The final image size chosen was 320x180, which is small enough to work well even on a laptop. It also set the required aspect ratio at 16:9, which is the standard aspect ratio nowadays, webcams and modern phone cameras tend to show in 720p or 1080p both of which are 16:9 aspect ratio. This is important since downscaling the image is simple if the aspect ratio is the same, otherwise padding and/or cropping would be required, so as to not distort the original image when scaling it to fit the input size, which could cause the hands to be cut out (cropping) or reduce the size of the hands in relation to the size of the image (padding). Since we wanted the input images to be captured easily, without the use of specialized equipment, using the aspect ratio that is the standard for both mobile phones and webcams means that capturing the trainee performing surgery requires little concern. The secondary aim of this system, after being an automated surgical skill assessment system, is to require as little change to the standard system as possible; all the focus should be on performing the exercise not to how to integrate this system into the exercise.

3.1.3 Labels/Outputs

For the labels, and by extension the outputs, it was decided to use a 2x4 matrix (i.e. two vectors of length four) where each column represents the bounding box of one hand (or all zeros for no hand). The representation is as follows, the x and y coordinate of the top left of the bounding box, the width of the box and the height. E.g.

$$\begin{bmatrix} X_{top\ left}^{hand\ 1} & X_{top\ left}^{hand\ 2} \\ Y_{top\ left}^{hand\ 1} & Y_{top\ left}^{hand\ 2} \\ Width^{hand\ 1} & Width^{hand\ 2} \\ Height^{hand\ 1} & Height^{hand\ 2} \end{bmatrix}$$

Whilst it would've been preferable to have made this a classification task, where each pixel either belongs to the hand or not, since then we could have the probability of whether a given pixel belongs to the hand, this way the output space is far smaller, 8 elements instead of the same as the input space. Furthermore we can use padding of the bounding box (add two pixels on all sides of the box) in order to account for possible inaccuracy, as an alternative to using pixel probabilities, and we can average out the output over a set number of frames in order to improve stability between frames. Since this system is mainly to simplify the task for the next system, Tip Classification, it doesn't matter if this system over predicts, whereas under prediction would be a problem. Finally it allows for far easier manipulation of the label given the randomizations applied to the training data and if the output data is used for a different use to the standard pipeline for this system, it is more customizable to allow for this.

3.2 CNN Model

The Convolutional Neural Network used for this Hand Tracking task contained 5 layers in between the input and output; however for some of the intermediate models used when designing the final one for use, two extra layers where used. These were eventually removed as they did not largely change the output results and slowed down the training procedure, however they may still be useful if the design of the overall system changes and so will be discussed nonetheless.

Figure 3.2 shows the intermediate layers between input and output, note the arrows skipping over the two optional Local Response Normalization Layers used to show that the layers nay not be used.

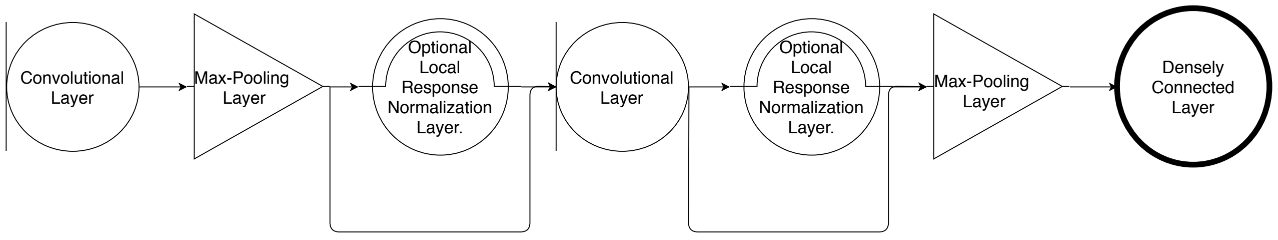


Figure 3.2: Hand Tracking CNN Model Layout

3.2.1 Input Preprocessing

Before the input image is sent to the CNN for training it is first normalized, sometimes referred to as standardized as it is in TensorFlow's api. This linearly scales the input image to have zero mean which helps to generalize the training data in order to reduce effect of brightness and contrast that would differ from scene to scene, where scene is the sequence of images being given as input to the CNN.

Normalization is applied by replacing each pixel with the following function:

$$x_{new} = \frac{(x_{original} - \mu)}{\sigma_{adjusted}}$$

where μ is the mean of every pixel, $\mu = \frac{1}{N}\sum_{i=1}^{N} pixel_i = \frac{pixel_1 + pixel_2 + \cdots + pixel_N}{N}$, in the image and $\sigma_{adjusted}$ is the following function:

$$\sigma_{adjusted} = \max(\sigma, \frac{1}{\sqrt{\#pixels}})$$

where σ is the standard deviation of every pixel in the image, $\sigma = \sqrt{\frac{\sum_{i=1}^{N}(pixel_i - \mu)^2}{N-1}}$, but is capped away from zero in order to prevent division by 0 in the case of uniform images. Max picks the largest of its two inputs and #pixels is the number of pixels in the image, in this case $\#pixels = 320 \times 180 = 57,600$

For some of the older models, during training only, the images had two possible preprocessing transformations, being flipped horizontally and being flipped vertically, each of which were applied with 50% probability one after the other which allowed for increasing the observed size of the dataset. This caused some issues by increasing the variance in the data and including images that are unlikely to ever occur such as upside down hands. A solution for this was to include it after tens of thousands of steps of training over the original dataset had already occurred.

3.2.2 Convolutional Layers

Convolutional layer's take as input a three-dimensional array with x two-dimensional matrices, sometimes referred to as feature maps, each of size m by n. In this case x is 1, the number of channels, m is 180, the height of the image, and n is 320, the width of the image. Keep in mind that whilst one usually refers to an image as width by height, the height on an image is actually the number of rows and the width is the number of columns; since matrices are referred to in row by column form we will treat images in the same way.

For example with a standard colour, RGB, image of size 3 by 2 the input would look as follows:

$$\begin{bmatrix} \begin{bmatrix} Red_{1,1} & Red_{1,2} & Red_{1,3} \\ Red_{2,1} & Red_{2,2} & Red_{2,3} \end{bmatrix} \begin{bmatrix} Green_{1,1} & Green_{1,2} & Green_{1,3} \\ Green_{2,1} & Green_{2,2} & Green_{2,3} \end{bmatrix} \begin{bmatrix} Blue_{1,1} & Blue_{1,2} & Blue_{1,3} \\ Blue_{2,1} & Blue_{2,2} & Blue_{2,3} \end{bmatrix} \end{bmatrix}$$

Where for $Channel_{r,c}$, Channel is the image channel, r is the row and c is the column.

The output is another three-dimensional matrix, an m by n matrix where each value is a vector of length k, where k is the number of filters to use. Technically the output need not be m by n (the same dimension as the input image) since convolution kernels treat the edges of images differently depending on what is specified, however in this case it is always the same dimensions as we specified the Convolutional Layer uses the same padding for output as was received by the input. Each filter has a size or kernel of dimension u by v by w, where u and v are the dimensions in two-dimensional space and w is like a skip for multi channel images, i.e. with an RGB image if w is 2 then only the red and blue channels get filters applied and G is skipped, in this case we used a kernel/filter size of 5 by 5 (by 1 though this is implied since the input image only has one channel).

An example output based on the example input described above with 2 filters (i.e. k = 2) would be:

$$\begin{bmatrix} \begin{bmatrix} F^1_{1,1} & F^2_{1,1} \end{bmatrix} & \begin{bmatrix} F^1_{1,2} & F^2_{1,2} \end{bmatrix} & \begin{bmatrix} F^1_{1,3} & F^2_{1,3} \end{bmatrix} \\ \\ \begin{bmatrix} F^1_{2,1} & F^2_{2,1} \end{bmatrix} & \begin{bmatrix} F^1_{2,2} & F^2_{2,2} \end{bmatrix} & \begin{bmatrix} F^1_{2,3} & F^2_{2,3} \end{bmatrix} \end{bmatrix}$$

Where for $F^n_{r,c}$ F is the filter value, r is the row, c is the column and n is the filter number.

The Convolutional layer computes the output element F using the equation $F^s = W^s_{r,c} \star x_{r,c} + b^s$

where s is the filter number and is between 1 and k, W is the weight, r and c is are row and column respectively, x is a pixel, $\star$ is a function that does the two-dimensional discrete convolution between W and x and b is the bias, although bias is not used in this model (bias = 0).

Essentially a Convolutional Layer tries to learn the optimum weights, and bias but not in this model, in order to create the optimal convolution kernel, of dimensions given by the kernel size (in this case 5 by 5).

The model used for the task of hand tracking used two Convolutional Layers as shown in the diagram at the beginning of this section 3.2. Both layers used a kernel size of 5 by 5 and didn't skip any channels (i.e. kernel size was 5 by 5 by 1). The first layer taking the model's input directly returned 32 filters and the second Convolutional Layer used after the first Max_Pooling Layer, or in some earlier cases the Local Response Normalization Layer, applied 64 filters. These are rather standard number of filters, the CNN tutorial for MNIST on Tensorflow used the same number of filters and the CIFAR-10 tutorial used 64 for both filter numbers. 32 was chosen for the first layer since 64 didn't make much difference and was slower to train and 128 filters was far too many.

3.2.3 Pooling Layers

Pooling layers are used to down sample the input image. Since the first Convolutional Layer creates 32 filters it becomes imperative that we reduce the number of inputs/neurons to the next stage or else we will have too many inputs and the system will either crash or train far too slowly.

Max Pooling is one of the pooling strategies that reduces the number of neurons. As the name suggests it looks for the maximum value and keeps that whilst discarding the rest. A pooling layer has three important parameters, the size, the stride and the padding. The padding, as with the Convolutional Layer was set to be the same so that the aspect ratio is the same, for cases where the pool size doesn't fully fit into the actual input size. The two Pooling Layers used have a pooling size of 4 by 4 and 5 by 5 respectively and both have the same stride as pooling size, i.e. the pooling only considers each area of pool size once. Therefore the first Pooling Layer, which occurred after the first Convolutional Layer, has an input of 180 by 320 by 32, Image Height by Image Width by Number Of Filters, and so quarters this to 45 by 80 by 32, since Max Pooling doesn't affect the number of filters, it just chooses the maximum filter for each section. The second Pooling Layer occurred after the Second Convolutional Layer, or after the Local Response Normalization Layer however since that doesn't change the number of neurons it is inconsequential to the Pooling Layer. It therefore receives an input of 45 by 80 by 64, since the second Convolutional Layer has 64 filters, and returns a size of 9 by 16 by 64.

An example of a 2 by 2 Max_Pooling Layer with a stride of 2 operating on an input of 4 by 4 by 1 would result in the following 2 by 2 by 1 matrix:

$$\begin{bmatrix} 1 & 5 & 12 & 14 \\ 8 & 9 & 7 & 6 \\ 32 & 8 & 5 & 18 \\ 17 & 32 & 12 & 5 \end{bmatrix} = \begin{bmatrix} 9 & 14 \\ 32 & 18 \end{bmatrix}$$

Another 2 by 2 Max_Pooling Layer with a stride of 2 on an input of 2 by 2 by 2 would result in the following 1 by 1 by 2 matrix:

$$\begin{bmatrix} \begin{bmatrix} 5 & 18 \end{bmatrix} & \begin{bmatrix} 12 & 5 \end{bmatrix} \\ \begin{bmatrix} 7 & 14 \end{bmatrix} & \begin{bmatrix} 28 & 16 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} 28 & 18 \end{bmatrix} \end{bmatrix}$$

3.2.4 Local Response Normalization (Lrn) Layers

A Local Response Normalization (Lrn) Layer can be very useful as it allows one to normalize the inner most elements of a matrix, reducing the affect of variance in the photos, however since we already standardize/normalize the images this is rather redundant in this case.

The three-dimensional matrix is treated as a two-dimensional array of vectors, in this case the vectors are the various filters for that pixel. Each vector is then normalized independently of the others by dividing the vector by the weighted squared sum of the elements in the vector within the depth_radius, which is a parameter of the Lrn Layer. The Lrn Layer also has parameters for bias which is kept positive to avoid dividing by 0, an alpha which is used as a scale factor and a beta which is used as the exponent.

