An Interactive Augmented-Reality Video Training Platform for the da Vinci Robotic Surgical System

NSF Summer Undergraduate Fellowship in Sensor Technologies
Jaimie Carlson, Rachleff Scholar (Bioengineering, Computer and Information Science) - School of Engineering and Applied Science, University of Pennsylvania
Advisor: Dr. Kuchenbecker (Mechanical Engineering and Applied Mechanics, Computer and Information Science)

ABSTRACT

Robotic surgical systems such as the da Vinci Surgical System support minimally invasive surgeries which decrease risk to patients and hospital stay time. However, the learning curve on the da Vinci system is quite steep, and existing training curriculums on surgical simulators primarily offer students the opportunity to practice basic tasks rather than experience a full, realistic operation. This paper presents an augmented-reality video training platform for the da Vinci which will allow medical students to rehearse any variety of real surgery, with any possible complication. While a user operates the da Vinci, the video of tool movements is extracted from the surgical system and overlaid on the "teacher" video of an expert so that the user can mirror the expert's motion. This combined video is streamed back into the stereoscopic eyepiece of the da Vinci. Tools are identified in the image via color segmentation and kernelized correlation filter tracking, and their depth is calculated from the da Vinci's stereoscopic video feed. If any of the user's tools venture too far away from the experts', the system will alert the user and pause to let them catch up. Statistical feedback on path length and distance between user and expert tools is given to the user at the end of the program.

Table of Contents

1. INTRODUCTION AND BACKGROUND

1.1 Introduction to Robotic Surgery

The developing field of robotic surgery facilitates surgeons' completion of complicated and potentially life-threatening procedures. It combines the mobility of the human hand with the small incisions seen in laparoscopy to ensure a smooth, minimally invasive operation for both physician and patient. A doctor using a robotic surgical system, such as the prominent da Vinci Surgical System (Intuitive, Sunnyvale, CA), sits at a master console, which is linked to a slave console. This slave console has two to four arms with interchangeable tools and one endoscope with stereoscopic camera which are inserted into small incisions in the body. The device features a three-dimensional eyepiece and highly mobile (7-DOF) grippers on the master console [1]. With these, a surgeon can complete a complex operation with greater agility and vision than a laparoscopic procedure.

Many concrete benefits have been observed in procedures utilizing robotic surgical systems. Minimally invasive procedures such as laparoscopy or robotic surgery feature lower pain and faster recovery time than open surgery. Furthermore, robotic surgery offers benefits over even laparoscopy, such as instruments with wrist joints and the filtering out of any low-amplitude tremors in the surgeon's hand motion [2]. Robotic surgical systems such as the da Vinci have been successfully used in operations as diverse as cholecystectomies, prostatectomies, kidney transplants, and endoscopic coronary artery grafting [1]. Researchers have observed that patients who undergo robotic surgeries often have shorter hospitalization times and fewer post-surgical complications in abdominal [3] and gynecological surgery [4], and lower blood loss in tumor removal [5], [6], as compared to laparoscopic or open surgery.

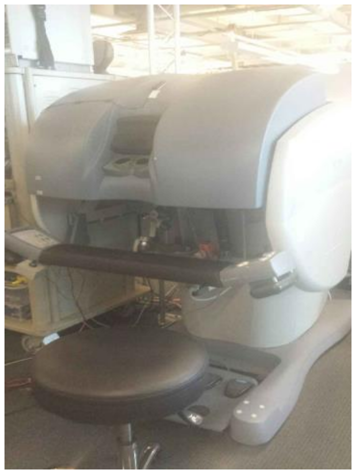
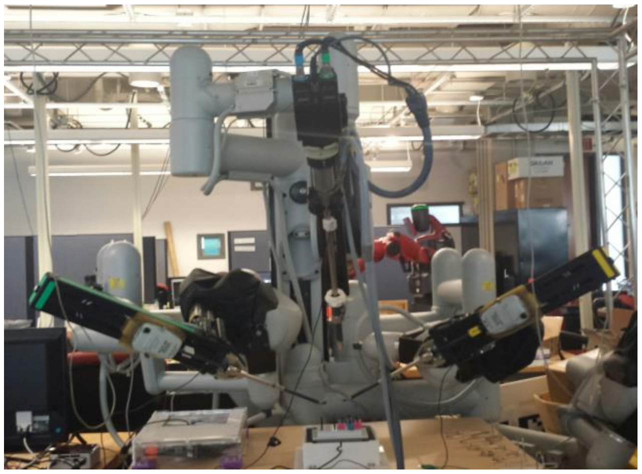



Figure 1a: da Vinci Surgeon Console; Figure 1b: da Vinci Patient Console

However, robotic surgery systems have several disadvantages. First, they currently lack haptic feedback, meaning that surgeons cannot feel how hard they are pressing on tissue. Surgeons generally prefer to have this vital feedback, which makes them more aware of their anatomical surroundings and prevents them from accidentally breaking vital tissue [7]. Second, they often have a very steep learning curve. Studies have shown that gaining proficiency in the use of the da Vinci machine can take a significant amount of time; however, surgical residents often have limited access to these highly coveted machines.

1.2 Learning Curve for Robotic Surgery

The learning curve for robotic surgery is often quite steep. Although the learning curve required to improve at fundamental skills such as suturing or knot-tying can take as short as ninety minutes [8], six hours [9], or within five trials [10], more difficult skills take more time. Long-term surgical studies found that the learning curve can take many real-life cases to overcome: the learning curve for surgeons can take about 15 cases for colorectal [11], and 50 cases for gynecological procedures [12]. However, decreases in operating time and/or positive surgical margin can be found after 33 [13], 50 [14], or over 100 [15] cases of robotic-assisted radical prostatectomies.

1.3 Existing Simulators

Clearly, the performance of many actual robotic surgeries is required for improvement. However, most learning procedures for robotic surgery do not include simulation of full surgeries.

Residents often train by either using a da Vinci machine itself or by using a simulator. The da Vinci Skills Simulator, or dVSS [16], attaches to an actual da Vinci machine, and can drill students in a library of fundamental skills such as basic manipulation, camera targeting, and suturing. Other simulators require the purchase of a separate physical simulator, such as the Mimic dV-Trainer, or MdVT [17], and the Robotic Surgery Simulator, or RoSS [18]. All of these simulators drill students in a library of fundamental skills such as camera targeting, suturing, energy use, and tissue dissection.

