robotics

MDPI

Article

A Natural Language Interface for an Autonomous Camera Control System on the da Vinci Surgical Robot

Maysara Elazzazi [1], Luay Jawad [2], Mohammed Hilfi [1] and Abhilash Pandya [1,*]

[1] Department of Electrical and Computer Engineering, Wayne State University, Detroit, MI 48202, USA; melazzazi@wayne.edu (M.E.); mhilfi@wayne.edu (M.H.)
[2] Department of Computer Science, Wayne State University, Detroit, MI 48202, USA; ljawad@wayne.edu
* Correspondence: apandya@ece.eng.wayne.edu

Abstract: Positioning a camera during laparoscopic and robotic procedures is challenging and essential for successful operations. During surgery, if the camera view is not optimal, surgery becomes more complex and potentially error-prone. To address this need, we have developed a voice interface to an autonomous camera system that can trigger behavioral changes and be more of a partner to the surgeon. Similarly to a human operator, the camera can take cues from the surgeon to help create optimized surgical camera views. It has the advantage of nominal behavior that is helpful in most general cases and has a natural language interface that makes it dynamically customizable and on-demand. It permits the control of a camera with a higher level of abstraction. This paper shows the implementation details and usability of a voice-activated autonomous camera system. A voice activation test on a limited set of practiced key phrases was performed using both online and offline voice recognition systems. The results show an on-average greater than 94% recognition accuracy for the online system and 86% accuracy for the offline system. However, the response time of the online system was greater than 1.5 s, whereas the local system was 0.6 s. This work is a step towards cooperative surgical robots that will effectively partner with human operators to enable more robust surgeries. A video link of the system in operation is provided in this paper.

Keywords: da Vinci; robotic surgery; laparoscopic surgery; autonomous camera control; natural language processing

check for updates

Citation: Elazzazi, M.; Jawad, L.; Hilfi, M.; Pandya, A. A Natural Language Interface for an Autonomous Camera Control System on the da Vinci Surgical Robot. *Robotics* **2022**, *11*, 40. https://doi.org/10.3390/robotics11020040

Academic Editors: Mario Selvaggio, Sara Moccia and Bruno Scaglioni

Received: 9 February 2022
Accepted: 22 March 2022
Published: 25 March 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Over 20 years, more than 25 K publications relating to robotic surgical systems were peer reviewed with clinical and engineering-based research into robotic surgery [1]. With the integration of robotics in surgery, many robotic surgical procedures have been safely and successfully completed. However, the clinical systems are still master–slave controllers with minimal (if any) autonomous behaviors. One area where automation could make a substantial difference is in camera viewpoint automation [2]. Maintaining an optimal view of the surgical scene is fundamental to surgery success.

Positioning a camera during laparoscopic procedures is challenging. During surgery, if the camera view is not optimal, surgery becomes more complex and potentially error-prone. The camera operator must try to predict the surgeon's needs, and the surgeon must operate safely and effectively despite any potential undesirable movements by the camera operator. This is no longer the case in fully robotic surgeries, as the surgeon is responsible for the camera's movement. However, this introduces a new problem wherein the surgeon must stop operating to move the camera. The distracting shift in focus can lead to accepting suboptimal views, longer surgery times, and potentially dangerous situations such as having tools out of the view. Therefore, automatic camera positioning systems that solve some of these problems have been developed and could be used to a significant extent in both traditional laparoscopic and fully robotic surgeries. However, there are times when

the surgeon's strategies change with different stages of surgery, and these changes can be unpredictable.

A critical barrier to overcome in camera positioning during surgery is that it is difficult to precisely articulate the ideal camera placement. There is a lack of documentation on how a camera operator should move a camera during laparoscopic procedures or how a camera should be placed for proper views during robotic procedures. We have interviewed expert robotic surgeons about camera placement. While they can describe context-specific rules of thumb, they cannot provide general principles from which an autonomous system can be derived [3]. Indeed, to quote a surgeon: "When it's hard for me to communicate what I want to see, then I just take over the camera." An autonomous system for camera placement in robotic surgery will similarly need to take direction from the surgeon.

To address this need, we have developed a voice interface to our existing autonomous camera system that can trigger behavioral changes and be more of a partner to the surgeon. Similarly to a human operator, it can take cues from the surgeon to help create optimized surgical camera views. It has the advantage of nominal behavior that is helpful in most general cases and has a natural language interface that makes it dynamically customizable and on-demand. It permits the control of a camera with a higher level of abstraction.

We introduce the utilization of Natural Language Processing (NLP) as an interface for an autonomous camera system (Figure 1). By introducing this interface, we allow surgeons to utilize preference-driven camera control algorithms. Voice interfacing can create an environment where the surgeon can access the algorithm's parameters. This feature enables the surgeon to adjust parameters to fit the current surgical situation or personal preference.