The Lrn Layer uses the following functions: [62]

$$Norm\_Vector = input\_vector/(bias + alpha \times sqr\_sum)^{beta}$$

Where Norm_Vector is the normalized output vector which replaces the input_vector and sqr_sum is:

$$sqr\_sum = \sum_{i=m-d}^{m+d} i^2$$

Where m is the middle of the vector and d is the depth_radius.

For both the Lrn Layers we used depth_radius = 4, bias=1.0, alpha=0.001 / 9.0 and beta=0.75.

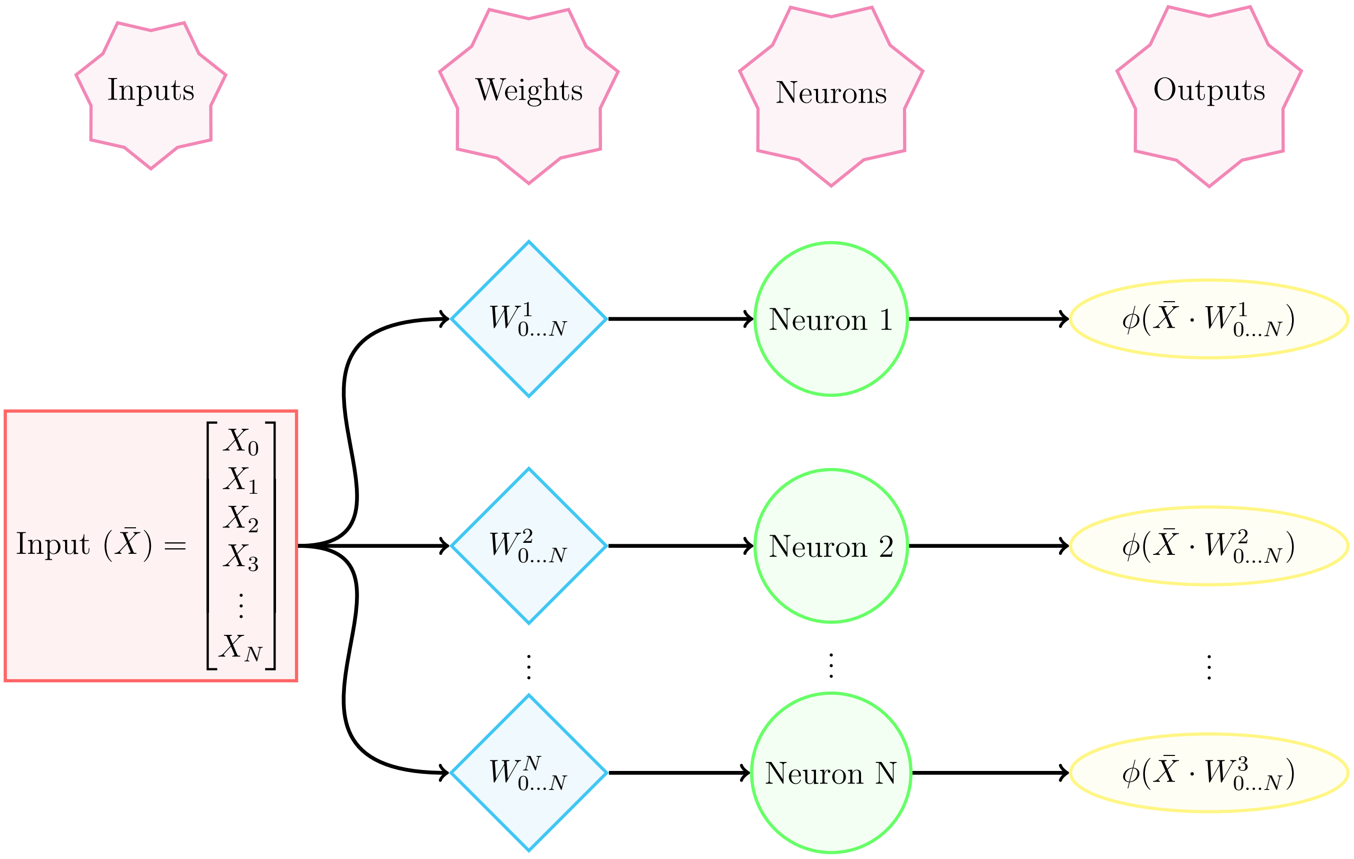
3.2.5 Dense Layers

The Dense Layers are what most people would associate with a standard Neural Network. Each neuron in the layer (in this model we used one layer with 1,024 neurons) is connected to every output from the previous layer (in this case the second Max Pooling Layer). Every output in the previous layer is multiplied by different weights before being input to the Neuron i.e.

$$\begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_N \end{bmatrix} \cdot \begin{bmatrix} W_0 \\ W_1 \\ W_2 \\ W_3 \\ \vdots \\ W_N \end{bmatrix} = Neuron\_Input$$

where X_Z is the Z output of the previous layer and W_Z is the weight associated with the neuron applied to X_Z

It is these weights that the layer learns. Further more we decided to include a bias term, Tensorflow makes this optional but it did increase the models accuracy, however it is not used in the Convolutional Layers. The bias is usually defined by adding an input (or output of the previous layer depending on the way you look at it) called X_0 which is usually defined as 1 and so the corresponding weight W_0 is the actual bias term that is learnt. Essentially the output of each neuron in a layer is $outputs = \phi(inputs \cdot weights + bias)$. Note that since the weights and the inputs are combined with the dot product and the bias is $X_0 \cdot W_0$ where X_0 is 1 then the output is $output = \phi(\begin{bmatrix} X_1 \\ \vdots \\ X_N \end{bmatrix} \cdot \begin{bmatrix} W_1 \\ \vdots \\ W_N \end{bmatrix} + W_0)$ Where ϕ is the activation function, for this model we use the relu activation function, for more details see the activation functions section 3.2.6.



Inputs
Weights
Neurons
Outputs
Input $(\bar{X}) = \begin{bmatrix} X_0 \\ X_1 \\ X_2 \\ X_3 \\ \vdots \\ X_N \end{bmatrix}$
$W^1_{0...N}$
$W^2_{0...N}$
$W^N_{0...N}$
Neuron 1
Neuron 2
Neuron N
$\phi(\bar{X} \cdot W^1_{0...N})$
$\phi(\bar{X} \cdot W^2_{0...N})$
$\phi(\bar{X} \cdot W^3_{0...N})$

Figure 3.3: High level view of a Dense Layer

Figure 3.3 shows a Hight level overview of the Dense Layer where $\phi()$ is the activation function (relu in this system), $\bar{X}$ is the input vector, $W^Z_{0...N}$ is $\begin{bmatrix} W^Z_0 \\ W^Z_1 \\ W^Z_2 \\ W^Z_3 \\ \vdots \\ W^Z_N \end{bmatrix}$, Z is the Neuron that weight vector belongs to and W_y is the weight that is applied to input y. Note X_0 is the bias term and W_0 is the weight corresponding to the Bias. Usually X_0 is set to 1 and the Bias is just the value of W_0.

3.2.5.1 Dropout

Dropout is a type of regularization, for more types of regularization look at section 3.2.7, that allows us to avoid overfitting by dropping a percentage of the outputs from the neurons belonging to the layer that dropout is applied to. For this model we chose a dropout rate of 0.4 meaning that 40% of the neurons output 0 (i.e. don't output) and the remaining 60% output their original values scaled by $\frac{1}{(1-dropoutrate)}$ (in this case $\frac{1}{0.6}$) in order to keep the expected sum the same as without dropout [63]. Dropout is only applied in training, as with all regularization, and so is disregarded in prediction and evaluation of the system.

3.2.6 Activation Function/Non Linearity Layer

Most CNN's have a Non Linearity Layer that only contains a pointwise function, most commonly the rectified linear unit (ReLU) [64] which is what we used for both the Convolutional Layers and the Dense Layer. The function essentially clamps the output of a neuron between 0 and positive infinity, i.e. $relu(x) = x^+ = max(0, x)$. It has strong biological motivations and is, as of 2018, the most popular activation function for deep neural networks [65].

3.2.7 Regularization

Regularization is a process of adding a penalty to the optimizations applied by an Optimizer, a few of which are talked about section 3.2.8, in order to fight against overfitting. Overfitting is where the system can predict, very accurately, on the training data but is poor at predicting new unseen data.

There are two very common regularization functions in Machine Learning called L1 Regularization (Lasso Regression) and L2 Regularization (Ridge Regression). For this model we tried using the L2 norm in both the normal way (as a penalty to the loss function) and tried both regulizing the weights in the Dense Layers and in the Convolutional layer and just in the Dense layer but not in the Convolutional Layer. However we noticed no difference in the reduction of overfitting, most of reduction in the overfitting came from reducing the variance in the training data as explained above. Generally speaking L1 regularization is $\lambda \times |\beta|$ where as L2 regularization is $\lambda \times \beta^2$ where β is a function and λ is a hyperparameter that controls the importance of the regularization [66]. When using L2 regularization with the loss in TensorFlow we calculate

$L' = L + \lambda \times \frac{1}{2}\|w\|_2^2$ where L' is the new loss, L is the current loss, w are the weights and $\|\|_2$ is the L2 Norm i.e. $\frac{1}{2}\|w\|_2^2 = \frac{1}{2}(w_1^2 + w_1^2 + w_1^2 + \cdots + w_n^2)$ [67], note one often see this with a summation after the λ since it is part of the loss, however as TensorFlow calculates this iteratively (for each batch) the summation happens automatically.

3.2.8 Optimizers

Optimizers are how machine learning algorithms update themselves to become more accurate and learn. There are many optimizers out there however after trying a few we still found the standard Gradient Decent algorithm was the best. The following sections go over three optimization algorithms briefly.

3.2.8.1 Gradient Decent

Gradient Decent works by calculating the gradients of the weights in the network with respect to a loss function and updates the weights in the direction given by the gradient that results in the minimization of said loss function. Throughout our work we used the Mean Absolute loss, as opposed to the Mean Squared loss which caused issues with the way the system was implemented.

Using the basic example of house prices where given a house size x we want to predict the house prices y. If we aim to predict this with a straight line of best fit ($y_{pred} = mx + c$ where y_{pred} is the predicted y from our system) then we want to optimize weights m and c. Starting with random m and c we calculate the loss for all our training examples, in our case we used Mean Absolute loss (MAL) $\frac{\sum_{i=0}^{N}(|y - y_{pred}|)}{N}$ where N was the number of examples. Note that when Mean Squared loss is used we usually multiply the loss by $\frac{1}{2}$ so that the square term cancels when calculating the derivative.

We then calculate the gradients $\frac{\delta MAL}{\delta m}$ and $\frac{\delta MAL}{\delta c}$ and then update the weight using the following formula $w = w - \frac{\delta MAL}{\delta w} \times learning\_rate$ where w is the weight, i.e. m or c, and the learning_rate is a variable of the optimizer that relates to the size of the change in the weights, for this model we used a static learning rate of 0.001 as it worked better than a variably decaying rate and better than any other value (values of the learning_rate should be kept small for Gradient Decent). [68]

3.2.8.2 ADADELTA

The ADADELTA optimizer is that dynamically adapts its hyperparameters (the learning rate etc) over time using only first order information and also has only minimal additional overhead when compared with the gradient descent optimizer [69]. Whilst this is very useful as finding the best learning rate manually relies solely upon trial and error, in practice it did not cause any improvements for this particular model. In fact, even though the extra overhead is minimal, since the optimizer did not significantly improve the model as opposed to Gradient Decent, the overhead was not worth it.

That is not to say ADADELTA was not good, however since the purpose of this project was not to find the optimal system but to create the system in the first place, which could then be optimized in future works, there was not time to experiment further with ADADELTA; however it may well turn out that ADADELTA would become superior if it had been given longer training.

3.2.8.3 Adam

The Adam Optimizer is another optimizer which is invariant to diagonal rescaling of the gradients and is useful for problems which have a large amount of data, such as this problem since the images used were larger compared to what is conventional, and/or a large number of parameters [70]. However again this optimizer did not improve the system over Gradient Decent. In fact, although [70] states that "The hyper-parameters have intuitive interpretations and typically require little tuning" we found that they were not so simple; of course again it is worth mentioning that since the point of the project was to build the system and not optimize it (especially since this part of the project does not contribute as much to the field, since there are many hand trackers out there, and only acts to allow the next part of the project: Finger Tip Tracking) we did not try a large number of different hyperparameters, since Adam has four hyperparameters the number of different combinations makes the manual finding of these parameters quite difficult. Furthermore whenever Adam was used the system predicted the exact same location for every single hand and so was unusable for this system, however this again could be to do with the choice of hyperparameters.