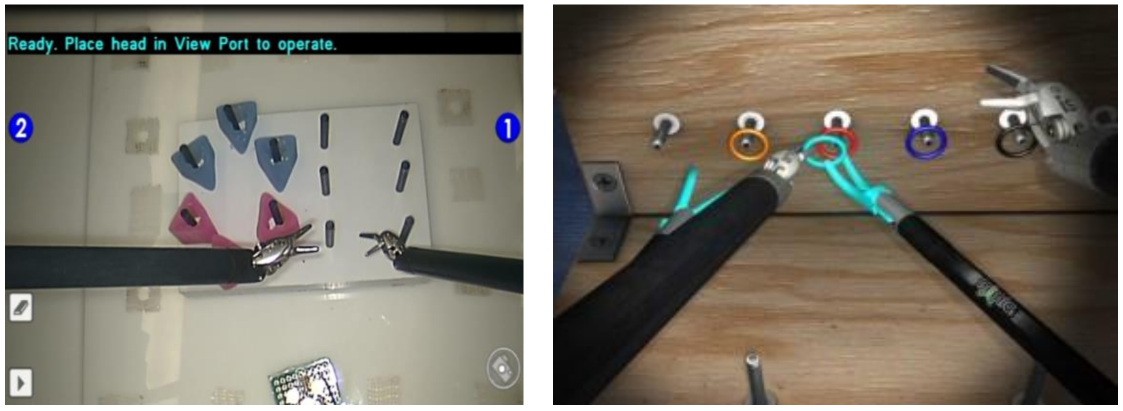


Figure 2a: Actual peg board task; Figure 2b: Simulated ring transfer task [17]

Many studies have confirmed face and content validity for the dVSS, MdVT, and RoSS, as well as construct validity on varying tasks with varying metrics [19]. Several skills transfer studies also determined that students practicing on the MdVT, RoSS, and dVSS with standard tasks could improve their performance on inanimate or live models [19].

Recently, developments have been made in the creation of more realistic simulators. The ROSS was the first system to do so, creating a Hands-On Surgical Training, or HoST, module in addition to its regular training modules [20]. It shows users real video from an operation while guiding their hands through the movements taken by the original surgeon, providing additional information and anatomical annotations on top of the video.

However, very recently, an increasing number of companies have explored this idea. The Robotix Mentor uses a separate platform to simulate a variety of tasks such as prostatectomy and hysterectomy [21]. It divides the tasks into sub-tasks to aid learning and allows the user to pre-program organ placement. In addition, the MdVT recently piloted the Maestro AR [17], which uses the dV-Trainer to control the image of computer-generated tools projected onto three real surgical videos: partial nephrectomy, hysterectomy, and inguinal hernia repair. However, face and content validity have not yet been proven for any of these new simulation systems; in addition, they do not run on the da Vinci machine itself. This makes the experience less realistic for the trainee and less accessible to hospitals who perhaps possess a da Vinci machine but have not purchased an expensive simulator.

A newly-launched addition to the da Vinci Xi Skills Simulator builds off the Robotix Mentor simulation platform, but runs on the da Vinci machine itself. It is able to simulate an entire hysterectomy, as well as specific procedural tasks within it, providing in-operation guidance and the ability to program certain organ placements. [21]. However, it uses three-dimensional simulation instead of using augmented real video like the ROSS HoST or MdVT Maestro.

So far, the only one of these systems with proven face and content validity is the ROSS HoST; its skills transfer ability was proven when students using the ROSS HoST to practice urethrovesical anastomosis performed better than those who simply viewed videos of the procedure [22]. However, the other systems' validity in simulating a whole procedure has not been tested. It has been noted that virtual reality training in general has not yet been conclusively proven to transfer skills to robotic surgery [19], [23].

1.4 Aims of Proposed System

The proposed software will allow a user to operate on an actual da Vinci Surgical System console. It is based upon combining previously filmed videos of expert surgeons performing specific surgeries ("teacher video") with the stereoscopic live feed from the da Vinci endoscope of a learner ("student video"). The student, sitting at the da Vinci console and operating the master grippers in the conventional manner, will view an overlay of their instruments onto the background of the real-time teacher video. The student will be able to "follow along" with the teacher video, actively trying to emulate the efficiency and economy of motion of an expert surgeon by trying to keep their tools as close as possible to the surgeon's in the teacher video. The teacher video feed will pause if the student's tools are too far from the teacher's tools, as determined by a three-dimensional model gained from stereoscopic video, and allow the student to catch up. In the end, automatic scoring of such features like time to complete surgery, number of pauses, distance from teacher's to student's tool, and student's path length will be presented to the student for immediate feedback.

This proposed system will improve upon previous efforts. Unlike the RoSS HoST, MdVT Maestro, or Robotix Mentor, it will run on the da Vinci machine itself, making it more realistic and accessible. It will also be able to run on any model of da Vinci machine and is not just tailored for a specific generation. As long as calibration is performed on a da Vinci system (a process taking under an hour), that system can be used with the application. Furthermore, the Robotix Mentor or the new addition to the da Vinci Xi Skills Simulator use orderly-looking but not overly convincing simulated organ environment, whereas this application will run on real video.

Most importantly, unlike all aforementioned systems, which are built upon specific procedures and slowly release new modules for individual types of operation, the proposed system will be able to step the viewer through any type of expert surgery, as long as a previous video of it has been taken. Instead of waiting on the release of new pre-programmed simulations of the type of surgery in which they specialize, surgeons can use their own old operation videos to train students. Thus, the student will be able to learn directly the motion patterns of an expert in this realistic setting, instead of following a series of instructions as in the da Vinci Xi Skills Simulator.

Furthermore, this will allow students to practice what to do in the event of unforeseen complications. Although the newest da Vinci Xi Skills Simulator allows the placement of internal organs to be adjusted beforehand in the simulation, creating some variation, none of the simulators' modules allow for more widely varied problems which a surgeon might face, such as large amounts of adipose tissue obscuring a view, organs of unusual size and shape as well as position, or internal bleeding. In the proposed system, any complication or anatomical variation which has been previously observed and recorded on video can be presented to the student.

Overall, this system will provide a way for medical students to train on the da Vinci machine by superimposing their tool motion on top of that of an expert surgeon. It offers a realistic setting, a multitude of possibilities for the type and course of the operation, and the opportunity to learn from an expert surgeon.

2. APPLICATION DESIGN

2.1 Software Specifications

The program was developed in C++, with the use of the OpenCV libraries [24]. The application's graphical user interface was designed with QT Creator. Individual user data was saved and retrieved using an SQL database. Preprocessing of videos was done in MATLAB.

2.2 Hardware Specifications

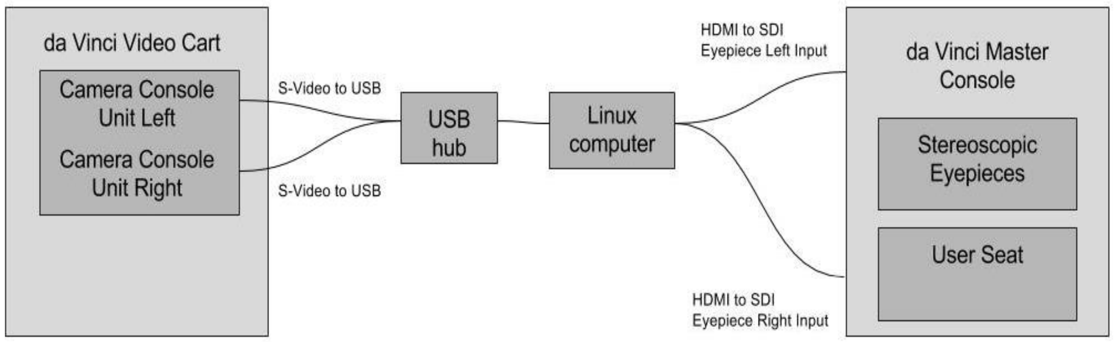
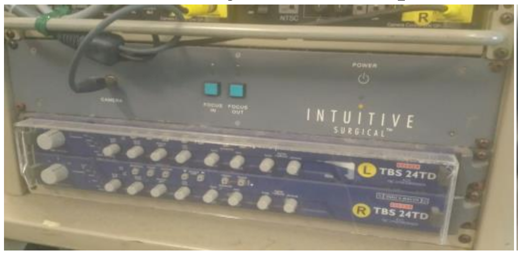