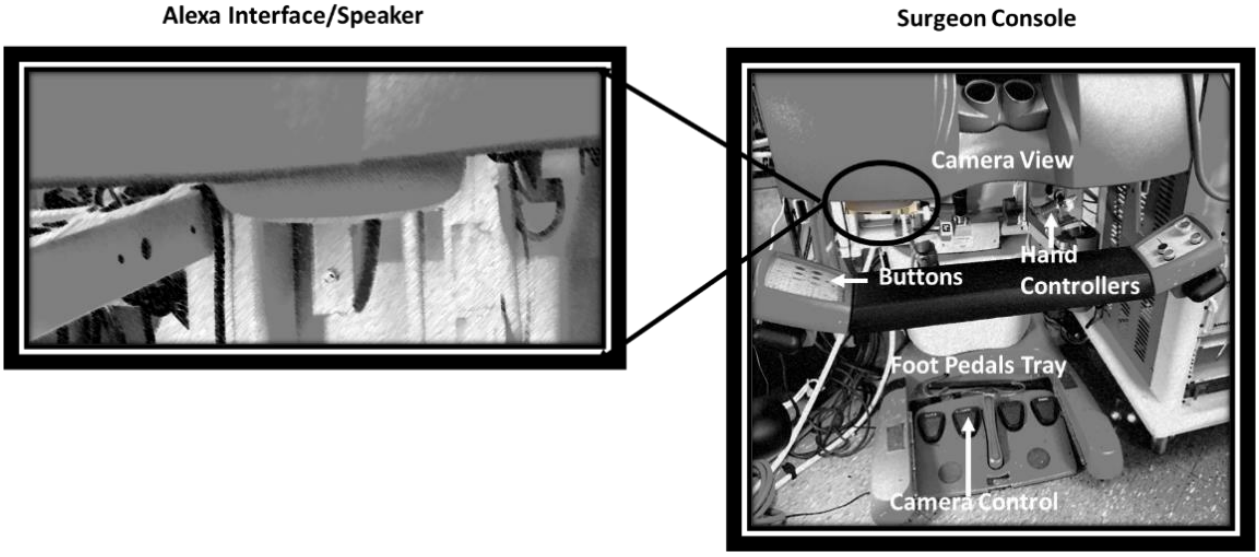


Figure 1. Overview of the da Vinci Surgical System and the Natural Language Processing (NLP) integration hardware. Our system uses a voice interface. The traditional interfaces are buttons and foot pedals to control the da Vinci system. The Alexa echo-dot system (with built-in microphone and speaker) is mounted near the user.

2. Background

Historically, surgical robotic platforms such as the da Vinci have utilized foot pedals, hardware buttons, and touchpads for menu navigation and robot operation. Research has shown that providing more direct human–robot interaction methods can decrease surgical time. For instance, Staub et al. utilized a gesture-based input method for directly accessing robotic commands without navigating a menu. This method of operation took significantly less time to command the robot [4].

Similarly, voice recognition and Natural Language Processing technologies have also been introduced into the medical field, from document creation and analysis to robot

control [5–12]. In using this technology within the operating room (OR), we can see its effect in preventing the need for extra surgical staff and the ability for the surgeons to interact with surgical equipment directly [6]. These methods have also been applied to surgical robotics by using voice-controlled endoscopic manipulators.

One of the first uses of voice-controlled robotics in the OR was AESOP; a seven degree-of-freedom arm used to maneuver a laparoscopic surgery camera [5,7,9,11,13]. This voice-controlled robot enabled surgeons to utilize either joystick or voice control as needed. In practicing on cadavers, it becomes clear that there are some situations where joystick control is necessary and others where voice control allows for the greatest flexibility. One particular note during this study was the impact of unrecognized spoken commands on time and safety, particularly in attempting to stop the voice recognition mode [11]. In addition, AESOP was controlled with very low-level commands such as "Move Left" and "Move In". This robot and associated technology did not merge into mainstream surgical robotics. We conjecture that, to be a helpful tool, a higher level of abstraction of commands is needed. For instance, as we have developed here, commands such as "Follow my Right tool", etc., would potentially make it easier to adopt.

The current state-of-the-art automated camera control involves visual servoing and different tool tracking/prediction algorithms [14–16]. Several autonomous camera systems have been created for the specific application of minimally invasive surgery [17–21]. For tracking, most of these systems used image processing or robot kinematics to identify the position of the tooltips relative to the camera. The methods generally use a limited number of rules to set the camera's target position and zoom level to move the camera. For instance, we have implemented a set of rules on our da Vinci platform that positions the camera to point to the midpoint of two tracked tooltips and alters the zoom level as necessary to keep them in the camera's view [18,19]. Briefly, this autonomous camera algorithm maintains the field of view around the tools so that the surgeon does not have to stop working to press the clutch and move the camera, and then continue working again. The algorithm utilizes the kinematic properties of the Patient Side Manipulator (PSM) to generate a midpoint between the left and right PSMs. Inner and outer zones govern the mechanical zoom and the field of view. Although this system outperforms an expert camera controller with essential metrics such as keeping the tool in the field of view, the expert camera operator still resulted in faster execution times [19].

The design of our current system is an extension of our previous da Vinci work [19]. It was also strongly influenced by extensive interviews we performed with eight laparoscopic surgeons on camera control during 11 surgical subtasks (suturing, dissection, clip application, etc.) [3]. Some of the key findings were that surgeons often prefer to teach by demonstration and that different subtasks had different requirements (highlighting the necessity of context/subtask awareness). We also obtained important information from observing numerous minimally invasive surgeries [22]. For example, we have observed cases where the surgeon had to move the camera nearly 100 times in an hour.

Moreover, to avoid further camera work, the surgeons sometimes let one or more instruments leave the camera's view. By examining our interviews and interactions with surgeons, we have also learned that the surgeon must maintain a view of the surgery through the video screen for as long as possible without removing their head from the console. In addition, the surgeon must be capable of operating the algorithms to their preference. The developed system is designed to minimize the surgeon's workload and to address these situations and suggestions.

Recent advancements in artificial intelligence, voice recognition, and NLP have facilitated a much more intelligent, natural, and accurate speech recognition experience. Several interfaces and open-source projects are available today that simplify the integration of well-developed and well-trained neural networks for speech recognition. The popular Alexa interface is a good and ubiquitous example of this. We chose Alexa as our online interface due to the ease of integration for the proof-of-concept presented here. Furthermore, we also utilized, tested, and compared Alexa to Vosk [23], a system that does not compromise

patient security and executes locally, which is preferable for actual implementation in an OR.

3. Materials and Methods

This section shows the implementation details of several valuable extensions of our baseline autonomous camera algorithm and their natural-language integration. Essentially, a voice interface relative to the da Vinci system will allow the natural control of the parameters of the autocamera algorithm. For instance, the inner and outer zones (used to control the zoom level) can be configured to allow direct zoom control during specific subprocedures.

This section will first describe our da Vinci robot and the test platform. It will then explain the Alexa interface required for natural language processing. Lastly, each of the seven commands extending the baseline algorithms is detailed. These commands include the following:

- "Start/Stop the autocamera"—toggles whether the endoscope should automatically follow the trajectory of the tools;
- "Find my tools"—finds the surgeon's tools when out of the field of view;
- "Track left/middle/right"—has the autocamera algorithm follow the right, middle, or left tool;
- "Keep left/middle/right"—maintains a point in space specified by the right, middle, or left tool position within the field of view;
- "Change inner/outer zoom level"—changes the settings associated with zoom control.