3.2.9 Losses

There are two Loss functions that are already made to be used with Tensorflow: Mean Squared Error and Absolute Difference. Both of these methods are exactly the same as their Evaluation Metric counter parts which we talk about in the next section so we won't go over what they do yet, if you want to read ahead they are covered in section 3.2.10, 3.2.10.2 and 3.2.10.3 respectively. For this model we used Absolute Difference, whilst my particular preference is with Mean Squared Error given that it penalizes larger errors, this loss metric gave better results with the Gradient Decent optimization, most likely due to the very large errors we were getting at the beginning.

These loss functions do differer from their evaluation counterparts by allowing the use of a weight parameter to scale the loss, either for the entire batch or specifically for each element in the batch. Using this it may have been possible to improve the system to allow the Mean Squared Error Loss.

3.2.10 Evaluation Metrics

When evaluating a CNN there are multiple types of metrics that one can use. They each have their own particularities in the information they give but essentially amount to the same thing. As such TensorFlow calculates them all in a very similar manner but with differences in the underlying function used. Additionally since we use regression there is no need to apply additional functions, these metrics are all we need.

All the metrics have two local variables, the total and the count that are used to compute the various error metrics. The average can be weighted by weights, however for this system we did not use any. The system then applies the specific metric function to the total divided by the count.

3.2.10.1 Root Mean Squared error (RMSE)

The standard Root Mean Squared error function is:

$$\sqrt{\frac{\sum_{i=0}^{N}(expected_i - actual_i)^2}{N}}$$

Where N is the number of elements being evaluated $expected_i$ is the value of the label for that element (the actual answer/ground truth) and $actual_i$ is the value we got.

However TensorFlow accumulates the total by adding the squared difference for every element in the batch. This effectively computes $(\sum_{i=0}^{N}(expected_i - actual_i)^2)$ but does so in batches so that the evaluation metric can be found for each mini batch. It then divides by the count (N) and applies the square root essentially performing the standard function. All Tensorflow metrics described in the rest of this report work in the same way so we won't go through it again but each metric calculates its specific function in batches by accumulating total and dividing by count to get the mean.

The Root Mean Squared error is a perfect mix of the Mean Squared error and the Mean Absolute Error. We will go over both of the other metrics' pros and cons in their own sections however the Root Mean Squared Error has the advantage by being in easy to visualize units, like Mean Absolute error, but also penalizes larger errors, such as Mean Squared Error.

3.2.10.2 Mean Squared error (MSE)

The standard Mean Squared error function is:

$$\frac{\sum_{i=0}^{N}(expected_i - actual_i)^2}{N}$$

Where N is the number of elements being evaluated $expected_i$ is the value of the label for that element (the actual answer/ground truth) and $actual_i$ is the value we got.

The Mean Squared error penalizes larger differences between the expected and the actual, since the difference is squared. This is very useful for getting a better idea of how well the system is working as we will show in the evaluation of these metrics. Unfortunately by squaring the differences the results are far larger error numbers, which aren't easy to visualize.

3.2.10.3 Mean Absolute error (MAE)

The standard Mean Absolute error function is:

$$\frac{\sum_{i=0}^{N}(|expected_i - actual_i|)}{N}$$

Where N is the number of elements being evaluated $expected_i$ is the value of the label for that element (the actual answer/ground truth), $actual_i$ is the value we got and $|x|$ is the absolute function (i.e. for $x < 0 \quad x = -1 \times x$ otherwise $x = x$).

The Mean Absolute Error uses easy to visualize units, it is literally the distance between the expected and actual value. However it doesn't penalize larger differences and as such a system can seem more accurate than it is since the error may not seem so large, especially if it is mostly accurate with only a few outliers.

3.2.10.4 Evaluation Metrics Comparison

The various evaluation metrics all have particular pros and cons, for example let us look at a system that has a systematic or constant absolute error of 6 except for one which is out by 11. This error of 11 will effect the different evaluation metrics differently, since it essentially has different weightings in the various metrics.

Mean Absolute Error: $errors = [6, 6, 6, 6, 11] \quad output = 7$

Mean Squared Error: $errors = [36, 36, 36, 36, 121] \quad output = 53$

Root Mean Squared Error: $errors = [36, 36, 36, 36, 121] \quad output = 7.28 \quad (to \quad 3sf)$

As one can see the Mean Absolute Error has an error of 7, so close to the systematic error, which could easily lead the user to believe that there is only a small discrepancy when in fact there is a much larger error hiding. However this is far easier to visualize, on average we know that there is a distance of 7 between our system and the true answer.

Mean Squared Error on the other hand has a higher weighting on the larger error even though it is only one element. If we were to square the Mean Absolute error we get 49, which is smaller than the Mean Squared error, and so by penalizing the larger errors we can see more clearly that our system has some larger errors; it is very useful to look at both the Mean Absolute Error and the Mean Squared Error and in fact comparing the Mean Absolute error squared (MAE^2) to the Mean Squared Error tell one a lot about the number of larger errors lurking in a system.

The Root Mean Squared Error, as mentioned before, is a great match between the previous two metrics. It still penalizes the larger errors in the system but is also easy to visualize since it is in the same units as the Mean Absolute Error and good for comparison with the MAE, in fact it is very similar to comparing the MAE^2 with the MSE.

3.3 System Output

The system's last layer is another Dense Layer, without dropout or bias, and has eight neurons, one for each predicted value. The output is two vectors, one for each hand, containing four elements: The x coordinate of the top left of the box surrounding the hand, the y coordinate of the top left of the box surrounding the hand, the width of the box and the height of the box; in the case of no hands or only one, the missing hand is given a value of [0, 0, 0, 0]. Using these four elements it is simple to display a box over the image using almost any graphics library, we used OpenCV to draw the boxes. We contemplated using other outputs to define the boxes such as a 4 by 2 matrix per hand for the coordinates of each corner of the box, however that would cause there to be sixteen output neurons instead of eight increasing the size of the output space. Furthermore by returning these four elements it makes transformations much simpler; for example, as mentioned before the input images were randomly flipped/mirrored in the x and y axis, with a matrix of four corners, eight elements, each corner had to be mirrored separately, TensorFlow is quite particular when it comes to multiplying Tensors (TensorFlow's main object class), whereas by only having x and y coordinates all that was necessary was to flip one corner, and if the flip was vertical, i.e. in the x-axis, then the height and the width were switched too, which is trivial.

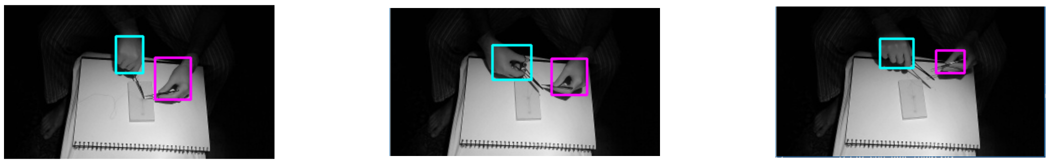


Figure 3.4: output Images For The Hand Detection CNN

There were some difficulties to do with the left and right hand in the training set, if only one hand was present we always set it as the left hand. This made manually validating the system faster, as we didn't have to check which hand was which, but the CNN may have learnt this as a feature, which it wasn't, potentially causing issues. Furthermore since the predictions are displayed on top of a video of inputs when being displayed, there is some jagged movement caused by the error in the prediction in the previous frame and the movement of the hand between the previous frame and the current frame. We managed to stabilize this by averaging the box's location and size over a set number of frames, although currently this parameter is set by hand and is not automatically determined.

4 Conventional Vision Approach

In this chapter we talk about the Conventional Vision approach to Hand Tracking and more generally Object Detection. We will first cover a selection of the methods available and then talk about the limitations involved with using a solely Vision approach.

4.1 Methods

The following section covers a set of common vision approaches to object tracking. Whilst we stuck to the most simple approach for generation of the dataset, there are many great detection methods out there.

4.1.1 Colour Spaces

Using Colour Spaces is the simplest approach to object detection and, with the right amount of tinkering, can be very accurate. We first convert the image into the Hue Saturation Value (HSV) Colour Space from the standard Red Green Blue (RGB), note that OpenCV (which was used for this task) stores images in Blue Green Red (BGR) Colour Space. By having the image in HSV Colour Space we can then define a threshold, a lower and upper threshold, of which we will detect. For example in this project we used [110, 255, 255] as the upper threshold and [90, 85, 85] as the lower threshold. A hue of 110 and 90 are both cyan, which is the same colour as the gloves used in the training examples. In general, when defining an upper and lower threshold, we take ± 10 of the hue we want to track and set the saturation and value for the lower bound to 100 and 255 for the upper bound. However for these gloves a lower bound of 100 was too large, so next we tried 75 but that caused shadows to be tracked too, since shadows have a bluish tinge to them. Finally we settled on 85, which had a high rate of accuracy, even though it misplaced the occasional hand, though this was caught during the manual verification stage. Although this tinkering of values takes time, it took approximately an hour to find the best thresholds for the skin colour matcher which resulted in [20, 255, 255] for the upper threshold and [0, 75, 75] for the lower. This is much the same as finding hyper parameters for a Neural Network (without having to wait long times for training to run. However once the optimal values were found we ended up with a rather decent system that we then used to generate the Hand Tracking CNN's dataset.

4.1.2 Three Dimensional Camera's

We tried using a Three Dimensional Camera, the addition of depth information is very powerful. For example we used the standard Colour Space technique but then, given a set of possible locations for an object, we considered the depth information. We used basic heuristics about the hand such as the fact that the back of a hand has a slight curve to it. By using this heuristic we can actively reject all false positives caused by blueish shadows, since shadows are flat against the object of which they are cast upon.

There was also the possible plan to build a CNN to learn depth information from a two dimensional image and then use this to detect hands, however we would still require a hand detection system so we found no point in adding an extra step to our goal.

4.1.3 Feature Detection

Feature detection has many different methods and is a very good way of performing matching across images. Here we discuss a few of the possible methods that exist. We dappled with all of these methods; however since they all still needed human verification to ensure they were accurate enough to act as training inputs for the hand tracking system, it seemed counter intuitive to spend time implementing them when we can get similar results with Colour Spaces which requires the same amount of time in terms of human interaction.

4.1.3.1 Harris Corner Detection

Harris Corner detection is a famous method for detection of corners. Corners are great features due to variance within them, think of a jigsaw puzzle, the middle/uniform pieces are difficult to figure out where they belong, the edge pieces are more obvious (it is clear what group they belong to but not necessarily where along an edge they lie, corners on the other hand are very obvious and therefore make a good feature for matching with. Furthermore Harris is rotation-invariant, i.e. even if the image is rotated we can find the same corners since corners still remain corners in rotated images.

Harris works by finding the areas of maximum intensity change given a change in location from an initial window. The formula is: $E(u,v) = \sum_{x,y} \underbrace{w(x,y)}_{\text{window}} [\underbrace{I(x+u,y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}]^2$. x and y are pixel coordinates and the window is a section of the image, for example the window could be a 3x3 grid entered around x and y.

The aim of Harris corner detection is to then maximize this function $E(u,v)$. Therefore one needs to maximize the Intensity terms $([\underbrace{I(x+u,y+v)}_{\text{shifted intensity}} - \underbrace{I(x,y)}_{\text{intensity}}]^2)$. By applying Taylor Expansion $(\sum_{n=0}^{\infty} \frac{f^{(n)}(a)}{n!}(x-a)^n = f(a) + \frac{f'(a)}{1!}(x-a) + \frac{f''(a)}{2!}(x-a)^2 + \cdots + \frac{f^{\infty}(a)}{\infty!}(x-a)^{\infty}$ [71], we can transform the equation to $E(u,v) \approx \begin{bmatrix} u & v \end{bmatrix} M \begin{bmatrix} u \\ v \end{bmatrix}$ where $M = \sum_{x,y} w(x,y) \begin{bmatrix} I_x I_x & I_x I_y \\ I_x I_y & I_y I_y \end{bmatrix}$.