Figure 3: Hardware Diagram

The program was run on a Linux machine and connected to a first generation da Vinci Surgical System. Incoming student video was obtained from left and right camera control units (CCUs) on the da Vinci Surgical System vision cart. S-video output from the vision cart was converted to USB format with two Sabrent S-Video to USB capture cables, then connected through USB 2.0 ports to the Linux machine. The outgoing left and right video feed created by the Linux machine was sent out through two HDMI cables, then through HDMI to SDI converters, and fed into the left and right video inputs in HD-SDI format in the back of the surgeon console.

The da Vinci Surgical System serves as the student console. On the left tool of the da Vinci machine, a green tape marker is placed; on the right tool of the da Vinci machine, a blue tape marker is placed. These colors were chosen due to their relative infrequency in natural tissue, and they are used to pick out the original tools.



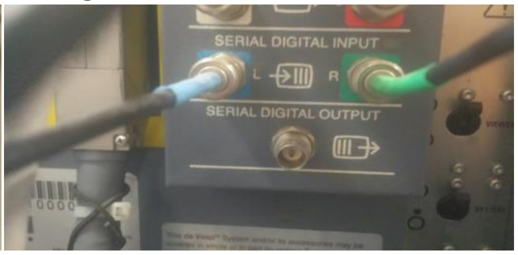
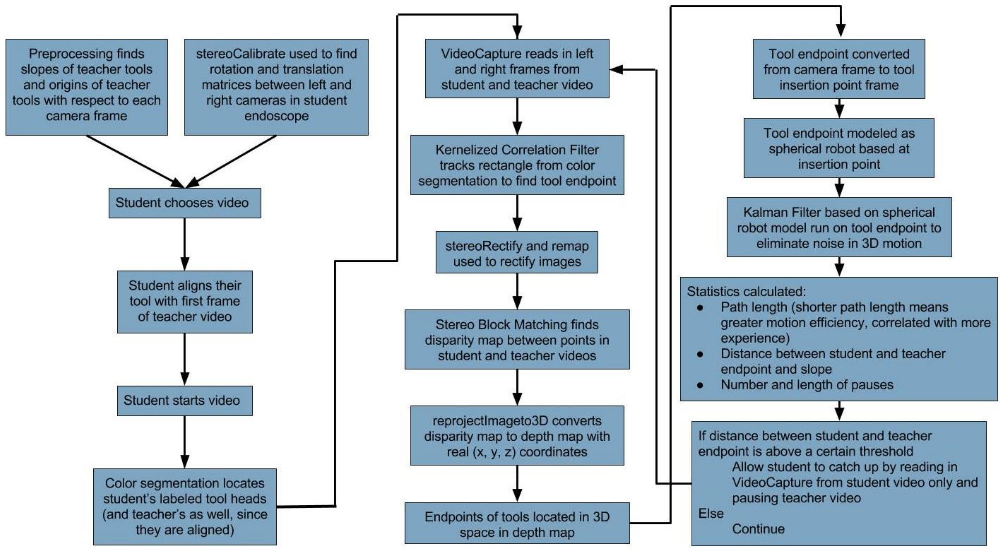


Figure 4a: Camera Control Units; Figure 4b: SDI input into da Vinci

Figure 5a: da Vinci stereoscopic eyepiece; Figure 5b: USB to S-Video converter

3. DESIGN AND RESULTS


Preprocessing finds slopes of teacher tools and origins of teacher tools with respect to each camera frame
stereoCalibrate used to find rotation and translation matrices between left and right cameras in student endoscope
Student chooses video
Student aligns their tool with first frame of teacher video
Student starts video
Color segmentation locates student's labeled tool heads (and teacher's as well, since they are aligned)
VideoCapture reads in left and right frames from student and teacher video
Kernelized Correlation Filter tracks rectangle from color segmentation to find tool endpoint
stereoRectify and remap used to rectify images
Stereo Block Matching finds disparity map between points in student and teacher videos
reprojectImageto3D converts disparity map to depth map with real (x, y, z) coordinates
Endpoints of tools located in 3D space in depth map
Tool endpoint converted from camera frame to tool insertion point frame
Tool endpoint modeled as spherical robot based at insertion point
Kalman Filter based on spherical robot model run on tool endpoint to eliminate noise in 3D motion
Statistics calculated:
Path length (shorter path length means greater motion efficiency, correlated with more experience)
Distance between student and teacher endpoint and slope
Number and length of pauses
If distance between student and teacher endpoint is above a certain threshold
Allow student to catch up by reading in VideoCapture from student video only and pausing teacher video
Else
Continue
Figure 6: Program Flowchart

3.1 Preprocessing

This algorithm models the da Vinci machine arm shaft as a spherical robot, with two degrees of freedom in rotation and one degree of freedom in extension. The joint of the arm itself has many more degrees of freedom; however, the shaft of the arm has a fixed insertion point (an incision into the body or a hole into a hollow shell modeling an inflated torso). The surgeon can rotate that arm around that insertion point with two degrees of freedom, or the surgeon can pull the arm in or out of the incision.

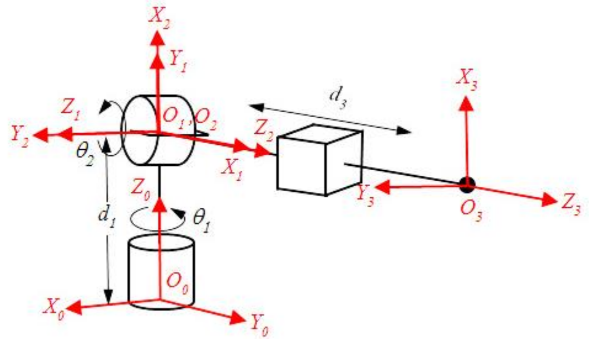


Figure 7: Spherical Robot Model

Here, it is assumed that d1, d2, and d3 have distance 0; all joints are located at the point of the incision – the tool's origin.

In order to accurately predict tool motion based on this spherical model, the system needs to perform some pre-processing to figure out the location of the spherical origin with respect to the camera frame, O^C. Since the surgeon on the da Vinci machine is able to move the camera, a matrix of locations of spherical origins will be calculated: $[O^C_1, O^C_2, \ldots, O^C_n]$ for n camera frames.