After each command has been recognized, the system responds back with either "Done" or a beep indicating that the action was triggered.

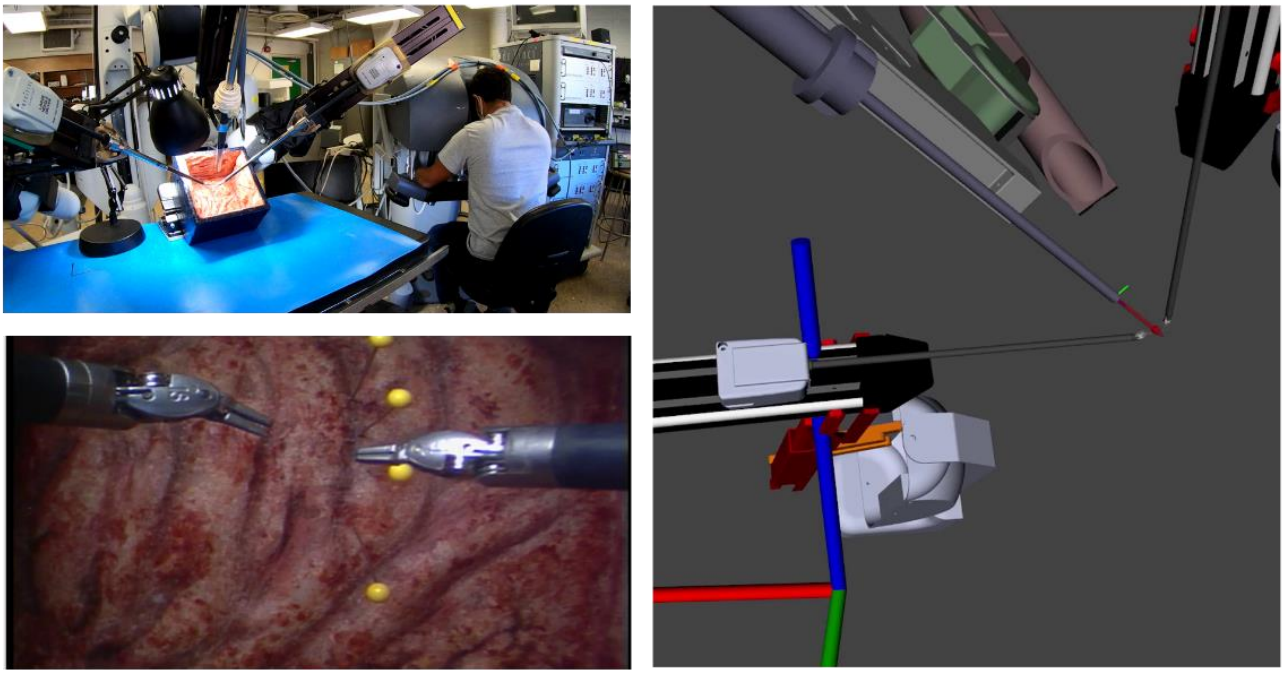
3.1. The da Vinci Standard Surgical System and Kit

Our research laboratory has a da Vinci Standard Surgical System modified to operate with the da Vinci Research Kit (dVRK) [24]. As shown in Figure 2, it uses open-source software and hardware control boxes to command and read feedback from the robotic system. This equipment, combined with the Robot Operating System (ROS) software framework [25], is used for this research study. We also have a software simulation of our da Vinci test platform used for algorithm prototyping and the playback/visualization of the recorded data [26].

3.2. Software Interface

Two voice assistants were integrated for testing and comparison. The voice assistant applications are built with the ROS middleware for direct access to dVRK state information and control capabilities. Both implementations were tested on a 64-bit Ubuntu 18.04 machine with an Intel i7-3770k CPU with 16 GB RAM. We describe both system implementations next.

The total system architecture was developed with modularity in mind to enable a wide variety of surgical assistants (user interfaces, voice assistants, and autonomous assistants). Future implementations can easily replace the voice assistant node by using the same assistant bridge interface. After the voice request is made and the correct function in the Voice Assistant is triggered, data captured in ROS messages with information from the voice assistant are communicated to the assistant bridge. The bridge handles surgeon requests by directly interacting with the da Vinci console or editing the desired software settings and parameters.



(**a**) (**b**)

Figure 2. Overview of our da Vinci surgical system setup. (**a**) Our da Vinci Surgical System is a test platform for algorithm implementation (**top**) and subsequent operator view through the endoscope (**bottom**). (**b**) Software simulation of the da Vinci test platform is used for algorithm prototyping and data playback/visualization. The simulated robot closely matches the real one, allowing rapid development and testing to be performed first in simulation.

3.2.1. Online NLP Interface (Alexa)

The first application is based on Alexa, Amazon's cloud-based voice service for Natural Language Processing. Amazon provides a well-documented and advanced toolset for creating "Skills" to integrate with their services [27]. Skills allow the creation of a set of phrases (intents) that can contain sets of variables (slots). For testing purposes, we also opened a secure tunnel to our localhost using ngrok [28]. The ngrok tool allowed us to field intents from the Amazon web server for hardware interaction. The backend connection to the Amazon Skill was developed in Python using the open-source package flask-ask. Commands are spoken to Alexa and registered by the skill; then, data from the request are forwarded via ngrok to the local flask-ask and ROS Python applications for handling (Figure 3).

3.2.2. Offline NLP Interface (Vosks)

The second voice assistant implemented is an offline implementation of speech recognition. This application relies on Vosk, an open-source program based on the Kaldi toolkit for speech recognition [23,29]. Vosk's architecture is similar but is processed locally and does not require an online server or internet connection. We used a USB connected microphone (ReSpeaker Mic Array v2.0 (Seeed Studios Inc., Shenzhen, China)) for testing.

Models for speech recognition are provided by Vosk, which contain the language model, the acoustic model, and the phonetic dictionary used in the recognition graph. Our implementation consists of an adapted language model that includes only the grammar spoken in the subset of commands utilized. This limited grammar set increases speed and accuracy and prevents the possibility of recognizing unintended commands. As with the Alexa implementation shown in Figure 3, the architecture for this system remains the same with the exception that the voice is processed and handled within the local host and voice module alone, thus eliminating the need for any cloud or online server.