I_x and I_y are image derivatives/gradients in x and y directions respectively, image gradients are easily computable with kernel filters such as the Sobel operator $\underbrace{\begin{bmatrix} +1 & 0 & -1 \\ +2 & 0 & -2 \\ +1 & 0 & -1 \end{bmatrix}}_{\text{Horizontal derivative approximation}}$

and $\underbrace{\begin{bmatrix} +1 & +2 & +1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}}_{\text{Vertical derivative approximation}}$ [72].

Finally we calculate $R = det(M) - k(trace(M))^2$ where $det(M) = \lambda_1\lambda_2$, $trace(M) = \lambda_1 + \lambda_2$ and λ_1 and λ_2 are the eigen values of M.

When the absolute value of R ($|R|$) is small, which happens when $\lambda_1 and \lambda_2$ are small, the region is considered flat.

When R ¡ 0, which happens when λ_1 is much greater than (>>) λ_2 or vice versa, the region is edge and when R is large, which happens when both $\lambda_1 and \lambda_2$ are large and λ_1 and λ_2 are similar ($\sim$), the region is a corner [73]. Have a look at figure 4.1 for a graphical representation of this.

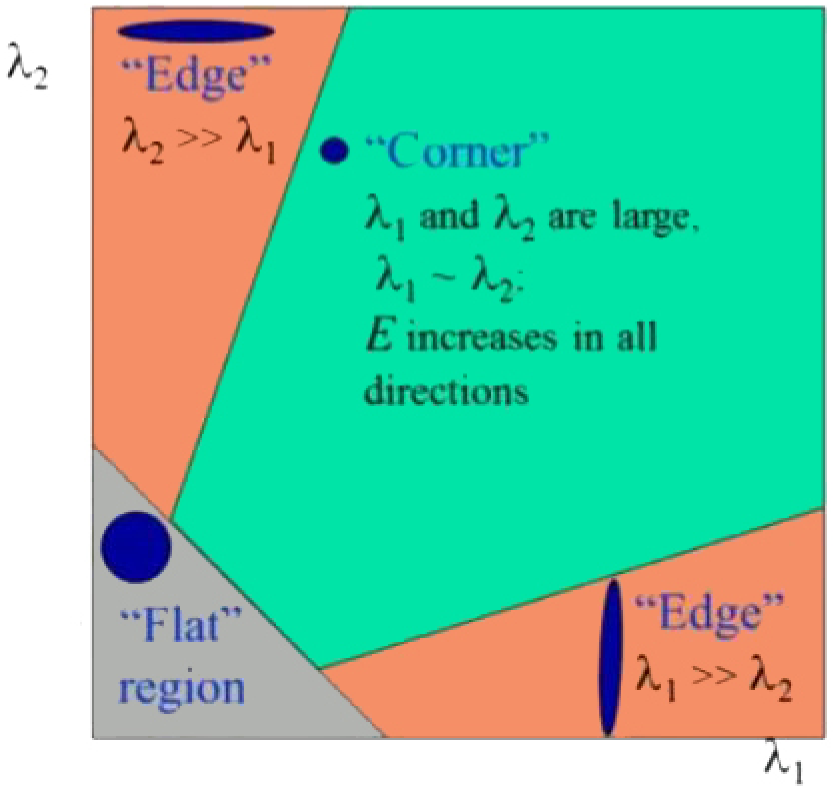


Figure 4.1: Harris Corner Detection [73]

4.1.3.2 SIFT (Scale-Invariant Feature Transform)

As we just explained, The Harris Corner Detector is invariant to rotation; however it is not invariant to scaling. A corner may no longer be considered a corner if the image is scaled, as can be seen in figure 4.2. A corner in a small image and viewed in a small window will be shown

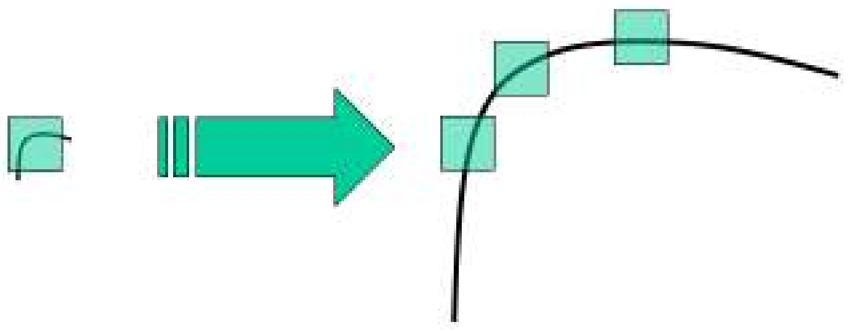


Figure 4.2: effect of scaling on corners [74]

as flat if one continuously zooms in within that window. Therefore Harris Corner Detection is not scale invariant.

Since it is not possible to use the same window to detect all corners in an image we use scale-space filtering. By finding the Laplacian of Gaussian (LoG), which acts as a blob detector, with various σ (scaling) values, we can detects blobs of various sizes, caused by the changes in σ. A Gaussian kernel with a small σ gives better results for small corners whereas a large σ works better for large corners. Using this we find the local maxima across multiple scales and locations giving a list of (x,y,σ) values corresponding to a corner at location x, y at a scale of σ [74]. Unfortunately LoG is an expensive operation and so SIFT instead uses Difference of Gaussians (DoG), which is an approximation of LoG, found by taking the difference between two Gaussian blurrings with different σ values applied to the same image. This is applied to a series of levels in an image pyramid. An image pyramid is a series of the same image at different resolutions. To find the local maxima we compare one pixel in the image with its 8 neighbours as well as 9 pixels in next image in the pyramid and 9 pixels in previous pyramid. If it is a local maxima, it is a potential corner and we can say that the corner is best represented at that σ scale. [74]

A common blob detector used, that is based on the LoG is given by $g(x, y, \sigma) = \frac{1}{2\pi\sigma} \exp^{\frac{x^2+y^2}{2\sigma}}$ [75]

4.1.3.3 SURF (Speeded-Up Robust Features)

SURF was designed to be a faster version of SIFT. SURF, instead of approximating LoG with DoG, instead approximates LoG with Box Filter, as shown in figure 4.3. One big advantage of this approximation is that, convolution with a box filter can be easily calculated with the help of integral images. And it can be done in parallel for different scales [76].

4.1.3.4 FAST Corner Detection

Whilst the previous corner detection methods are powerful they aren't quick enough for a real-time application. FAST (Features from Accelerated Segment Test) is a solution to this, a fast method for calculating corner locations in an image.

The FAST algorithm is computed using the following steps: First choose a pixel p in the image

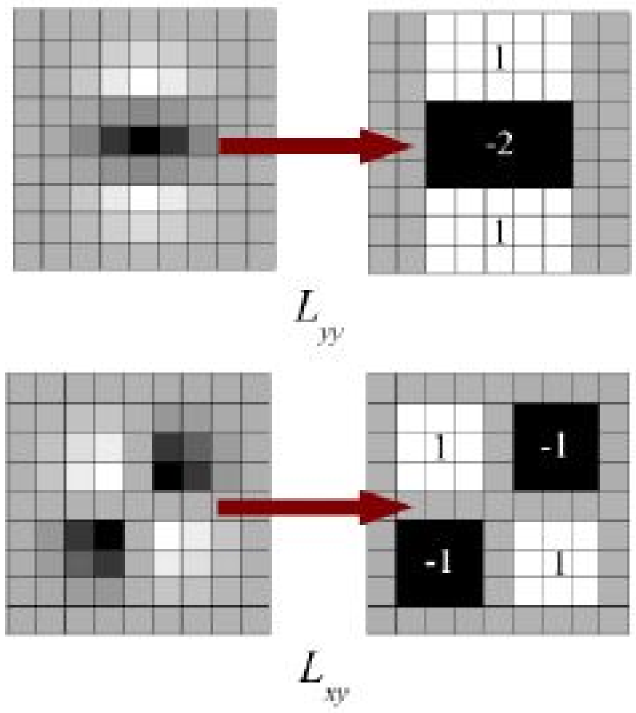


Figure 4.3: SURF Box Filter Approximation [76]

which is to be identified as a corner or not and calculate its intensity I_p and choose a threshold value t.

Second we analyse a set of 16 pixels surrounding p; p is considered a corner if there exists a set of n contiguous pixels in the circle of 16 pixels which are either all brighter than p by the threshold value, i.e. $I_p + t$, or are all darker than p by the threshold value, i.e I_pt. We then repeat this for every single pixel in the image.

A high-speed test was proposed to exclude a large number of non-corners. This test examines only the four pixels north, south, east and west of p; first north and south are tested to see if they are either brighter or darker than p, if so, east and west are also checked. If p is a corner at least three of these points must either all brighter than p by the threshold value, i.e. $I_p + t$, or all darker than p by the threshold value, i.e. $I_p - t$. If neither of these is the case, then p cannot be a corner. The full segment test criterion can then be applied to the passed candidates by examining all pixels in the circle. This detector in itself exhibits high performance, but there are several weaknesses such as:

- It does not reject as many candidate pixels as opposed to the original algorithm when $n < 12$.

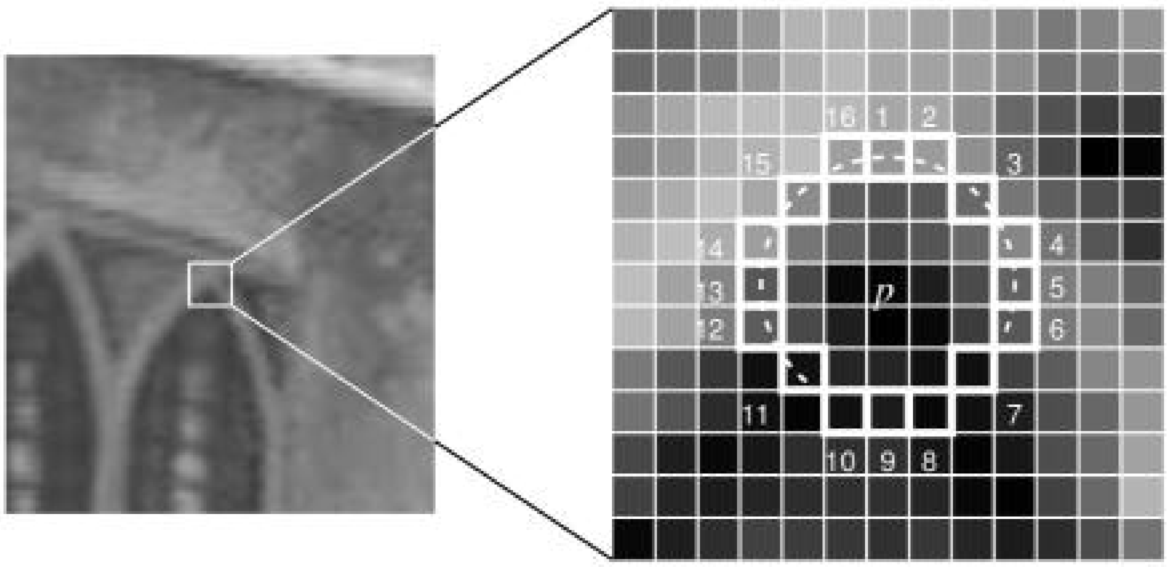


Figure 4.4: Fast Algorithm Analysing a set of 16 Pixels [77]

- The choice of pixels is not optimal because the efficiency depends on the distribution of the corners.
- The results of high-speed tests are not kept for later computations.

However these weaknesses can be addressed by applying machine learning to fast, using the following steps:

Firstly we choose a set of images for training (preferably from the target application domain) and run the FAST algorithm on every image in order to find the feature points.

Then for every feature point in every image, we store the 16 pixels surround it as a vector. The combination of all the vectors forms the feature vector P. For each pixel x in the vector of the 16 pixels can have one of the following three states: Brighter, the pixel is brighter than the target pixel by the threshold value, i.e. $I_p + t \leq I_x$, darker, the pixel is darker than the target pixel by the threshold value, i.e. $I_p - t \geq I_x$ or similar, the pixel is neither brighter or darker than the target pixel by the threshold value, i.e. $I_p - t < I_x < I_p + t$

Next, depending on these states, the feature vector P is divided into 3 subsets, P_d, P_s and P_b, and we define a new boolean variable, K_x, which is true if x is a corner and false otherwise.