For the purposes of this model, we will assume that the teacher's and student's ports are placed in the same location, so their tools will have the same origins – standard locations for port placement can be used here. Thus, for any i, O^C_i for the teacher video is assumed equivalent to O^C_i for the student video.

Initial preprocessing is done with C++ and OpenCV. In it, color segmentation is used to identify tools, which are typically gray on a red- or pink-toned background. This color segmentation creates a binary image: pixels within the specified tool color range are labeled 1, while those outside the color range are 0. Some modifications such as dilation and closing of the binary image are performed to make the outlines clearer, then the findContours() function in OpenCV is used to identify the largest contours in the picture, those containing the tools. For each tool, a point on the left side of the tool's bounding box at the center of its height in that x-position and a point at the right side of the tool's bounding box at the center of its height in that y-position are found. Next, the procedure to convert a two-dimensional point in an image to a three-dimensional point, described later, is run on the left and right points for each tool. These three-dimensional points are printed to a csv file.

The movement of the camera frame is detected by the appearance of a bright blue icon on screen which the da Vinci Surgical System produces when the surgeon presses the pedal to move the camera. This icon is located by color segmenting; when it is found, the calculation of the origin point resets to compute an origin point with respect to the new camera frame.

MATLAB is then used for the next stage of the preprocessing. It reads in the csv file which contains the left and right points of the tools in each frame. For each individual tool, the slopes of the tools can be calculated from these left and right points, so the line of the tool is fully defined for each frame. Using these lines, the system calculates the point closest to the intersection of all of them (noise in the tool location makes it possible that they will not all intersect in the same place). This is done by implementing an algorithm developed by Han and Bancroft to find the point closest to the intersection of lines in n-dimensional space using singular value decomposition [25].

After all the origin points with respect to different camera frames are calculated, the preprocessing is complete and the video itself can be streamed.

3.2 Camera Calibration

Calibration is performed on the da Vinci Surgical System stereoscopic endoscope camera, using the built in StereoCalibrate program included in the OpenCV library [24]. Calibration was performed upon 31 pairs of stereo images by taking pictures of chessboards from the left and right camera frames of the endoscope at varying angles and distances.

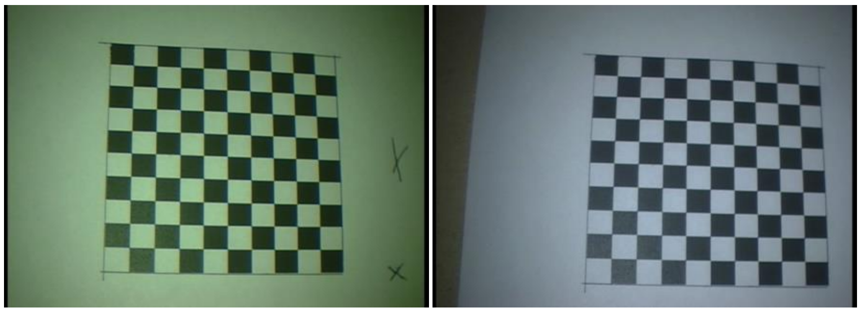


Figure 8a: Left image of chessboard; Figure 8b: Right image of chessboard

The findChessboardCorners() function was used to locate corners of this function, then stereo calibration was performed using these corners as corresponding landmarks in the image. The function stereoCalibrate() calculated the camera matrix for each individual camera: $\begin{bmatrix} fx & 0 & cx \\ 0 & fy & cy \\ 0 & 0 & 1 \end{bmatrix}$, where fx and fy are the x- and y- focal lengths for the camera, and cx and cy are the principal points at the image center, based on the locations of chessboard corners in corresponding images. The function stereoCalibrate() also calculated the distortion coefficients for each individual camera: [k1 k2 p1 p2 k3]. It also output R and T, the rotation and translation matrices between the two cameras.

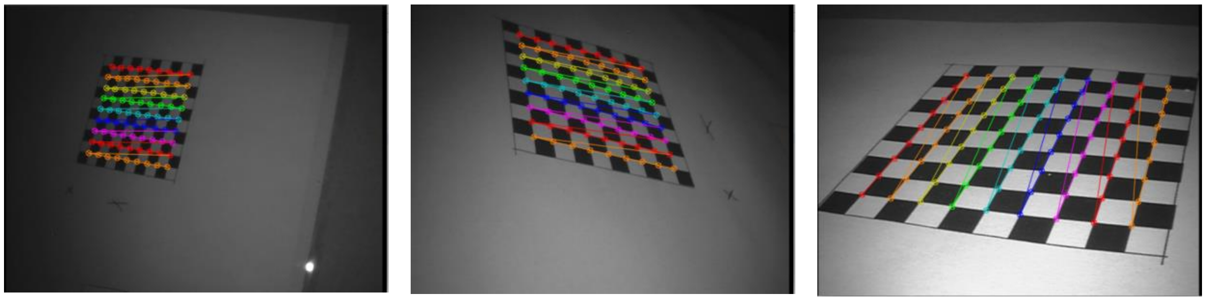


Figure 9: Chessboard corners located in image

Once the images were calibrated, they could be rectified. stereoRectify() was called to rectify the chessboard images, ensuring that corresponding points would be on the same horizontal line. This rectification produced chessboard images such as Figure 3 below: corresponding points in the chessboards are located on the same green lines; the images themselves are warped somewhat in order to achieve this effect. This function also output R1 and R2, the rectification transform rotation matrices for both cameras, as well as outputting P1 and P2, the projection matrices for the new coordinate systems for both cameras. It also output Q, the disparity-to-depth mapping matrix. initUndistortRectifyMatrix() and remap() were used to remap and draw images with correct rectification.

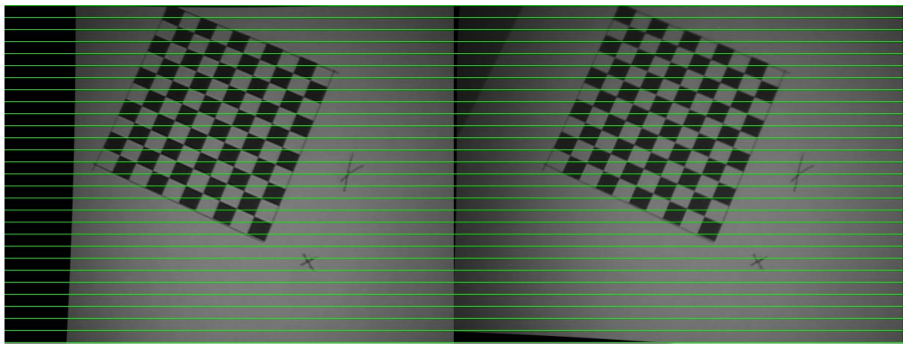


Figure 10: Rectified chessboard images

This calibration overall achieved a root mean square error of 0.842771, and an average epipolar error of 0.544317, meaning that it was quite accurate. The camera matrices, distortion coefficients, R, T, R1, R2, P1, P2, and Q were all written to a file, so that they could be used for later video rectification.