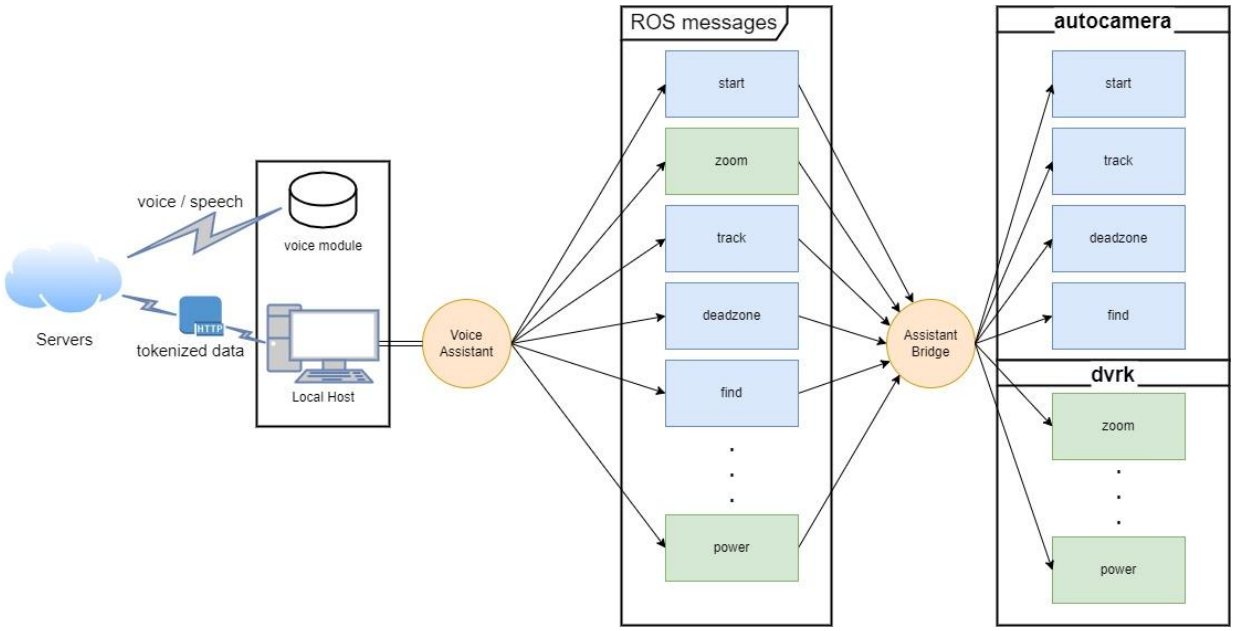


Figure 3. Architecture diagram for the Alexa voice assistant showing the local setup with the voice module communicating intents to the server and responses coming back as tokenized data through the secure tunnel. Moreover, the voice assistant's abstraction from interaction with the hardware and software algorithms is shown. Orange circles (Voice Assistant and Assistant Bridge) are ROS nodes we created for interaction between voice and hardware.

3.3. Voice Interface Implementation

3.3.1. Creating an On-Demand Autocamera System

The "start" and "stop" autocamera commands provide the surgeon the ability, when desired, to start or stop the autocamera software. Start and stop is communicated via a ROS topic through the assistant bridge and tells the autocamera algorithm when to publish joint commands to the Endoscopic Camera Manipulator (ECM) to follow the midpoint between the Patient Side Manipulator (PSM) tools. Shown in Algorithm 1, setting run to false will prevent the commands from being published and keep the ECM only in the position it was moved to by the operator or the final position before receiving the stop command.

Algorithm 1 Start/Stop.

1. function autocamera_algorithm(*run*)
   Input: Single boolean value (*run*) indicating run state of the autocamera
   Output: *None*
2. **While** *run = True*
3. | *ecm_desired*← compute_view_angle(*joint_angles, cam_info*)
4. | *ecm_desired*← find_zoom_level(*joint_angles, cam_info, ecm_desired*)
5. | move_joint(*ecm_desired*)
6. **end**

3.3.2. Find My Tools

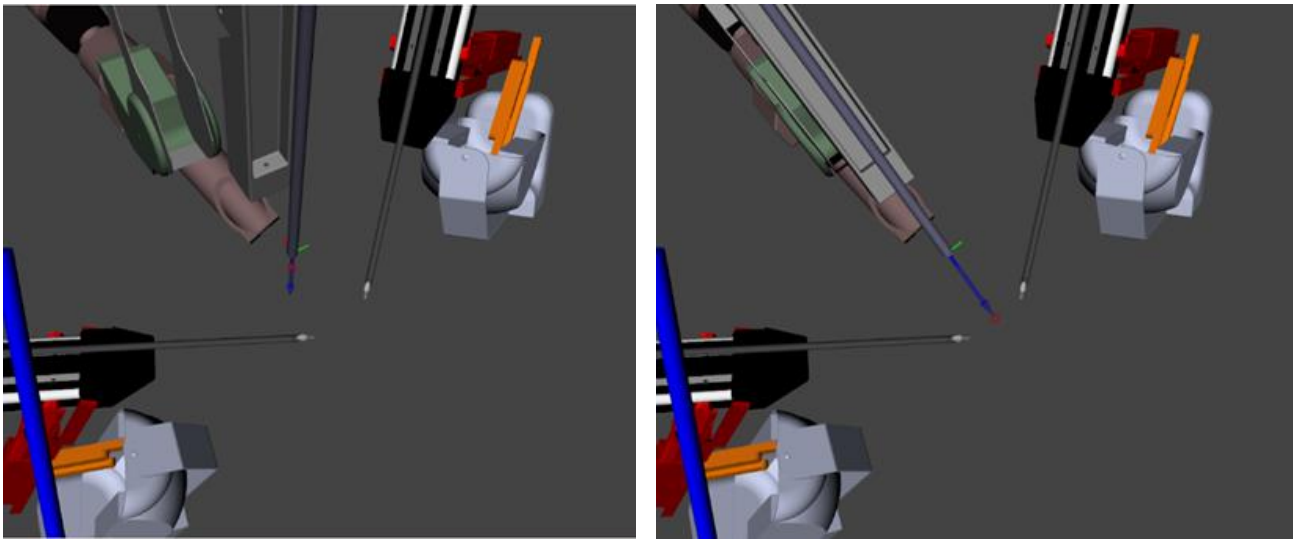
Find my tools is a command that directs da Vinci to place the surgeon's tools back into the camera's center field of view. It allows the surgeon to work without the full capability of the autocamera and allows the surgeon to quickly find the tools. The implementation shown in Algorithm 2, is similar to that of the autocamera algorithm. First, the joint values are used in the function to find the location of the two PSMs. The 3D coordinates are averaged to find the middle of the two tools. A rotation matrix is then calculated to provide the rotation between the current endoscopic manipulator position and the midpoint