Finally we use the ID3 algorithm, a decision tree classifier, to query each of these subsets where the variable K_p is used for the knowledge about the true class. We then apply this recursively to all the subsets until the entropy is zero.

By doing this we create a decision tree that can be used for fast detection in other similar images. [77]

4.2 Limitations

Unfortunately there are too many limitations to using a purely vision oriented approach, which we will discuss in this section.

4.2.0.1 Lighting

As mentioned before shadows can cause issues with the Conventional Vision Approaches, whilst outdoor lighting will not be a factor in practice (since operating rooms do not tend to have windows to the outside, self shadowing is a serious issue, caused, for example, by hands crossing over each other. Furthermore the colour of the light can be an issue, for example bright, cold coloured light, such as that in the 6500 K range, which are standard, in halogen bulbs common in operating rooms, have a blueish tinge. This is especially detrimental for use with blue or cyan coloured gloves, a common colour of surgical gloves, and would add a requirement to the type and colour of gloves that can be used with the system, which would reduce the usefulness of the system.

4.2.0.2 Image Type

These systems almost all require colour images, or are at least seriously improved by using colour information, this puts an extra, albeit standard, constraint on the image that can be used, we can easily make an image grayscale but cannot make a grayscale image coloured. Furthermore, by using colour, we add a more serious constraint, when we want to use the system in many different settings, same task (surgical skill assessment) but different locations (different hospitals and training rooms), we are restricted. The colour of the gloves may become essential, the background colours, the table colour or the model used to practice suturing, may be required to be the same or, even worse, the suturing tools may be required to be the same, some suturing thread is blue which may cause errors in the system.

Moreover the size of the images may be important, Neural Networks analyse each and every pixel, whereas Vision systems, in general, treat the image as a whole; although a Vision system may consider an individual pixel, the number of pixels is still important and individual pixel analysis is only part of what makes a vision system. For example FAST uses the 16 surrounding pixels as part of its feature detection, meaning there must be at least 49 pixels in the image, since pixels are in a square grid we have to look at pixels at a step of 3 pixels away, resulting in a surrounding diameter of 7 pixels (refer to figure 4.4 for a graphical representation of this). However a Neural Network can analyse very small images, it only depends on what it was trained with and once the actual model is designed, one can easily play around with the input size to see what works. These images are then made smaller by Max Pooling Layers.

4.2.0.3 Generalization/Robustness

Camera calibration is a serious issue and pure vision systems do not generalize well to changing environments. Although this can be mitigated somewhat by building a system to calibrate a camera given the parameters of the task, i.e. glove colour, lighting information and distance from the camera, it seemed unnecessary to spend time building a calibration system when we could just build a system that is environmentally invariant.

5 Surgical Skill Assessment Metrics

Although the majority of the contribution to Automated Skill Assessment comes from the Hand Tracking and Tip Classification systems, this project is still aimed at improving the way in which trainee surgeons are assessed and as such we created a sample of demo metrics that can be used to evaluate performance.

These are a mixture of metrics derived from the OSATS, and so are either easily generalizable to generic surgical procedures or are already general enough, or are suturing specific. These are only really meant as examples of how one can use the systems developed to asses surgical skill and by no means is a complete list of possible metrics, given more time and study into the ways in which surgical skill can be evaluated, more complicated and powerful evaluation metrics could be derived.

5.1 Time Taken

This is a very simple and common metric; the time taken is standard way of assessing the speed and by extension the skill level of a surgeon. The system has the advantage of being able to accurately time the procedure by starting the timer when both hands are seen, since the procedure requires both hands, and stopping when no hands are seen again, the timer records the time when no hands are seen and if no hands reaper again this time is considered the end time.

5.2 Smoothness

We measure the smoothness of the movements of the surgeon by calculating the standard deviation of the location of each hand throughout the procedure. We had to apply the averaging box technique, mentioned in section 3.3, to reduce the jagged movement of the hand tracking boxes.

Alternatively for comparing two surgeon it is irrelevant whether or not the boxes themselves are calculated smoothly since we can pay no attention to the actual standard deviation value and just compare results. We could therefore, with the help of an expert surgeon, develop benchmarks for the standard deviation for the procedure and then just compare the trainee to said benchmarks.

5.3 Knowledge Of Procedure

For this procedure we can consider the distance between the two hands of the trainee. When first starting out with suturing it is easy to make larger movements, for example when inserting the needle, pulling the thread through or wrapping the thread around the needle holder, which causes the hands to drift apart. Of course if the hands are too close together this also shows a lack of knowledge. Fortunately hands can only get so close to one another where as hands can get almost infinitely far apart (bounded by the length of the arms). As such we defined the optimal hand distance apart manually by experimentation and then take the absolute distance between the optimal and the actual. Of course with depth information this could be further enhanced since we can now calculate distance on the Z-axis as well as the X and Y-axis.

5.4 Rate of Suturing

We can define the Rate of Suturing as the speed at which the hand are on the same half of the image. For example, with the suturing model set in the middle of the image, the needle holder hand inserts the needle on one side of the model, and by extension the image, and exit on the other side of the image/model. By calculating the number of times one hand crosses them middle and dividing it by the time taken we can calculate this Rate of Suturing metric.

6 Convolution Neural Network: Joint Tip Classification

Tip Classification is the main contribution of this project. Although the Hand Tracking System is an important step to be run before this system, it isn't really new. As the previous chapters have shown, although other techniques may be better or worse, there is nothing new with hand, or more generally object, detection. Finger Tip Classification on the other hand is newer. Being able to detect the tip of the thumb and the index finger is very useful since we know where they should be placed in relation with the tools used. We effectively end up with more precise data to use for the evaluation metrics previously mentioned, having smaller areas to track provides a smaller window of error, if we measure with an accuracy of $\pm 10\%$ and we locate features, finger tips, with a 10x10 grid, then we're only out by 1 pixel as opposed to being up to 10% out on a 320x180 which is around 3 by 2 pixels.

Further more this system is a classification problem, which has the benefit of returning the probability of each patch in the input image belonging to each category, we will explain what these patches are in the next section 6.1. Since some patches partially overlap we can combine the probabilities for every pixel to calculate the overall probability that any given pixel belongs to witch category. Furthermore we can use a multicoloured mask, as opposed to binary white and black or shades of gray for various probabilities, to show how much each region belongs to each category. For example we use red to show no finger tip, green for a thumb and blue for an index finger tip. Furthermore the colours combine uniquely and obviously, for example a magenta area means that it is undecided whether or not there is a index finger tip or not, the bluer it the more likely it is a finger tip and the more red, the less likely. We then combine this with heuristics to determine what is more likely, for example we know a thumb has to be next to an index finger and that each hand only has one thumb and one index finger so we can remove impossible probabilities. We can continue to add more heuristics to help further increase the accuracy of the system meaning it is not so reliant on training.

please note that at the time of writing the hand tracking system is not finished, we have built the dataset generator and the initial model, however these are subject to change given the trial and error style of machine learning. We will give details in this section covering the current state of the system, which should be finished by in time for this project's presentation on the 26th of June 2018.

6.1 Inputs

6.1.1 Dataset Generation

Generating the dataset is quite difficult, the system went through many iterations and is very easy to accidental destroy if a mistake is made. Unlike the dataset for the Hand Tracking Task, which was simple to verify and rectify, since the whole image was saved and the labels can be manually adjusted. However for this task there were many issues with rectifying issues. In both the original system, where we just took ten pixels from every image, and the second iteration where we used the patches, we saved the patches in one directory and the labels, 0 for nothing 1 for thumb tip and 2 for index finger tip, in a text file.

However there were two main issues: If there was a mistake, that wasn't recognised straight away, all we hand to look at to decide if we had made a mistake was a 10x10 pixel section, or in the second iteration at most we had a 20x20 pixel section. This is generally not enough to decide if a tip is present and even though it was simpler for a thumb, differentiating between an index finger tip and an ordinary finger tip is not obvious. Moreover occasionally the thumb and index finger are extremely close together, especially with the forceps, and so we have to save the patch as both a thumb and finger tip. Originally we just saved the patch twice and then added both labels, but that resulted in a non deterministic number of patches per image. We contemplated combating this by adding blank patches to the set to make all sets of equal size but this would mean that the final model would have an arbitrary number of patches.

We finally settled on a system where if both a thumb tip and an index finger tip are present in the patch we set it to be both at once. TensorFlow uses a special type of Tensor called a one-hot Tensor that stores classifications as a vector of the length of the number of possible classification where every element is zero except for the index of the true class i.e.

$$Nothing = [1,0,0] Thumb\ Tip = [0,1,0] Index\ Finger\ Tip = [0,0,1]$$

We can therefore then define the images that contain both as $Both = [0, 1, 1]$, we considered using 0.5 for both implying that it was a fifty fifty whether or not it was a thumb or finger but since a probability is produced anyway there was no point, more so when considering the fact that sometimes it truly was spot on for both categories.

To generate these patches we overlayed the shape of the patch on top of the actual image, since for human recognition context is vital, and manually decided which category the patches belonged to. Whilst initially we saved the individual patches we now use TensorFlow's image patch function, since it is specifically aimed for use with a CNN and then used the original image for training, reducing the number of images needed saving.

We get the initial image by using the Hand Detection CNN to extract the hands. Then we resize the images to be 40x40 pixels by padding the original image until it is easily scaled up to 40x40 pixels, this way the image aspect ratio is preserved. We then generate the following patches: Four regular patches of 10x10 with a step of 10, 20x20 with a step of 10, 10x20 with a step of 10, 20x10 with a step of 10 and four diagonal patches. The diagonal patches had two types, small diagonal and large diagonal, the small diagonals are two 10x10 boxes overlapping the bottom right 4x4 grid of the top box with the top left 4x4 of the bottom box for one box and the top right 4x4 grid of the bottom box overlapping the bottom left 4x4 grid of the top box. These boxes had a step of 8 pixels. The two large diagonals were the same but with a third box added again overlapping with a 4x4 grid section but with a step six of 6 pixels. This results in 113 patches per image for training, the only downside of which is that there were 14,330 hand images extracted from the dataset for the Hand Tracking system. This meant that a very large number of patches need analysing which is extremely slow.

6.1.2 Data Format

The patches that get input to the system are in a format more akin to the filters returned by the Convolutional Layers of the CNN. Since we want the image "dimensions" of the image to be the same TensorFlow automatically adds empty patches. For each image we end up with a patch sequence of dimensions $4\ by\ 4\ by\ Patch_x * Patch_y * Number\_Of\_Channels$, since input images are again in grayscale, the advantages of which are explained in the Hand Tracking section 3. The 4x4 is considered to be the width and height of the image and the $Patch_x * Patch_y * Number\_Of\_Channels$ is essentially the number of filters. However since not all of the 8 patch types have the same number of elements we have to reshape/flatten the inputs so that the last dimension, the filter like dimension, is every single pixel from every single patch. Fortunately TensorFlow reshapes its dimensions in order, that is to say if we do patch one first (10x10) and then patch two (20x20) then the first 100 elements are from the first (10x10) patch, the next 400 are from the second patch (20x20) etc. This results in $\underbrace{10^2}_{\text{patch 1 size}} + \underbrace{20^2}_{\text{patch 2 size}} + \underbrace{10 * 20}_{\text{patch 3 size}} + \underbrace{20 * 10}_{\text{patch 4 size}} + \underbrace{16 * 16}_{\text{patch 5 size}} + \underbrace{22 * 22}_{\text{patch 6 size}} + \underbrace{16 * 16}_{\text{patch 7 size}} + \underbrace{22 * 22}_{\text{patch 8 size}} = 2380$ elements, which when one considers the amount of data that contains is quite small. Therefore the total number of input is $\underbrace{4}_{\text{patches in x direction}} \times \underbrace{4}_{\text{patches in y direction}} \times \underbrace{2380}_{\text{number of pixels}} = 38080$ which is again small; if we look at the input space of the Hand Detection system we have $320 \times 180 = 57600$ so this system can defiantly be run with Tensorflow without making it smaller. However we cant use a Max Pooling layer in the traditional way since the patches technically are of image size 4x4 which is far too small to be pooled down. One possible solution that we are trying is to split the data into two halves, treat these halves as two different depth levels and use a three-dimensional Convolutional Neural Network. However this is untested and needs further evaluation.