3.3 Runtime Processing

Before the video starts, the student is prompted to align their tools with the teachers' tools in the first frame of the video. Video from left and right teacher frames, as well as left and right student frames, is streamed to four OpenCV Mat images: studentLeft, studentRight, teacherLeft, and teacherRight.

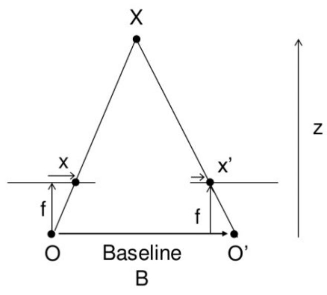
The images studentLeft and teacherLeft are added together into the image sumLeft, as well as studentRight and teacherRight into sumRight. The image sumLeft is displayed in the left eyepiece of the da Vinci surgeon console, and the image sumRight is displayed in the right eyepiece of the da Vinci surgeon console. In that way, the student is provided with augmented reality so that they can see clearly both their own tools and those of the surgeon they are mirroring. This provides a realistic environment, unlike other computer-generated environments which may appear overly smooth or neat.

Since they are taken from the da Vinci endoscope, these images are remapped. The function initUndistortRectifyMatrix() provided by OpenCV is used to calculate the remapping matrices for the left image and the right image needed to align corresponding points in left and right images to make sure that they are on the same horizontal line, as the chessboard images before were. It calculates them from the camera matrix, distortion coefficients, and rectification transform rotation matrix of their corresponding camera. Then, it remaps the images of da Vinci tools using these remapping matrices and the function remap(). This remapping causes corresponding points to have the same horizontal locations in their image, which will come in useful later when the disparity between corresponding points is calculated in order to deduce depth.

In the first frame, the ends of the tools are located by color segmentation, as the end and base of the student’s left tool are blue, and the end and base of the teacher’s left tool are green (they have been marked in this way). The teachers’ tools were not marked in this way. It would be possible to film video with teachers using these markers on their tools in a real surgery, as the markers are made of colored tape and would not interfere with the surgery in any way. However, the aim of this platform is to work with any pre-existing surgical video, not only videos which have been specially constructed for the occasion. Thus, we must assume that the student has correctly aligned their tools with the teachers’ tools in this first frame. Thus, the bounding boxes around the end of the students’ tools will also mark bounding boxes around the end of the teachers’ tools. These provide information on the ends and bases of the tool, so their endpoints and slopes can later be calculated.

On the first frame, a Kernelized Correlation Filter, provided by OpenCV as a KCFTracker, will be initialized on these starting bounding boxes in order to track these bounding boxes in the teachers’ frame, keeping track of the ends of the teachers’ tools throughout the application. Thus, after being initialized by a color segmentation in the student video, KCF Tracker was used across frames to keep track of the unmarked tools of the teacher video. KCF Tracker stores eight points (calculated from the centers of its bounding boxes of interest): in each of the student and teacher’s left and right videos, it stores the points of the left tool and the right tool.

Next, the StereoBM, or stereo block matching, function is used to find the disparity between corresponding points in the rectified images. Objects whose corresponding points in separate images are farther apart are typically closer to the cameras, while objects whose corresponding points in separate images are close together are farther from the camera. These can be expressed in the simplified equation:



$$disparity = x - x' = \frac{Bf}{Z}$$

Figure 11: Epipolar geometry

The disparity between corresponding points is calculated in pixels; then, the reprojectImageTo3D() function uses the Q reprojection matrix calculated in the calibration phase in order to turn a disparity map showing the distance between points into a depth map, expressing the X, Y, and Z locations of each pixel in the image with respect to one camera frame.

Next, the depth map is used to look up the X, Y, and Z locations of the endpoints of the student’s and teacher’s left and right tools. The teachers’ X, Y, and Z locations of the tools’ origins with respect to the camera frame were calculated during preprocessing. Specific tool origins with respect to the camera frame are known as a potentially different value for each frame, since the teacher can shift the camera during operation. Here, we will assume that the students and teachers have the same tool origins with respect to the camera frame. That is, the students are using the same basic port locations (common, established port locations are known and set up beforehand) and operating their camera in the correct way. Now, these three-dimensional positions students' and teachers' tool endpoint with respect to the camera frame are known; in addition, the X, Y, and Z locations of the students' and teachers' origins with respect to the camera frame are also known. Thus, the students' and teachers' tool endpoints and basepoints can be converted from the camera frame (xc, yc, zc) to the origin frame (xo, yo, zo).

Once the tool endpoints and basepoints can be expressed in three-dimensional space with respect to the origin frame, they can be converted into spherical coordinates as required for their spherical model. A simple conversion is performed from (xo, yo, zo) to (θ1, θ2, d3).

$$\theta 1 = \arctan(\frac{y0}{x0})$$

$$\theta 2 = \arctan\left(\frac{x0}{\sqrt{x0^2 + y0^2}}\right) + \pi/2$$

$$d3 = \sqrt{x0^2 + y0^2 + z0^2}$$

Then, a Kalman filter initialized in the first frame is run upon this set of spherical coordinates. Because the da Vinci arm is being modeled as a spherical robot, its motion is limited by certain constraints. We cannot necessarily assume that dx0/dt, dy0/dt, or dz0/dt are constant. However, we can assume that the rates of change of θ1, θ2, and d3 are constant, and we can filter out random noise from these measurements with a Kalman filter. A Kalman filter is thus initialized in three dimensions based on the spherical coordinates; it takes in the measured θ1, θ2, and d3 coordinates and outputs the theoretical, smoothed θ1, θ2, and d3 coordinates. Then, these coordinates are converted back into a Cartesian reference frame centered at the tool origins:

$$x1 = d3 * \cos(\theta 2) * \cos(\theta 1)$$

$$y1 = d3 * \cos(\theta 2) * \sin(\theta 1)$$

$$z1 = d3 * \sin(\theta 2)$$

This new point, (x1, y1, z1), is based on the smoothed Kalman filter of the spherical model. These points become the new endpoint and basepoint locations for the tool.

From the endpoint and basepoint locations, the slope of the tool can be calculated: $\langle x, y, z\rangle = (x_{end} - x_{base}, y_{end} - y_{base}, z_{end} - z_{base})$. Furthermore, the endpoint and basepoint's previously calculated spherical coordinates can be used to ascertain the accuracy of the model: if correct, they should have the same θ1 and θ2 coordinates and only differ by a constant amount in d3 each time.

In addition, the Cartesian coordinates of the last endpoint can be subtracted from the Cartesian coordinates of the current endpoint and added to a growing sum in order to approximate the path length traveled by each tool. On average, more experienced surgeons tend to have lower path lengths because they make clearer and more decisive movements, while less experienced surgeons are more subject to dithering, repetitive motion, and less economy of motion.

$$\text{Path Length} = \sum_{i=1}^{n} \sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2 + (z_i - z_{i-1})^2}$$

The distance between the student's endpoint and the teacher's endpoint, as well as the distance between the student's slope and the teacher's slope, can now be calculated for each tool in three dimensions. If the distance or the difference in slope is above a certain threshold, the program will pause on its current video frame before continuing in the while loop which continuously captures teacher and student video frames. The teacher's video will stay on its current frame, while the student video can move. The student will be able to observe their own motion, while the teacher's frame will remain still. While the student's endpoints or slopes are too disparate from the teacher's, the teacher's frame will remain still, giving the student a chance to catch up so the teacher video can start again.