location of the two tools. The rotation matrix is then multiplied by the current endoscopic orientation to provide the desired look-at direction. The inverse kinematics are computed to provide the joint angles required to position the endoscope. The zoom level is adjusted to bring the tools within the field of view.

Algorithm 2 Find Tools.

1. function find_tools(*joint_values*, *cam_info*)
   Input: arm joint angles *(joint_values)* and camera projection matrix (*cam_info*)
   Output: *None*
2. *psm1_pos* ← forward_kinematics(*joint_values ("psm1")*)
3. *psm2_pos* ← forward_kinematics(*joint_values ("psm2")*)
4. *midpoint* ← *psm1_pos* + *psm2_pos* / 2
5. *rot* ← ecm_to_midpoint_rotation(*joint_values("ecm")*, *midpoint*)
6. *ecm_desired* ← inverse_kinematics(*rot*ecm_current*)
7. move_joint(*ecm_desired*)
8. autocamera_algorithm*(False)*
9. **end**

Figure 4 shows the tested implementation method of find my tools in the Rviz simulation software. The blue arrow is an indication of the current Endoscopic orientation. The red dot is the calculated midpoint of the two PSM tools. After commanding find my tools, ECM is positioned at an angle that places the tools back in the field of view of the operator.



(**a**) (**b**)

Figure 4. This figure shows how the camera view is altered to place the tools in the field of view. (**a**) Is the orientation of ECM when the tools would be out of the field of view. (**b**) The orientation of the ECM after the find my tools voice command has been given, and the tools are placed back into the field of view.

3.3.3. Track Left/Middle/Right

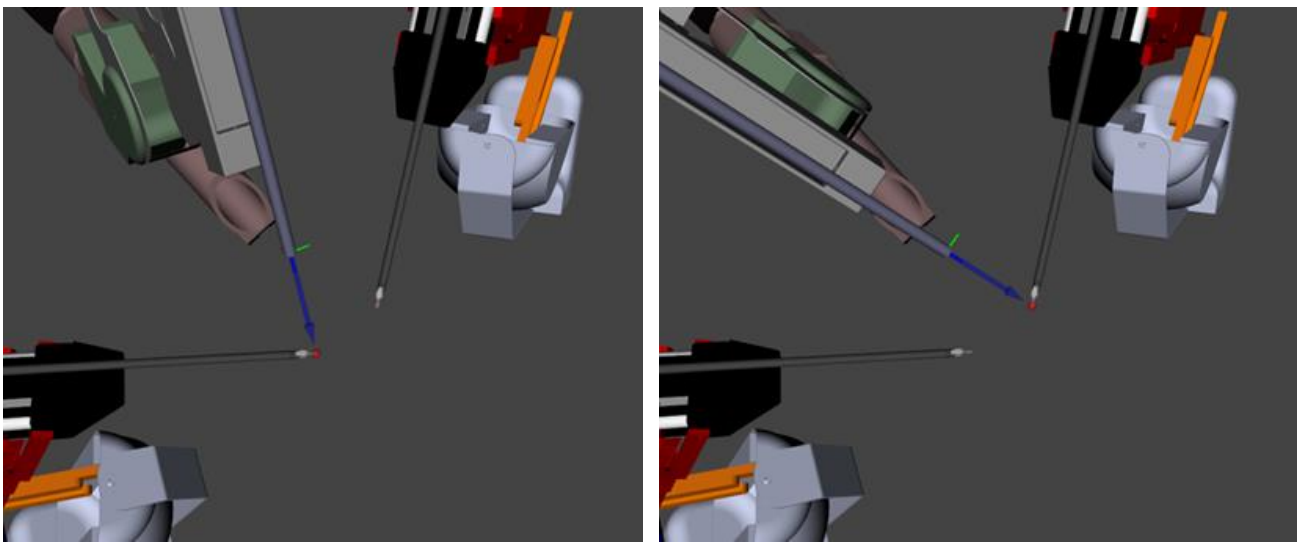
Track left/middle/right is an extension of the original autocamera algorithm that provides the da Vinci operator access to more preference-based settings that can easily be set and accessed by the voice assistant. The original autocamera algorithm is modified to relocate the midpoint, derived initially through the centroid of the two PSM positions, to reference the right or left PSM tool end effector. Depending on the operator's selection and through forward kinematics, Algorithm 3 finds the left and right tool 3D coordinates and then determines the rotation matrix to the endpoint of either tool. By setting the right or

left tool as the midpoint, the autocamera algorithm works to keep the selected tool within the center endoscopic field of view.

Algorithm 3 Track Tool.

```
1. function track(tool)
   Input: Single string value (tool) indicating which tool is to be tracked
   Output: None
2. autocamera_algorithm(joint_values, tool )
3.     psm1_pos ← forward_kinematics(joint_values ("psm1"))
4.     psm2_pos ← forward_kinematics(joint_values ("psm2"))
5.     if tool = right
6.         midpoint ← psm1_pos
7.     else if tool = left
8.         midpoint ← psm2_pos
9.     else
10.        midpoint ← (psm1_pos + psm2_pos) / 2
11.    rot ← ecm_to_midpoint_rotation(joint_values("ecm"), midpoint)
12.    ecm_desired ← inverse_kinematics(rot*ecm_current)
13.    move_joint(ecm_desired)
14. end
```

Figure 5 shows the changes to the desired viewpoint position (red dot) and the subsequent positioning of the endoscopic camera to track only that point. When either right or left is selected for tracking, the algorithm will ignore information about the position of the opposite manipulator, only focusing on maintaining the chosen tool within the operator's field of view. The operator can also voice their selection to track the middle, which will return to utilizing the original algorithm and centroid.