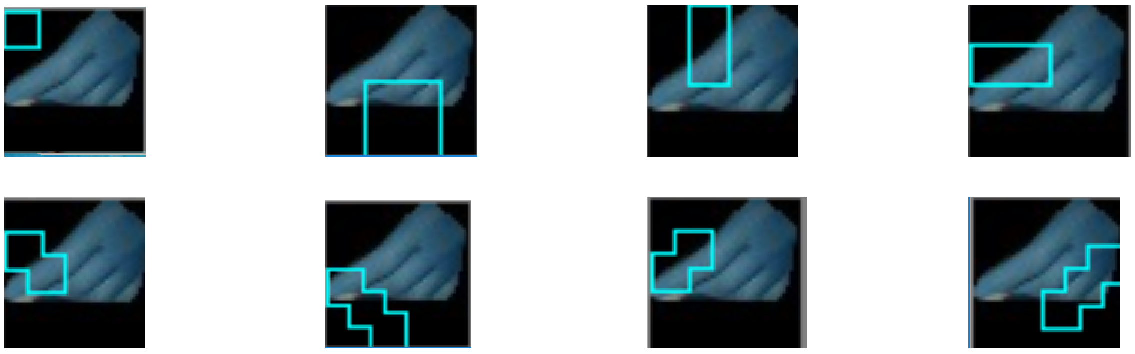


Figure 6.1: Input Images With Patches overlayed For The Tip Classification CNN

6.2 Model

Whilst no model currently exists we do have a full model yet, however we are currently testing with the same model initial model as with Hand Detection, with the differences in the output layer, which we will describe below.

6.3 Outputs

The final layer will return an two dimensional index and a three-dimensional vector containing the probability of each category. The index specifies which patch number and which patch by that number got that result. For example a return of $(0, 5), [10, 20, 70]$ means that for the sixth patch obtained by the 10x10 patching has 10% likelihood of being neither a thumb or index finger tip, 20% of being a thumb tip and 70% of being an index finger tip. As mentioned before we can now use this data we can generate reconstruct the original image and overlay a colour map to visually display the likelihood of each pixel being in each category.

7 Evaluation

7.1 What Have We Achieved Compared To The State Of The Art

7.1.1 Contributions To Surgical Skill Assessment

Currently, in the world of automated surgical skill assessment there is not much in the way of an autonomous system. As mentioned in the background chapter 2 there are various sensor systems that track the trainee under test and evaluate their performance, however the data recorded often needs to be reviewed by a professional surgeon. Whilst we admit that our system is not completely ready to act as an autonomous skill assessment system we do believe that we have shown that this is a possibility with the same core system, but with improvements made. Furthermore the majority of the current state of the art systems rely on sensors, which are often bulky and not natural for a surgeon, or on specialised equipment; our system on the other hand requires neither. Moreover its retroactive, as long as a surgery was recorded all that is left to do is feed it into the system and the requirements on the camera for recording is seriously relaxed, as long as its in 16:9 aspect ratio there is no problem what so ever, and even then only padding or cropping of the image would be necessary. We have created a system that put little to no constraints on the trainees and assessors involved in the process.

7.1.1.1 Our Surgical Evaluation Metrics

The evaluation metrics we have proposed work quite well. It must be noted however that since I was the only test subject on which the metrics were applied, I performed suturing and knot tying 6 times, firstly as poorly as possibly (the worst scoring), secondly where I made a lot of sudden movements (the next worst), thirdly as calmly as possible (middle score), fourthly as best I could but taking my time (second best) and lastly as fast and accurately as I could (which took many attempts). Using these as the golden classification labels our metrics performed quite well, they categorised the 6 attempts as expected. However these metrics have never been tested on actual professionals and it could well be that given the many different styles of surgeons some metrics will incorrectly evaluate a professional, hence why we suggested using the scores as benchmarks and forgetting the units of the metrics, i.e. just comparing scores, individual scores have no semantic meaning.

7.1.2 Contributions To Vision and Machine Learning

Whilst Currently there are many Vision techniques for Hand Tracking, by implementing a system that can do this in grayscale we have added something that is not simple to do purely with Vision. Furthermore, although feature detection is popular and powerful, it is usually slow, FAST uses machine learning to speed it up so by using a purely CNN system we cut out the middle man.

As for Machine Learning there are many Hand Detection/Tracking systems, however a few of the famous require the use of three-dimensional cameras, such as Intels Realsense CNN Hand Tracking system. We have added this ability with only a standard camera. Moreover we only use our initial hand detection system to lower the size of the problem for our Tip Classification problem. Whilst pose estimation systems are indeed quite popular for these types of problems, we have built a system specifically for finger tips which also has the ability to show probabilities which can then be useful to another system. The core idea behind all these systems is to create a pipeline, here we can either feed the output of one system into the other or stop at any point and use the current output. This is exactly how we uses the system anyway, The Hand Tracking CNN acts as a black box, decreasing the search space for the Tip Classification system. We could for example, after using the Hand Tracking system, use a pure Vision system to classify the finger tips, perhaps with fast or another feature matcher.

7.2 How Could We Improve

In spite all these contributions to the field, there are numerous ways in which we can improve the system.

7.2.1 CNN Hand Detection

Firstly, by using a more complex decaying weight function for the learning rate we could probably use the Mean Squared Error loss for the optimization of the system. This would result in a faster learning model since the larger errors would be penalised more heavily which in this instance is really useful. Furthermore we could then have more choice of optimizer and, given more time, pick better values for the hyperparameters.

We also would really like to change the Hand Detection problem to a classification task since having probabilities is extremely useful when one has strong domain knowledge, i.e. there can only be a maximum of two hands so the two largest probabilities are most likely correct.

As for the dataset generation, whilst we currently find the contours of the detected object and then the regular bounding rectangle, we toyed with both the tightest bounding rectangle or the convex defect-free bounding rectangle. However both resulted in a larger output space. Whilst the tightest bounding rectangle only required one extra parameter, the angle of the box, it did not increase the effectiveness of the whole system by very much, since we only use the hand tracking to decrease the input space for the Tip Classification system. The convex defect-free bounding rectangle had many more points, which were variable in number and so not suited for output of a Neural Network. However given more time it may be possible to use one or both of these methods to get a more accurate output.

We also had some issues with training detection of the left from the right hand. Whilst we did not care which hand was which, the CNN may have ended up treating this as a feature. Unfortunately our dataset generation system was not careful to specify which hand is detected if only one hand is seen, it is always the left hand. Whilst this is easy to fix when displaying the CNN's predictions (whichever hand is closer to the left of the image is the left hand etc) the CNN most likely used this in its learning and so this is a possible, although easily rectifiable, issue. However we could fix this by predicting one hand at a time, once the first hand is detected we blank it out of the image and then run the image through the CNN again to find the second one.

Finally, although the CNN obviously had great detection for the seen training data, it also had great detection in the evaluation data, which whilst completely unseen, was taken immediately after the training data, with extra images taken at random from within the training video, i.e.

we took a video of us suturing and extracted every eleventh frame to be in the evaluation set, these images were not used at all in training but remain similar to those in training. There was poorer detection in new images, taken in the same location but with different environmental lighting. We suspect there was some element of overfitting however, since the evaluation data was truly never seen and was well detected we believe that the issue lies in the robustness, or lack there of, of the system and so further refinement of the model may be required.

7.3 Hand Detection Statistics

The Hand Detection system went through many iterations with various degrees of accuracy. Below we display the results from the various systems and models, the first few use the old dataset that contained examples of zero, one or two hands where as the last two use the new dataset that contains only suturing images with hands going naturally in and out of the images.

Tested on the old dataset, containing 10,000 training images and 1,000 evaluation images, we have:

1. The same model as the final CNN but for the old dataset, with 27,004 steps of training, i.e 27,004 images seen (27 time through the dataset):
   - Mean Absolute Error for hand 1: 5.12975
   - Mean Absolute Error for hand 2: 3.93975
   - Mean Squared Error for hand 1: 102.74625
   - Mean Squared Error for hand 2: 109.18825
   - Root Mean Squared Error for hand 1: 10.136382
   - Root Mean Squared Error for hand 2: 10.449318
2. This CNN model contained two LRN Layers, both Convolutional Layers had 64 filters, and there were 5 dense layers size decreasing in power two, i.e. first layer had 1204 neurons, second 512 etc, all of which had a dropout rate of 0.4, we also used a decay for the learning rate (initial learning rate was 0.01 with a decay of 0.01 for every 384 epoch's). This model had 26,003 training steps:
   - Mean Absolute Error for hand 1: 11.33725

- Mean Absolute Error for hand 2: 5.934
- Mean Squared Error for hand 1: 325.25726
- Mean Squared Error for hand 2: 159.1725
- Root Mean Squared Error for hand 1: 18.03489
- Root Mean Squared Error for hand 2: 12.616359

Tested on the new dataset, containing 8,000 training images and 174 evaluation images, we have:

1. The same model as the final CNN but with two Lrn Layers, with 62,007 steps of training:
   - Mean Absolute Error for hand 1: 5.038793
   - Mean Absolute Error for hand 2: 5.979885
   - Mean Squared Error for hand 1: 117.8865
   - Mean Squared Error for hand 2: 200.01724
   - Root Mean Squared Error for hand 1: 10.857554
   - Root Mean Squared Error for hand 2: 14.142745
2. The final CNN model with 108,005 steps of training:
   - Mean Absolute Error for hand 1: 4.8735633
   - Mean Absolute Error for hand 2: 5.1005745
   - Mean Squared Error for hand 1: 119.33046
   - Mean Squared Error for hand 2: 166.78162
   - Root Mean Squared Error for hand 1: 10.923848
   - Root Mean Squared Error for hand 2: 12.914395

As mentioned before the CNN used with the Adam optimizer assumed the hands were always in the same place so we don't show the evaluation here.

8 Conclusion

8.1 What we have accomplished

I have learnt a lot about the world of surgical skill assessment, from how it has evolved over time, how it is conducted nowadays, the pros and cons of this and how we may be able to improve it and the various benefits that result from this. By automating the assessment of trainee surgeons we free up professionals who's skills are in high demand with little supply given the years and years of training and proactive it takes to get to such a level. This results in faster turnover times for patients and reduces strain and stress on the NHS and other medical institutions. Moreover trainees get more practice, since they are no longer reliant upon a supervisor and so can train longer and still receive useful feedback.

I have also been able to see first hand how Machine Learning can be used either in conjunction with or to supersede a purely vision based system. Machine Learning, especially CNN's, are an accurate and powerful system for handling problems to do with images. I have had a lot of hands on experience with popular frameworks, mainly TensorFlow, and can see how the simplicity of CNN's can be so useful; having said that I personally believe the field has a long way to go given that far too much of building a CNN relies upon trial and error and just "knowing" what works with certain problems in general.

I look forward to the chance to try more combinations of the separate systems I have looked at, the sum of the parts is always less than the whole and I would find it very interesting to see if, with more time, would the FAST algorithm be more useful in the long run over Colour spaces? The majority of the trade-offs I made in this project came from what would be faster to run and take less time to implement. However with more time it would be simple to try a more varied mix of techniques.

9 Future Work

I fully intend to continue working on this system in one form or another. For my PhD I plan on building a learning system to teach a medical robotics platform a given surgery. Firstly I would need to adapt this system, instead of evaluating a trainee we want the system to learn what the optimal features of a successful surgery are, to learn its own understanding of what a good procedure is, as opposed to us feeding it evaluation metrics. Then we would further need to adapt this system to evaluate a robot as opposed to a human, there are many fascinating questions related to this, for example given an algorithm that can assess the skill of a human, what do we need to do to adapt it to teach a robot, how do robots differ to humans when performing the exact same task?

Following this we would then attempt to train a robot, that can only use a Vision System a series of Neural Networks working as a black box system and optionally some medical scans such as CT and MRI, to perform a basic procedures, such as suturing and knot tying. From this I would then aim to create a general learning framework which could be adapted to teach an autonomous robot to perform a given procedure. The proposed steps for this are as follows. Firstly we would run a virtual reconstruction of what would happen in reality. By using a graphical model with physically accurate components to simulate the surgery we can get an insight into how the system will work. Since the simulation is virtual the vision component of the robot would operate perfectly since its task is already completed by having the world as a mathematical object, the graphical representation. Furthermore the CT or MRI scans would also be perfectly generated, no anomalies, since the “patient” is virtual. Therefore this step essentially tests only the Neural Network’s ability to simplify the problem. Therefore we should be able to see if the surgery is feasible within the constraints of the robotic platform and can use this testing system to see what abilities the robotic platform needs to be able to perform a given surgical procedure.