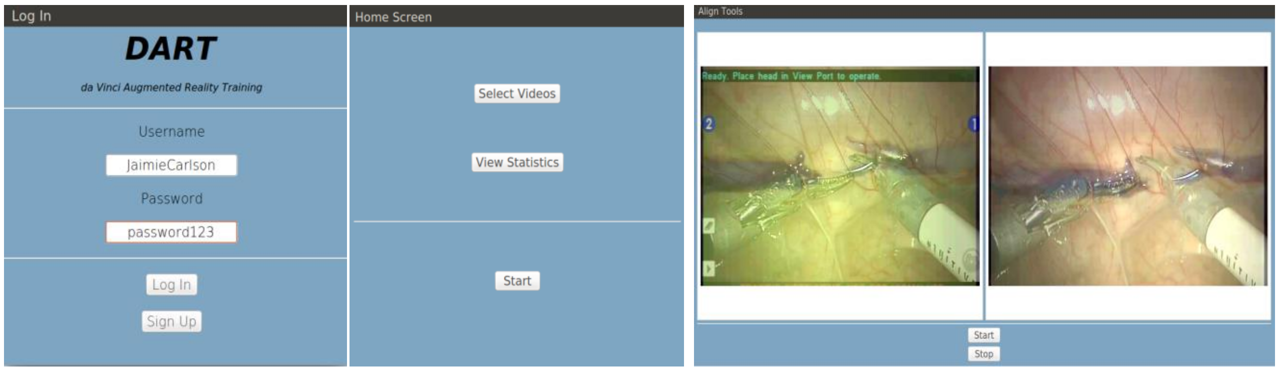
The start and stop times of this pausing are recorded with the QTTimeAndDate module; in addition, an incremental count is kept of the number of pauses, so that these can be recorded for user statistics. The difference between student and teacher endpoint positions, path length for each student and teacher tool, and difference between student and teacher slopes are also recorded for user statistics.

At the end of this process, the while loop begins over again, and new video frames from student and teacher left and right videos are read into the algorithm.

3.4 Workflow of Application

After opening the application, users are directed to the Login screen, in which they can sign up with a new username and password or log in with a pre-existing one. If they choose to sign up, their username and password will create a new database entry.

Once they log in, they are sent to the Home screen, which gives them three options: Select Videos, View Statistics, or Start. They must first choose Select Videos in order to choose the video for which they want to either view statistics or run the video. This brings up two windows to the file system for the user to choose left and right teacher videos, which should already be downloaded to the file system. Left and right video names should be formatted as "filename_Left.mpeg" and "filename_Right_Audio.mpeg," in order to ensure that the videos correspond to one another. If the filenames do not follow this format, an error screen will prompt the user to choose correct videos.



Username
JaimieCarlson
Password
password123

Figure 12a: Login screen; Figure 12b: Home screen; Figure 12c: Align screen

If viewers select Start on the Home screen, they will be directed to the Align screen. This screen features two graphics views – one for the left camera and one for the right camera. Each view displays a live feed of the student's video for its corresponding camera, superimposed onto the first frame of the teacher's video for that camera. The screen prompts students to align their tools with the tools of the camera, which they can visually ascertain by looking at the two graphics views. Once they are satisfied with the alignment, they can press the Start button on the Align screen.

This will cause two new screens to appear – one screen represents the left video feed and the other represents the right video feed. Each screen will appear in the corresponding da Vinci eyepiece. At this point, the algorithm described in the previous section will run in order to process the video. From the students' perspective, their video will appear in real time in each eyepiece, superimposed on top of the teachers' video in real time. As long as they are performing well by keeping their da Vinci arms in close proximity to the teachers' arms, the video will proceed in real time. However, if they move their arms too far away from the teachers' arms in the video, the teachers' video feed will pause, and a message will appear on screen instructing them to correct their tools' position or alignment. As soon as they successfully do so, the students' video feed will resume.

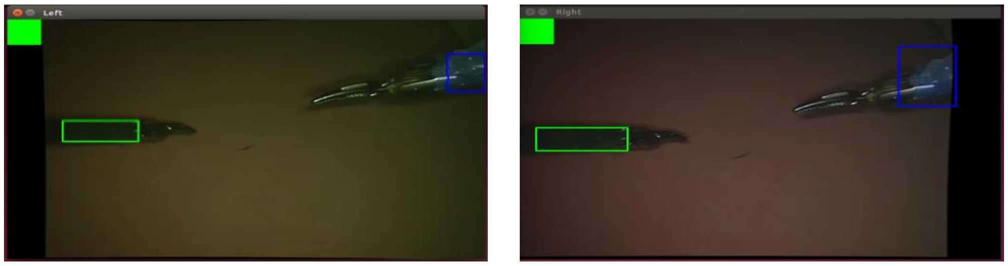


Figure 13: Left and Right Augmented Reality screens

Once the student wishes to stop, they (or a helper) can press Stop on the Align screen, which remains open. They will then be directed back to the Home screen. At this point, they can choose to press Start again to perform another surgery, choose View Statistics to see their performance on the surgery they have just completed, or choose Select Videos to play or see statistics on another video.

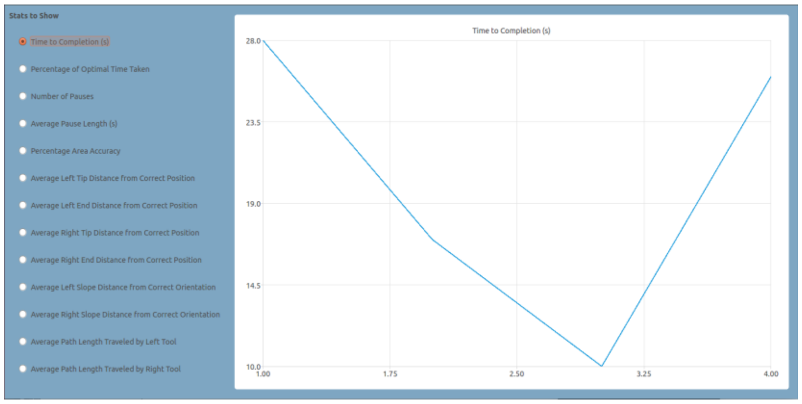

Stats to Show
Time to Completion (s)
Percentage of Optimal Time Taken
Number of Pauses
Average Pause Length (s)
Percentage Area Accuracy
Average Left Tip Distance from Correct Position
Average Left End Distance from Correct Position
Average Right Tip Distance from Correct Position
Average Right End Distance from Correct Position
Average Left Slope Distance from Correct Orientation
Average Right Slope Distance from Correct Orientation
Average Path Length Traveled by Left Tool
Average Path Length Traveled by Right Tool
Time to Completion (s)
28.0
23.5
19.0
14.5
10.0
1.00
1.75
2.50
3.25
4.00

Figure 14: Statistics screen

If they choose View Statistics, they can see statistics on the video which they have chosen. (Note: If no video has been selected yet, View Statistics will prompt the user to choose a video – a user cannot simply open the application and immediately view their statistics without choosing a video.) A numerical display of all statistics from their most recent surgery is shown at the bottom of the screen, and line graphs illustrative of progress over time can be seen for specific statistics. A series of radio buttons allows the user to select the statistic for which they wish to see a line graph: time to completion, percentage of optimal time taken, number of pauses, average pause length, average left tip distance from correct position, average right tip distance from correct position, average left slope distance from correct orientation, average right slope distance from correct orientation, average path length traveled by left tool, and average path length traveled by right tool.