(**a**) (**b**)

Figure 5. This simulation shows how the camera moves to keep the left (or right) tool in the field of view. (**a**) Shows the endoscope tracking and pointing towards the left tool. (**b**) Shows the endoscope tracking and pointing to the right tool.

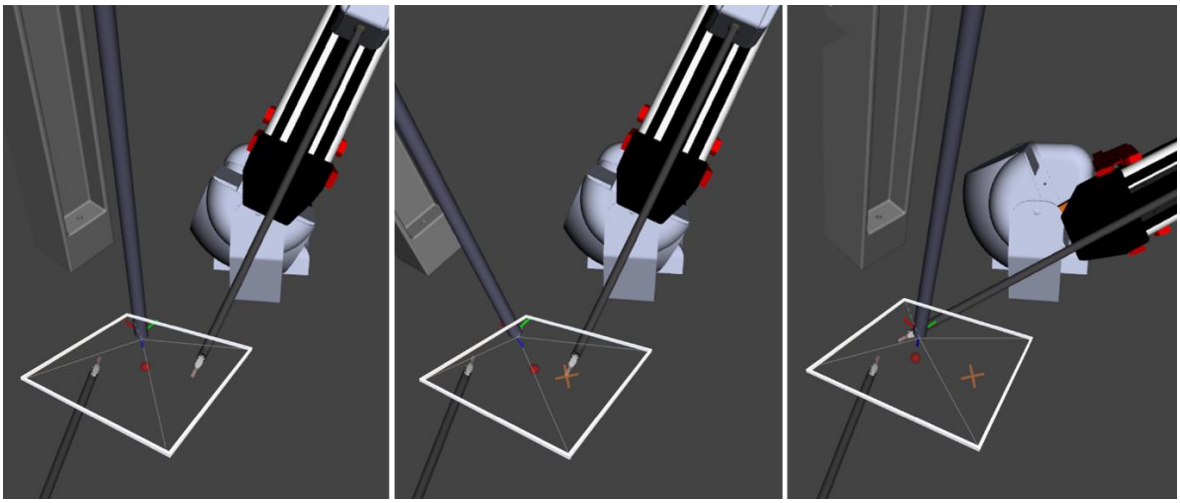
3.3.4. Keep Left/Middle/Right

Keep is another extension of the original autocamera algorithm. This command allows the surgeon to maintain another point in space chosen by one of the tools within the field of view. Shown in Algorithm 4, when the operator voices "keep left" or "keep right", the current position of either the left or right tool will be saved and used in the autocamera algorithm computation. The algorithm relies on the forward kinematics of either the right or left tool positions when the operator voices the selection to determine the saved position. That position is then maintained and utilized along with the midpoint of the two tools to create a centroid centered on the two PSM tools and the selected position. The autocamera algorithm factors in the third point to keep both tools and the saved position within the field of view. If the keep method is called without the right or left tool through voicing a command such as "keep middle" or "keep off", the algorithm will default back to the original midpoint of the two PSM tools and disregard the previously chosen position.

Algorithm 4 Keep Position.

```
function keep(tool)
Input: Single string value (tool) indicating which tool position needs to be kept
Output: None
if tool = "right"
    keep_pos ← forward_kinematics(joint_values ("psm1"))
    keep_set ← True
else if tool = "left"
    keep_pos ← forward_kinematics (joint_values ("psm2"))
    keep_set ← True
else
    keep_set ← False
autocamera_algorithm(joint_values, keep_pos, keep_set )
    psm1_pos ← forward_kinematics(joint_values ("psm1"))
    psm2_pos ← forward_kinematics(joint_values ("psm2"))
    if keep_set = True
        midpoint ← (psm1_pos + psm2_pos + keep_pos) / 3
    else
        midpoint ← (psm1_pos + psm2_pos) / 2
    rot ← ecm_to_midpoint_rotation(joint_values("ecm"), midpoint)
    ecm_desired ← inverse_kinematics(rot*ecm_current)
    move_joint(ecm_desired)
end
```

In Figure 6, the keep algorithm can be seen portrayed in simulation. The red dot corresponds to the desired camera viewpoint calculated in Algorithm 3 as the midpoint. The white box is a drawn-in representation of the camera frustum. In Figure 6a, the midpoint can be seen centered between the tools and the camera viewpoint before selecting the keep position. In Figure 6b, the selection of the keep position after the voice command is highlighted as the orange "X". It is at this point in which the end effector's position is saved, and the auto camera algorithm considers the position into its midpoint calculation. In this simulated scenario, "keep right" was commanded; thus, the right tool position is used in midpoint calculation and viewpoint selection. The effect of the save position can be seen by the midpoint marker in Figure 6b as it moves closer to the right tool even when the tools are closer together in a position that would remove the saved position from the field of view, as Figure 6c shows the newly configured midpoint remains in a position that allows it to be captured by the endoscopic field of view.



(**a**) (**b**) (**c**)

Figure 6. This figure shows how the viewpoint is kept between the selected point and the current tool position. The view centers around these two points along with any given point in the three-dimensional space. (**a**) Shows the camera view before selection. (**b**) Shows the adjusted camera view and the selected point drawn in as "X". (**c**) Shows the adjusted midpoint and camera view after selection and moving to keep the chosen point in view.