Finally the robot would then perform the surgery on a physical model. This would attempt to add in the real world environment absent from the virtual world constructed in step two, to test the robustness of the system and the other elements not tested in the virtual setting.

Whilst using both of the testing systems, virtual and physical, we can run the robotic skill assessment algorithm to constantly update the robot based on how well it performed until it is over a particular threshold.

Alternatively this current project can be adapted to problems related to general hand tracking/detection problems. For example one idea was to create a real world heads up display (H.U.D) using a VR headset and an android mobile phone, since TensorFlow works with Java we could create an app which will allow us to do this. This could even be adapted to work as a security system for detection of shoplifters by tracking each persons hands and where they put the products on sale, i.e. in a shopping cart or in their pockets.

10 Appendix

10.1 User Guide

10.1.1 Dataset generation

The various dataset generation tools are in various python files. Since both CNN systems have their own defined data format requirements, they do not care how the data is generated. The code is there more to show what is possible for automating dataset generation and not particularly expected to be used.

10.1.2 CNN's

Both the CNN's have the same functionality when calling them from the command line, that is to say they work in the same way when you call them but obviously do different things. Depending on which is required one starts by typing "python surgical_cnn.py" or "python tip_cnn.py" followed by the command and the arguments [1].

[1]Note that one can always use -h or –help for a description of what to do.

Commands	
train	Trains the neural network, with the training data specified by the input function
eval	Evaluates the neural network, with the evaluation data specified by the input function
predict	predicts an image
all	predicts a sequence of images

Table 10.1: Commands for CNNs

Arguments	
-m/–model	Optional, chooses the model to use. This is a python file with a function called cnn_model_fn that is the model.
-v/–video	For all command, plays the sequence as a video
-s/–start	For all command, optional start index of the given sequence
-e/–end	For all command, optional end index of the given sequence
baseDir	positional argument for predict and all, where to look for the images
-sa/–save	Optional for all and predict, saves the output either as a sequence of images or a video as .mp4
-sp/–speed	For all command, speed at which to play the sequence, if a video

Table 10.2: Arguments for CNNs

Bibliography

[1] A. Zia, Y. Sharma, V. Bettadapura, E. L. Sarin, T. Ploetz, M. A. Clements, and I. Essa. Automated video-based assessment of surgical skills for training and evaluation in medical schools. *International journal of computer assisted radiology and surgery*, 11(9):1623–1636, September 01 2016. LR: 20171104; JID: 101499225; OTO: NOTNLM; 2016/01/27 00:00 [received]; 2016/08/03 00:00 [accepted]; 2016/08/29 06:00 [entrez]; 2016/08/29 06:00 [pubmed]; 2017/02/01 06:00 [medline]; ppublish.

[2] Hamed Pirsiavash, Carl Vondrick, and Antonio Torralba. *Assessing the Quality of Actions*, pages 556–571. Computer Vision – ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part VI. Springer International Publishing, Cham, 2014. ID: Pirsiavash2014.

[3] M. Neumann, C. Hahn, T. Horbach, I. Schneider, A. Meining, W. Heldwein, T. Rosch, and W. Hohenberger. Score card endoscopy: a multicenter study to evaluate learning curves in 1-week courses using the erlangen endo-trainer. *Endoscopy*, 35(6):515–520, Jun 2003. LR: 20041117; JID: 0215166; 2003/06/05 05:00 [pubmed]; 2003/12/25 05:00 [medline]; 2003/06/05 05:00 [entrez]; ppublish.

[4] Carol Reiley, Henry C Lin, David D Yuh, and Gregory Hager. *Review of methods for objective surgical skill evaluation*, volume 25. 2011.

[5] Richard K. Reznick and Helen MacRae. Teaching surgical skills — changes in the wind. *N Engl J Med*, 355(25):2664–2669, 2006. doi: 10.1056/NEJMra054785; 25.

[6] C. J. DeFrances and M. J. Hall. 2005 national hospital discharge survey. *Advance Data*, (385)(385):1–19, Jul 12 2007. LR: 20070813; JID: 7703830; 2007/08/19 09:00 [pubmed]; 2007/08/28 09:00 [medline]; 2007/08/19 09:00 [entrez]; ppublish.

[7] C. Zhan and M. R. Miller. Excess length of stay, charges, and mortality attributable to medical injuries during hospitalization. *Jama*, 290(14):1868–1874, Oct 8 2003. LR: 20161017; JID: 7501160; CIN: JAMA. 2004 Jan 21;291(3):304; author reply 304-5. PMID: 14734591; CIN: JAMA. 2004 Jan 21;291(3):304; author reply 304-5. PMID: 14734590; CIN: JAMA. 2003 Oct 8;290(14):1917-9. PMID: 14532322; CIN: JAMA. 2004 Jan 21;291(3):303-

4; author reply 304-5. PMID: 14734589; 2003/10/09 05:00 [pubmed]; 2003/10/15 05:00 [medline]; 2003/10/09 05:00 [entrez]; ppublish.

[8] C. B. Barden, M. C. Specht, M. D. McCarter, J. M. Daly, and T. J. Fahey 3rd. Effects of limited work hours on surgical training. *Journal of the American College of Surgeons*, 195(4):531–538, Oct 2002. LR: 20081121; JID: 9431305; CIN: J Am Coll Surg. 2003 Apr;196(4):661. PMID: 12691949; CIN: J Am Coll Surg. 2003 Apr;196(4):661-2. PMID: 12691950; CIN: J Am Coll Surg. 2003 Apr;196(4):662-3. PMID: 12691951; 2002/10/12 04:00 [pubmed]; 2002/12/04 04:00 [medline]; 2002/10/12 04:00 [entrez]; ppublish.

[9]

[10] Francis NK, Hanna GB, and Cuschieri A. The performance of master surgeons on the advanced dundee endoscopic psychomotor tester: Contrast validity study. *Archives of Surgery*, 137(7):841–844, 2002.

[11] S. D. Bann, V. K. Datta, M. S. Khan, P. F. Ridgway, and A. W. Darzi. Attitudes towards skills examinations for basic surgical trainees. *International journal of clinical practice*, 59(1):107–113, 2005.

[12] K. A. Ericsson. Deliberate practice and the acquisition and maintenance of expert performance in medicine and related domains. *Academic medicine : journal of the Association of American Medical Colleges*, 79(10 Suppl):70, Oct 2004. LR: 20051116; JID: 8904605; RF: 119; 2004/09/24 05:00 [pubmed]; 2005/03/04 09:00 [medline]; 2004/09/24 05:00 [entrez]; ppublish.

[13] Alberto Peracchia. Surgical education in the third millennium. *Annals of Surgery*, 234(6):709–712, 2001. J1: Ann Surg.

[14] Ara Darzi, Simon Smith, and Nick Taffinder. Assessing operative skill : Needs to become more objective. *BMJ : British Medical Journal*, 318(7188):887–888, 1999. J1: BMJ.

[15] Iscp. Intercollegiate Surgical Curriculum. "https://www.iscp.ac.uk/curriculum/surgical/surgical_sylla
14 June, 2018 11:27. [Online; last accessed 16-June-2018].

[16] J. A. Kopta. An approach to the evaluation of operative skills. *Surgery*, 70(2):297–303, Aug 1971. LR: 20081121; JID: 0417347; 1971/08/01 00:00 [pubmed]; 1971/08/01 00:01 [medline]; 1971/08/01 00:00 [entrez]; ppublish.

[17] S. E. Scallon, D. J. Fairholm, D. D. Cochrane, and D. C. Taylor. Evaluation of the operating room as a surgical teaching venue. *Canadian journal of surgery.Journal canadien de chirurgie*, 35(2):173–176, Apr 1992. LR: 20081121; JID: 0372715; 1992/04/01 00:00 [pubmed]; 1992/04/01 00:01 [medline]; 1992/04/01 00:00 [entrez]; ppublish.

[18] J. Heppell, G. Beauchamp, and A. Chollet. Ten-year experience with a basic technical skills and perioperative management workshop for first-year residents. *Canadian journal of surgery.Journal canadien de chirurgie*, 38(1):27–32, Feb 1995. LR: 20081121; JID: 0372715; CIN: Can J Surg. 1995 Feb;38(1):8-9. PMID: 7882216; 1995/02/01 00:00 [pubmed]; 1995/02/01 00:01 [medline]; 1995/02/01 00:00 [entrez]; ppublish.

[19] A. G. Lossing, E. M. Hatswell, T. Gilas, R. K. Reznick, and L. C. Smith. A technical-skills course for 1st-year residents in general surgery: a descriptive study. *Canadian journal of surgery.Journal canadien de chirurgie*, 35(5):536–540, Oct 1992. LR: 20081121; JID: 0372715; 1992/10/01 00:00 [pubmed]; 1992/10/01 00:01 [medline]; 1992/10/01 00:00 [entrez]; ppublish.

[20] J. H. Cauraugh, M. Martin, and K. K. Martin. Modeling surgical expertise for motor skill acquisition. *American Journal of Surgery*, 177(4):331–336, Apr 1999. LR: 20041117; JID: 0370473; 1999/05/18 00:00 [pubmed]; 1999/05/18 00:01 [medline]; 1999/05/18 00:00 [entrez]; ppublish.

[21] R. K. Reznick. Teaching and testing technical skills. *American Journal of Surgery*, 165(3):358–361, Mar 1993. LR: 20171026; JID: 0370473; 1993/03/01 00:00 [pubmed]; 1993/03/01 00:01 [medline]; 1993/03/01 00:00 [entrez]; ppublish.

[22] Paul M. Fitts and Michael I. Posner. *Human performance.* Brooks/Cole, Belmont, Calif, 1967.

[23] J. A. Kopta. The development of motor skills in orthopaedic education. *Clinical orthopaedics and related research*, 75:80–85, 1971. LR: 20050303; JID: 0075674; 1971/03/01 00:00 [pubmed]; 1971/03/01 00:01 [medline]; 1971/03/01 00:00 [entrez]; ppublish.

[24] K. Moorthy, Y. Munz, A. Dosis, F. Bello, A. Chang, and A. Darzi. Bimodal assessment of laparoscopic suturing skills: Construct and concurrent validity. *Surgical Endoscopy And Other Interventional Techniques*, 18(11):1608–1612, Nov 2004.

[25] Zeev Friedman, Rita Katznelson, Isabel Devito, Mughina Siddiqui, and Vincent Chan. Objective assessment of manual skills and proficiency in performing epidural anesthesia—video-assisted validation. *Regional Anesthesia and Pain Medicine*, 31(4):304 – 310, 2006.

[26] Steven E. Swift and James F. Carter. Institution and validation of an observed structured assessment of technical skills (osats) for obstetrics and gynecology residents and faculty. *American Journal of Obstetrics and Gynecology*, 195(2):617 – 621, 2006.

[27] Philine A. van der Heide, Letty van Toledo-Eppinga, Maaike van der Heide, and Johanna H. van der Lee. Assessment of neonatal resuscitation skills: A reliable and valid scoring system. *Resuscitation*, 71(2):212 – 221, 2006.

[28] Anders J. Debes, Rajesh Aggarwal, Indran Balasundaram, and Morten B.J. Jacobsen. Construction of an evidence-based, graduated training curriculum for d-box, a webcam-based laparoscopic basic skills trainer box. *The American Journal of Surgery*, 203(6):768 – 775, 2012.

[29] JAMES D. WATTERSON, DARREN T. BEIKO, JAMES K. KUAN, and JOHN D. DENSTEDT. A randomized prospective blinded study validating acquistion of ureteroscopy skills using a computer based virtual reality endourological simulator. *The Journal of Urology*, 168(5):1928 – 1932, 2002.

[30] A. Alvand, T. Khan, S. Al-Ali, W. F. Jackson, A. J. Price, and J. L. Rees. Combining local appearance and holistic view: Dual-source deep neural networks for human pose estimation. *J. Bone Joint Surg. Am.*, vol.94, no. 13, 2012.

[31] "Richard Reznick, Glenn Regehr, Helen MacRae, Jenepher Martin, and Wendy McCulloch". "testing technical skill via an innovative "bench station" examination". *"The American Journal of Surgery"*, "173"("3"):"226 – 230", "1997".