This application's user interface easily allows students to select any videos which they wish to rehearse, perform surgeries using those videos, and view immediate feedback on recent progress as well as information about their progress over time.

4. DISCUSSION

The concept of this application represents an individualized improvement in the field of robotic surgery simulation. Since experienced surgeons have very different motion patterns than novices, much training on the da Vinci machine focuses on encouraging novice surgeons to acquire the motion habits of the expert surgeons, such as efficient movements and a lack of dithering. To that end, medical students often watch videos of experts performing robotic surgery. This approach combines that experience with more hands-on practice at the da Vinci, allowing students not only to practice a procedure, but to practice a procedure as an expert would perform it. One surgeon, stating his approval of the concept of this application, noted that it would especially benefit younger surgeons, as they are more likely to have experience in performing laparoscopies rather than open surgeries, whose techniques can actually prepare individuals better for using the da Vinci. Another surgeon concurred that this application would provide a useful way for medical students to practice work on minimally invasive robotic surgery.

This application utilizes color segmentation in a variety of ways. Color segmentation is used to pick out blue and green markers on the student tools. A conversion of the image from an RGB (red-green-blue) to HSV (hue-saturation-value) representation ensures that the markers can be recognized under a variety of lighting conditions found in the body. More challengingly, it is used to pick out entire tools during pre-processing of the teacher video, which is not provided with any colored markers beforehand. Its success is fairly robust to different da Vinci tools; color sliders are provided so that the user can tweak color segmentation in order to recognize the specific tools they are using. In addition, the success of color segmentation here depends largely on size. The tools are grey, black, or white, shades which are usually not found in the body, making color segmentation a good technique for preprocessing the tools. However, false positives may be created due to areas of deoxygenated or necrotic tissue in the body, which also appear grey; in addition, surgeons often place the two tools very near to or touching one another, causing the application to only register one very large contour instead of two separate ones. The size constraints for the tool contours typically prevent such small areas of tissue from being recognized as tools, and during preprocessing, the contours found were filtered by width to determine if they represented one tool or two tools touchcing one another. These difficulties were thus manageable during preprocessing, but they caused color segmentation to only be used for the teacher video in preprocessing, not the program's runtime stage.

Kernelized correlation filter was used for tracking the teacher's tools during runtime. It was chosen out of a variety of trackers built into OpenCV. The KCF Tracker has several advantages over other tracking algorithms offered by OpenCV, such as Multiple

Instance Learning (MIL Tracker), BOOST (BOOST Tracker), Median Flow (MF Tracker), TLD (Tracker Learner Detector), meanshift, and camshift. In a comparison of these trackers' ability to track da Vinci tools in video, the KCF Tracker worked best on both large and small regions of interest. The KCF Tracker also was often able to find a tool again after it temporarily left the frame. As expert surgeons often operate with their camera zoomed very far in, this ability of the tracker makes it uniquely suited for computer vision in surgical video: da Vinci tools often leave the screen and come back. The KCF Tracker is able to handle this case without either swelling up to encompass the whole screen, like camshift, being unable to handle the camera zooming in or out and drastically changing tool size, like meanshift, or becoming "stuck" on non-tool objects when small areas are selected, like the other tracking algorithms.

The StereoBM algorithm offered by OpenCV was not perfect at calculating the disparity. When the disparity map was converted to a depth map from (x, y) pixel coordinates to (x, y, z) coordinates in millimeters, the depths of the tools in certain images yielded approximately accurate responses. However, the disparity map itself was quite noisy, and required a large "minimum disparity" value to produce reasonable values, which severely limited the scope of the image. The OpenCV StereoSGBM, or Semi Global Block Matching, was tried as well and did not produce better results. According to one study, the da Vinci S endoscope has a divergent optical axis as well as "fisheye" radial distortion which makes the disparity between two pixels a function of not only z-position in the world frame (depth) but also x- and y-position as well. Using points at known depths, this study manually calculated the disparity as a function of x, y, and z location, and used that to calculate a more accurate depth; this technique may be tried in future [26]. Obviously, this would not be practical for garnering live operation video. However, future work will focus on improving the depth map's accuracy so that the Kalman filter can receive less noise to filter out of the data.

5. CONCLUSION

In conclusion, this project presents the first stage of an interactive augmented-reality da Vinci training simulator. It offers the marked advantage over other existing simulators of using real video filmed by an expert while operating on the da Vinci itself. The preprocessing algorithm, runtime algorithm and user interface supporting this product have been developed to successfully locate tools within video of previous da Vinci surgeries. Once successfully connected to the da Vinci surgical system via hardware, this system will be able to provide a cheap and easy way for medical students who have already honed basic skills on the da Vinci to practice a variety of more elaborate surgeries.

6. RECOMMENDATIONS

Future work for this project remains to be done in several areas.

First, although it is possible to stream in video from the da Vinci endoscope, it is currently only possible to do so from one eyepiece. Several S-video to USB converters used for the purpose of extracting video from the da Vinci have proven incompatible with the current Linux distribution and drivers. Future work includes creating and applying a patch to make the converters functional so that video can be streamed in from both eyes of the da Vinci. This will allow for the creation of more testing video as well as the finalization of the complete application itself.

Second, the correct outputting of video streams to the da Vinci eyepiece at the surgeon’s console has yet to be accomplished. Although newer da Vinci systems have a plug-and-play TilePro system which allows video to be easily streamed into the da Vinci, the first generation da Vinci machine lacks this functionality. Accordingly, streaming of video to the da Vinci machine has not yet been accomplished.

Different generations of da Vinci machines –first generation, the S, the Si, and the Xi – all have slightly different cameras. Accordingly, this program can only use video from the da Vinci machine on which it was calibrated – calibration which is required for the creation of the depth map calculates the rotation and translation matrices between the two cameras of the endoscope. These matrices differ from camera to camera, however. Since the first generation da Vinci is no longer used in surgery, a next step in this project would be to gain access to, calibrate the application on, and test students on a more modern da Vinci model. In that way, actual surgery videos could be compared to the user’s movements.

After the completion of these first two tasks, pilot testing, and then a proper user study, can be performed on this application, so that the third task – bringing the application to be used on a more modern da Vinci machine – can be achieved.