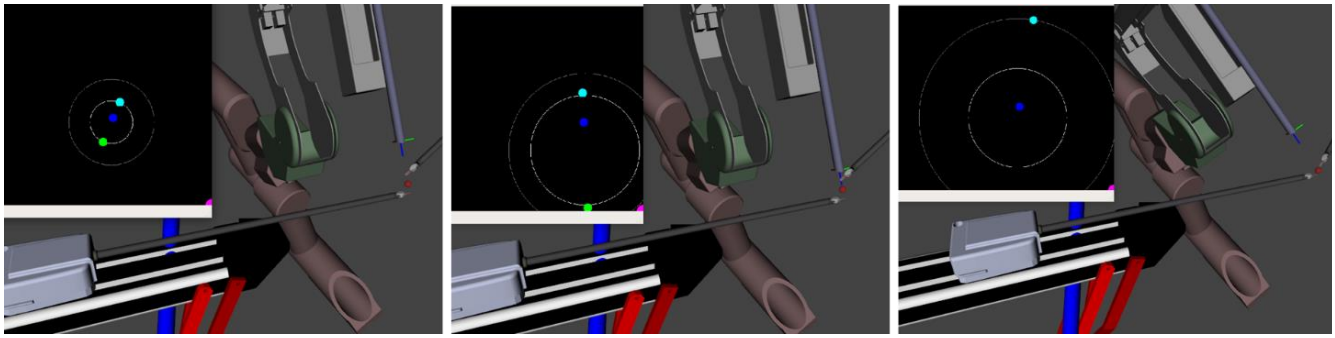
3.3.5. Change Inner/Outer Zoom

With the midpoint/tools in 2D camera coordinates, Algorithm 5 can be applied to maintain an appropriate zoom level and avoid unnecessary movement. The location of the tools in the 2D view determines the distance/zoom level. If the tools draw close together, the camera moves in. Conversely, as the tools move towards the outer edges of the view, the camera is zoomed out. There is also a dead zone to prevent camera movement if the tools are near the center of the view by an acceptable distance. The inner and outer edges of the dead zone are adjustable for different procedures and surgeon preferences. Those values are the original parameters of the autocamera that were maintained behind software configuration. Here, we expose these values to the operator through voice commands for preference-driven algorithm utilization. The zoom computation is altered to include the operator's voice-selected inner and outer zoom levels.

Algorithm 5 Inner/Outer Zoom Level Adjustment.

```
function inner_outer_zoom_adj_algorithm(inner, outer)
    Input: Two Float values (inner, outer) indicating the inner and outer zoom values
    respectively
    Output: Endoscopic hardware zoom value
adjust_zoom_level(joint_angles, cam_info, inner, outer)
    mid ← image_center
    psm1_pos, psm2_pos ← forward_kinematics(joint_angles('psm1','psm2'))
    dx,dy ← tool_to_mid_distance()
    ax,ay ← mid_to_image_edge_distance()
    if psm1_pos and psm2_pos in inner
        return min(dx/ax, dy/ay)
    else if psm1_pos and psm2_pos in outer
        return -1 * min((ax-dx)/ax, (ay-dy)/ay)
    else
        return
end
```

Figure 7 shows the real-time change in the simulated endoscopic camera view of the inner and outer zoom levels. Figure 7a shows the original parameter selection included in the startup of the autocamera. The inner circle indicates the inner zoom level, and the outer circle indicates the outer zoom level. The space between the two circles is referred to as the dead zone. The green and light blue dots in the simulated camera output are the 2D positions of the right and left PSMs, and the blue dot is the calculated midpoint between the two tools. Figure 7b shows the same view, but the inner zoom level increased from 0.08 to 0.2. After changing the inner zoom level, the endoscope manipulator zoomed out to move the tools from being inside the inner zoom level to just within the dead zone. Figure 7c shows the resultant position after setting the outer zoom value from 0.08 to the same inner value of 0.2. After moving the tools outside the outer zone, the endoscopic manipulator zooms out to maintain the right and left PSM positions to just within the dead zone.



(**a**) (**b**) (**c**)

Figure 7. The simulated camera view resulting from the Rviz simulation. (**a**) Shows the original parameters of the autocamera inner and outer zoom values. (**b**) The result in simulation of voicing the command to change the inner zoom level. (**c**) The result in simulation of voicing the command to change the outer zoom level.

4. Results/Discussion

4.1. Behavior of Voice Commands

4.1.1. Viewpoints Generated with Commands Issued

The "start" and "stop" commands activate the activation states of the autocamera algorithm. These commands allow the surgeon to quickly switch on the autocamera when necessary and switch it off again when manual control is desired. Performing this on-demand prevents the surgeon's needs from conflicting with the autocamera system.

The "find tools" command will move the camera such that both tools will be in view, as seen in Figure 8. This can be used by a surgeon operating without the autocamera algorithm to locate both tools quickly should they be out of view. It is more efficient than having to move the camera manually and adjusting the zoom level, and it is safer as the tools will be out of view for less time.

The "track" commands set the endoscopic camera to find the chosen tool (left/middle/right) and to keep it in view. In Figure 9, each set of four images demonstrates one command. The left image is an external photo of the setup, and the right image shows the view from the endoscopic camera. The difference between the top and bottom rows of each command is meant to relate the effect of the command. Figure 9a shows that when set to "track left," the camera centers on the left tool, regardless of the position of the right tool. In Figure 9b, the camera centers on the midpoint of the two patient-side manipulators, which is the original functionality of the autocamera algorithm. Figure 9c demonstrates the "track right" command, with the camera view focused on the right tool.

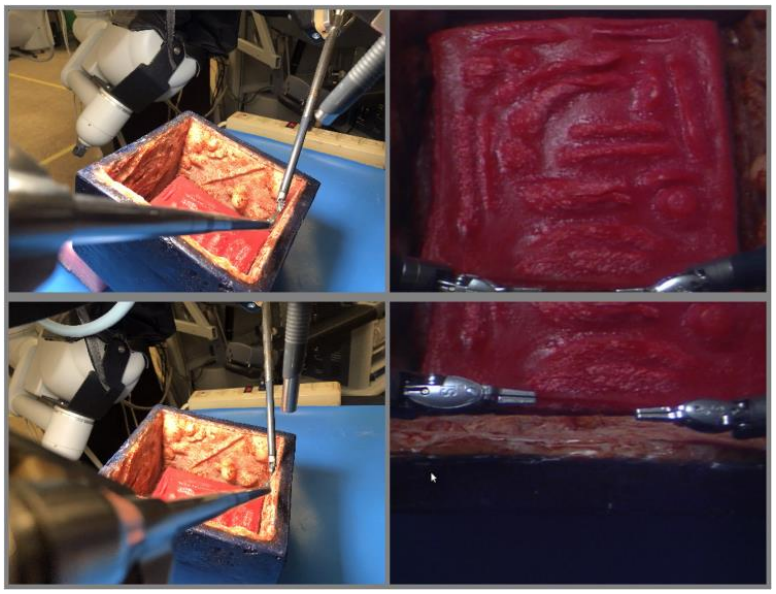


Figure 8. Demonstration of the "Find Tools" command. The "Find Tools" command begins with the tools at the edge of the camera view and is shown to move the camera to center the view on the tools. (**a**) Tools out of view. (**b**) Tools in view.