[32] R. C. King, L. Atallah, B. P. L. Lo, and G. Z. Yang. Development of a wireless sensor glove for surgical skills assessment. *IEEE Transactions on Information Technology in Biomedicine*, 13(5):673–679, Sept 2009.

[33] Aaron Insel, Bradley Carofino, Robin Leger, Robert Arciero, and Augustus D. Mazzocca. The development of an objective model to assess arthroscopic performance. *JBJS*, 91(9), 2009. ID: 00004623-200909000-00028.

[34] Abtin Alvand, Kartik Logishetty, Robert Middleton, Tanvir Khan, William F.M. Jackson, Andrew J. Price, and Jonathan L. Rees. Validating a global rating scale to monitor individual resident learning curves during arthroscopic knee meniscal repair. *Arthroscopy: The Journal of Arthroscopic & Related Surgery*, 29(5):906 – 912, 2013.

[35] N. R. Howells, H. S. Gill, A. J. Carr, A. J. Price, and J. L. Rees. Transferring simulated arthroscopic skills to the operating theatre. *The Journal of Bone and Joint Surgery. British volume*, 90-B(4):494–499, 2008.

[36] 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. PAPERS FROM THE NORTH PACIFIC SURGICAL ASSOCIATION.

[37] Thomas R Eubanks, Ronald H Clements, Dieter Pohl, Noel Williams, Douglas C Schaad, Santiago Horgan, and Carlos Pellegrini. An objective scoring system for laparoscopic cholecystectomy. *Journal of the American College of Surgeons*, 189(6):566 – 574, 1999.

[38] Melina C. Vassiliou, Liane S. Feldman, Christopher G. Andrew, Simon Bergman, Karen Leffondré, Donna Stanbridge, and Gerald M. Fried. A global assessment tool for evaluation of intraoperative laparoscopic skills. *The American Journal of Surgery*, 190(1):107 – 113, 2005.

[39] Melina C. Vassiliou, Pepa A. Kaneva, Benjamin K. Poulose, Brian J. Dunkin, Jeffrey M. Marks, Riadh Sadik, Gideon Sroka, Mehran Anvari, Klaus Thaler, Gina L. Adrales, Jeffrey W. Hazey, Jenifer R. Lightdale, Vic Velanovich, Lee L. Swanstrom, John D. Mellinger, and Gerald M. Fried. Global assessment of gastrointestinal endoscopic skills (gages): a valid measurement tool for technical skills in flexible endoscopy. *Surgical Endoscopy*, 24(8):1834–1841, Aug 2010.

[40] Rajesh Aggarwal, Teodor Grantcharov, Krishna Moorthy, Thor Milland, and Ara Darzi. Toward feasible, valid, and reliable video-based assessments of technical surgical skills in the operating room. *Annals of Surgery*, 247(2), 2008. ID: 00000658-200802000-00025.

[41] Allan Okrainec, Melina Vassiliou, Andrew Kapoor, Kristen Pitzul, Oscar Henao, Pepa Kaneva, Timothy Jackson, and E. Matt Ritter. Feasibility of remote administration of the fundamentals of laparoscopic surgery (fls) skills test. *Surgical Endoscopy*, 27(11):4033–4037, Nov 2013.

[42] van Hove P. D., Tuijthof G. J. M., Verdaasdonk E. G. G., Stassen L. P. S., and Dankelman J. Objective assessment of technical surgical skills. *BJS*, 97(7):972–987.

[43] H. Gill A. Alvand, S. Auplish and J. Rees. Innate arthroscopic skills in medical students and variation in learning curves. *J. Bone Joint Surg. Am.*, vol. 93, no. 19, pp. e115(1–9), 2011.

[44] Nick R. Howells, Mark D. Brinsden, Richie S. Gill, Andrew J. Carr, and Jonathan L. Rees. Motion analysis: A validated method for showing skill levels in arthroscopy. *Arthroscopy: The Journal of Arthroscopic & Related Surgery*, 24(3):335 – 342, 2008.

[45] 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 model1 1no competing interests declared. *Journal of the American College of Surgeons*, 193(5):479 – 485, 2001.

[46] A. Alvand, S. Auplish, T. Khan, H. S. Gill, and J. L. Rees. Identifying orthopaedic surgeons of the future; the inability of some medical students to achieve competence in basic arthroscopic tasks despite training: a randomised study. *The Journal of Bone and Joint Surgery. British volume*, 93-B(12):1586–1591, 2011.

[47] Krishna Moorthy, Yaron Munz, Sudip K Sarker, and Ara Darzi. Objective assessment of technical skills in surgery. *BMJ*, 327(7422):1032–1037, 2003.

[48] 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 model1 1no competing interests declared., 2001. ID: 271148.

[49] N. Taffinder, C. Sutton, R. J. Fishwick, I. C. McManus, and A. Darzi. Validation of virtual reality to teach and assess psychomotor skills in laparoscopic surgery: results from randomised controlled studies using the mist vr laparoscopic simulator. *Studies in health*

technology and informatics, 50:124–130, 1998. LR: 20081121; JID: 9214582; 1997/12/08 00:00 [pubmed]; 1997/12/08 00:01 [medline]; 1997/12/08 00:00 [entrez]; ppublish.

[50] V. Datta, S. Mackay, M. Mandalia, and A. 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, Nov 2001. LR: 20081121; JID: 9431305; 2001/11/16 10:00 [pubmed]; 2002/01/05 10:01 [medline]; 2001/11/16 10:00 [entrez]; ppublish.

[51] A. Darzi and S. Mackay. Assessment of surgical competence. *Qual Health Care*, 10:ii64, 2001.

[52] Carol E. "Reiley, Henry C. Lin, David D. Yuh, and Gregory D." Hager. "review of methods for objective surgical skill evaluation". *"Surgical Endoscopy"*, "25"("2"):"356–366", "Feb" "2011".

[53] M. P. Schijven, J. Jakimowicz, and C. Schot. The advanced dundee endoscopic psychomotor tester (adept) objectifying subjective psychomotor test performance. *Surgical Endoscopy And Other Interventional Techniques*, 16(6):943–948, Jun 2002.

[54] M. K. Chmarra, C. A. Grimbergen, and J. Dankelman. Systems for tracking minimally invasive surgical instruments. *Minimally Invasive Therapy & Allied Technologies*, 16(6):328–340, 2007. doi: 10.1080/13645700701702135.

[55] Narges Ahmidi, Gregory D. Hager, Lisa Ishii, Gabor Fichtinger, Gary L. Gallia, and Masaru Ishii. Surgical task and skill classification from eye tracking and tool motion in minimally invasive surgery. In Tianzi Jiang, Nassir Navab, Josien P. W. Pluim, and Max A. Viergever, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2010*, pages 295–302, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[56] N. E. Seymour, A. G. Gallagher, S. A. Roman, M. K. O'Brien, V. K. Bansal, D. K. Andersen, and R. M. Satava. Virtual reality training improves operating room performance: results of a randomized, double-blinded study. *Annals of Surgery*, 236(4):4, Oct 2002. LR: 20140611; JID: 0372354; OID: NLM: PMC1422600; 2002/10/09 04:00 [pubmed]; 2002/10/31 04:00 [medline]; 2002/10/09 04:00 [entrez]; ppublish.

[57] T. P. Grantcharov, V. B. Kristiansen, J. Bendix, L. Bardram, J. Rosenberg, and P. Funch-Jensen. Randomized clinical trial of virtual reality simulation for laparoscopic skills training. *The British journal of surgery*, 91(2):146–150, Feb 2004. LR: 20081121; CI: Copyright 2003; JID: 0372553; 2004/02/05 05:00 [pubmed]; 2004/02/26 05:00 [medline]; 2004/02/05 05:00 [entrez]; ppublish.

[58] S. Cotin, S. L. Dawson, D. Meglan, D. W. Shaffer, M. A. Ferrell, R. S. Bardsley, F. M. Morgan, T. Nagano, J. Nikom, P. Sherman, M. T. Walterman, and J. Wendlandt. Icts, an interventional cardiology training system. *Studies in health technology and informatics*, 70:59–65, 2000. LR: 20121115; JID: 9214582; 2000/09/08 11:00 [pubmed]; 2000/09/08 11:01 [medline]; 2000/09/08 11:00 [entrez]; ppublish.

[59] A. G. Gallagher and C. U. Cates. Approval of virtual reality training for carotid stenting: what this means for procedural-based medicine. *Jama*, 292(24):3024–3026, Dec 22 2004. LR: 20161017; JID: 7501160; CIN: JAMA. 2005 May 4;293(17):2091; author reply 2091-2. PMID: 15870409; 2004/12/23 09:00 [pubmed]; 2004/12/28 09:00 [medline]; 2004/12/23 09:00 [entrez]; ppublish.

[60] E. D. Matsumoto, S. J. Hamstra, S. B. Radomski, and M. D. Cusimano. The effect of bench model fidelity on endourological skills: a randomized controlled study. *The Journal of urology*, 167(3):1243–1247, Mar 2002. LR: 20041117; JID: 0376374; 2002/02/08 10:00 [pubmed]; 2002/03/01 10:01 [medline]; 2002/02/08 10:00 [entrez]; ppublish.

[61] D. J. Anastakis, G. Regehr, R. K. Reznick, M. Cusimano, J. Murnaghan, M. Brown, and C. Hutchison. Assessment of technical skills transfer from the bench training model to the human model. *American Journal of Surgery*, 177(2):167–170, Feb 1999. LR: 20081121; JID: 0370473; 1999/04/16 00:00 [pubmed]; 1999/04/16 00:01 [medline]; 1999/04/16 00:00 [entrez]; ppublish.

[62] TensorFlow. Local response normalization. https://www.tensorflow.org/api_docs/python/tf/nn/local_ May 25, 2018. [Online; last accessed 16-June-2018].

[63] TensorFlow. Dropout. https://www.tensorflow.org/api_docs/python/tf/nn/dropout, May 25, 2018. [Online; last accessed 16-June-2018].

[64] L. Zhang, L. Zhang, and B. Du. Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, 4(2):22–40, 2016.

[65] Wikipedia. Rectifier (neural networks). https://en.wikipedia.org/wiki/Rectifier_(neural_networks), 26 May 2018, at 17:25. [Online; last accessed 16-June-2018].

[66] Wikipedia. Regularization (mathematics). https://en.wikipedia.org/wiki/Regularization_(mathematic
6 May 2018, 15:55. [Online; last accessed 16-June-2018].

[67] Ritchie Ng. Regularization with TensorFlow. https://www.ritchieng.com/machine-learning/deep-learning/tensorflow/regularization/ , Jun 11, 2018. [Online; last accessed 16-June-2018].

[68] Jahnavi Mahanta. Keep it simple! How to understand Gradient Descent algorithm. https://www.kdnuggets.com/2017/04/simple-understand-gradient-descent-algorithm.html, Apr 2017. [Online; last accessed 16-June-2018].

[69] Matthew D. Zeiler. ADADELTA: an adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[70] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[71] Wikipedia. Taylor series. https://en.wikipedia.org/wiki/Taylor_series, 14 June, 2018 at 11:27. [Online; last accessed 16-June-2018].

[72] Wikipedia. Sobel operator. https://en.wikipedia.org/wiki/Sobel_operator, 21 April 2018, at 23:07. [Online; last accessed 16-June-2018].

[73] OpenCV. Harris Corner Detection. https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_features_harris/py_features_harris.html#harris-corners, Nov 10, 2014. [Online; last accessed 16-June-2018].

[74] OpenCV. Introduction to SIFT. https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_sift_intro/py_sift_intro.html#sift-intro, Nov 10, 2014. [Online; last accessed 16-June-2018].

[75] Wikipedia. Blob detection. https://en.wikipedia.org/wiki/Blob_detection#The_Laplacian_of_Gaussian
, 21 April 2018, at 23:07. [Online; last accessed 16-June-2018].

[76] OpenCV. Introduction to SURF. https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_surf_intro/py_surf_intro.html#surf, Nov 10, 2014. [Online; last accessed 16-June-2018].

[77] OpenCV. FAST Algorithm for Corner Detection. https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_feature2d/py_fast/py_fast.html#fast, Nov 10, 2014. [Online; last accessed 16-June-2018].