7. ACKNOWLEDGMENTS

The author thanks Dr. Brown and Dr. Kuchenbecker for their amazing mentorship; the Haptics GRASP lab for aiding this project; the Rachleff Scholars Program for providing a grant to support this research; Dr. Daniilidis for his advice during this project; and Dr. Murphy and Dr. Dumon for providing a surgical perspective.

8. REFERENCES

[1] A. R. Lanfranco, A. E. Castellanos, J. P. Desai, and W. C. Meyers, “Robotic Surgery,” *Ann. Surg.*, vol. 239, no. 1, pp. 14–21, 2004.

[2] A. Amodeo, A. Linares Quevedo, J. V. Joseph, E. Belgrano, and H. R. Patel, "Robotic laparoscopic surgery: cost and training," *Minerva Urol. Nefrol.*, vol. 61, no. 2, pp. 121-8, 2009.

[3] S. Maeso, M. Reza, J. A. Mayol, J. A. Blasco, M. Guerra, E. Andradas, and M. N. Plana, "Efficacy of the Da Vinci surgical system in abdominal surgery compared with that of laparoscopy: a systematic review and meta-analysis," *Ann. Surg.*, vol. 252, no. 2, pp. 254-62, 2010.

[4] M. Reza, S. Maeso, J. A. Blasco, and E. Andradas, “Meta-analysis of observational studies on the safety and effectiveness of robotic gynaecological surgery,” *Br. J. Surg.*, vol. 97, no. 12, pp. 1772–1783, 2010.

[5] Y. Yang, F. Wang, P. Zhang, C. Shi, Y. Zou, H. Qin, and Y. Ma, “Robot-assisted versus conventional laparoscopic surgery for colorectal disease, focusing on rectal cancer: a meta-analysis.,” *Ann. Surg. Oncol.*, vol. 19, no. 12, pp. 3727–3736, 2012.

[6] S. Lin, H.-G. Jiang, Z.-H. Chen, S.-Y. Zhou, X.-S. Liu, and J.-R. Yu, “Meta-

analysis of robotic and laparoscopic surgery for treatment of rectal cancer.,” *World J. Gastroenterol.*, vol. 17, no. 47, pp. 5214–20, 2011.

[7] J. K. Koehn and K. J. Kuchenbecker, “Surgeons and non-surgeons prefer haptic feedback of instrument vibrations during robotic surgery,” *Surg. Endosc. Other Interv. Tech.*, vol. 29, no. 10, pp. 2970–2983, 2015.

[8] Y. Sumi, P. W. Dhumane, K. Komeda, B. Dallemagne, D. Kuroda, and J. Marescaux, “Learning curves in expert and non-expert laparoscopic surgeons for robotic suturing with the da Vinci?? Surgical System,” *J. Robot. Surg.*, vol. 7, no. 1, pp. 29–34, 2013.

[9] L. Chang, R. M. Satava, C. A. Pellegrini, and M. N. Sinanan, “Robotic surgery: Identifying the learning curve through objective measurement of skill,” *Surg. Endosc. Other Interv. Tech.*, vol. 17, no. 11, pp. 1744–1748, 2003.

[10] P. Yohannes, P. Rotariu, P. Pinto, A. D. Smith, and B. R. Lee, “Comparison of robotic versus laparoscopic skills: Is there a difference in the learning curve?,” *Urology*, vol. 60, no. 1, pp. 39–45, 2002.

[11] M. B. Bokhari, C. B. Patel, D. I. Ramos-Valadez, M. Ragupathi, and E. M. Haas, “Learning curve for robotic-assisted laparoscopic colorectal surgery.,” *Surg. Endosc.*, vol. 25, no. 3, pp. 855–60, 2011.

[12] J. P. Lenihan, C. Kovanda, and U. Seshadri-Kreaden, "What is the learning curve for robotic-assisted gynecologic surgery?", *J. Minim. Invasive Gynecol.*, vol. 15, no. 5, pp. 589-94, 2008.

[13] F. Atug, E. P. Castle, S. K. Srivastav, S. V. Burgess, R. Thomas, and R. Davis, “Positive Surgical Margins in Robotic-Assisted Radical Prostatectomy: Impact of Learning Curve on Oncologic Outcomes,” *Eur. Urol.*, vol. 49, no. 5, pp. 866–872, 2006.

[14] V. R. Patel, A. S. Tully, R. Holmes, and J. Lindsay, “Robotic radical prostatectomy in the community setting — The learning curve and beyond : Initial 200 cases,” vol. 174, no. July, pp. 269–272, 2005.

[15] N. Doumerc, C. Yuen, R. Savdie, M. B. Rahman, K. K. Rasiah, R. Pe Benito, W. Delprado, J. Matthews, A. M. Haynes, and P. D. Stricker, “Should experienced open prostatic surgeons convert to robotic surgery? The real learning curve for one surgeon over 3 years,” *BJU Int.*, vol. 106, no. 3, pp. 378–384, 2010.

[16] “Intuitive Surgical - da Vinci Si Surgical System - Skills Simulator.” [Online]. Available: http://www.intuitivesurgical.com/products/skills_simulator/.

[17] “Mimic Simulation | dV-Trainer.” [Online]. Available: http://www.mimicsimulation.com/products/dv-trainer/.

[18] “What is RoSS_ Simulated Surgical Systems.” [Online]. Available: http://www.simulatedsurgicals.com/ross2.html

[19] A. Moglia, V. Ferrari, L. Morelli, M. Ferrari, F. Mosca, and A. Cuschieri, “A Systematic Review of Virtual Reality Simulators for Robot-assisted Surgery,” *Eur. Urol.*, vol. 69, no. 6, pp. 1065–1080, 2015.

[20] “Simulated Surgical Systems News.” [Online]. Available: http://www.simulatedsurgicals.com/index.html.

[21] "da Vinci Skills Simulator." [Online]. Available: http://intuitivesurgical.com/products/skills_simulator/

[22] A. Chowriappa, S. J. Raza, A. Fazili, E. Field, C. Malito, D. Samarasekera, Y. Shi,

K. Ahmed, G. Wilding, J. Kaouk, D. D. Eun, A. Ghazi, J. O. Peabody, T. Kesavadas, J. L. Mohler, and K. A. Guru, “Augmented-reality-based skills training for robot-assisted urethrovesical anastomosis: A multi-institutional randomised controlled trial,” *BJU Int.*, vol. 115, no. 2, pp. 336–345, 2015.

[23] A. Breda and A. Territo, “Virtual Reality Simulators for Robot-assisted Surgery,” *Eur. Urol.*, vol. 69, no. 6, pp. 1081–1082, 2015.

[24] G. Bradski and A. Kaehler, *Projection and 3D Vision*. 2008.

[25] L. Han and J. C. Bancroft, “Nearest approaches to multiple lines in n -dimensional space,” vol. 22, pp. 1–17, 2010.

[26] R. Dockter, R. Sweet, and T. Kowalewski, “A fast, low-cost, computer vision approach for tracking surgical tools,” *IEEE Int. Conf. Intell. Robot. Syst.*, pp. 1984–1989, 2014.