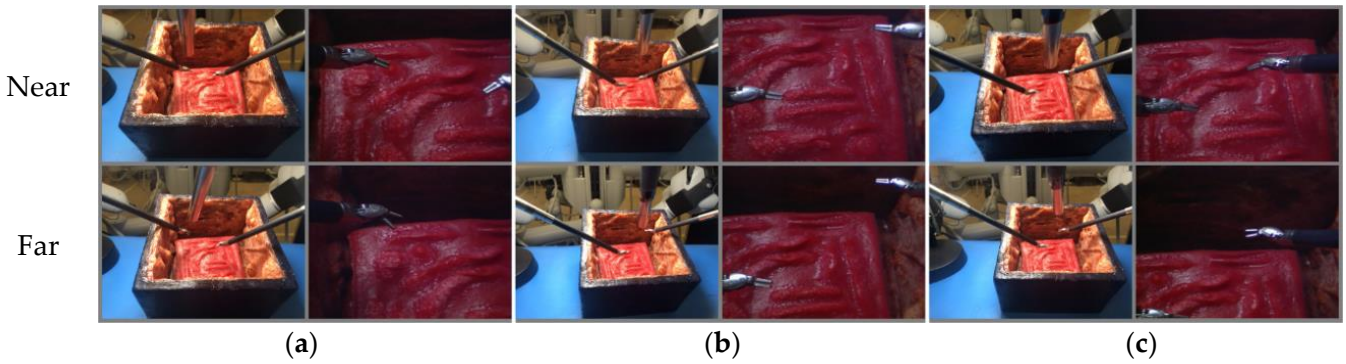
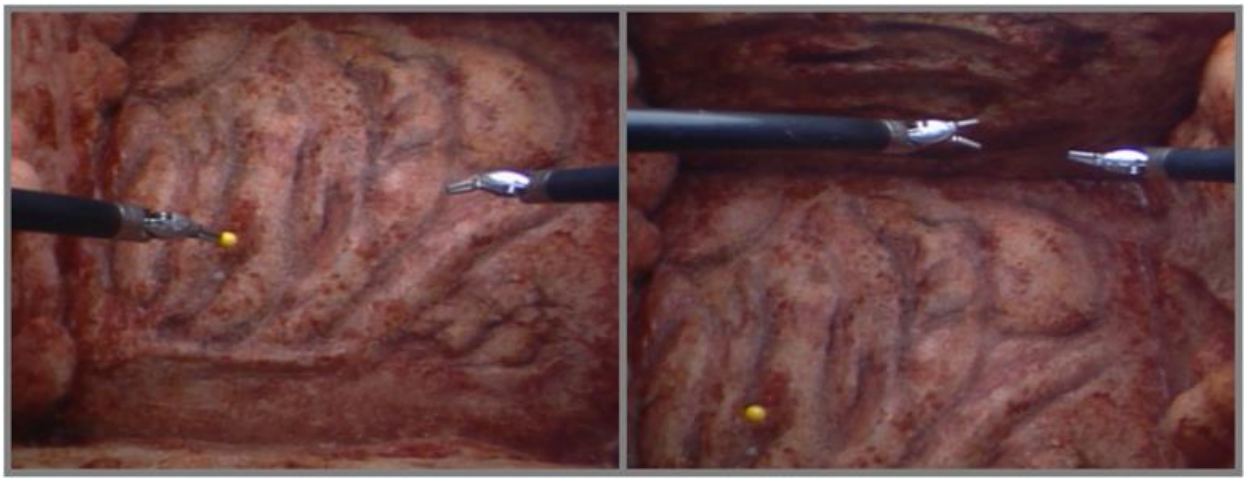


Figure 9. The set of "track" commands given to the surgical robot. (**a**) The result after the operator commands "track left". (**b**) The result after the operator commands "track middle". (**c**) The result after the operator commands "track right".

These commands will allow a surgeon to choose which tool to focus on during surgery without manually shifting the camera. This is particularly useful when one of the tools is used while the other sits idly or when one tool is being used more than the other. The "track" commands allow the surgeon greater flexibility in using the manipulators because they can have an unencumbered view if they do not require both tools.

The "keep" commands are used to set a position of interest to remain in the camera's view. The "keep left" will save the current position of the left tool and keep it in view even when the tools are moved away. The "keep right" command will do the same but for the right manipulator. As observed in Figure 10, the point is chosen with the "keep left" command, and it remains in view when the tools move to new positions.



(a) (b)

Figure 10. Demonstration of the "keep" command. The "keep" command used here is "keep left" and, as such, keeps the current position of the left arm located in the left image. The right image shows the yellow point remaining in view despite the tools moving far to the top of the scene. (**a**) Set Position. (**b**) Position kept in view.

The "keep" commands will allow surgeons to choose points of interest to keep in view during surgery. These points can be things such as a bleed, sutures, abnormalities, or other artifacts. These commands make it so that the surgeon does not need to constantly move the camera to check on points of interest and risk the tools going out of view, which is also a safety issue.

The "change inner/outer zoom" commands allow the user greater flexibility when the autocamera algorithm zooms in or out. In instances where the surgeon does not want the algorithm to zoom out, they can set a large value to the outer zoom level; moreover, in instances where they do not want the algorithm to zoom in, they can set a small value to the inner zoom level. In enlarging the inner and outer zoom levels equally, the surgeon can create a wider or narrower field of view. By changing one and not the other, they can increase the space within the dead zone while simultaneously viewing both a wide field of view when the tools are much further apart and a narrower detailed field of view when they are much closer together.

4.1.2. Voice Recognition Testing

We analyze the usability of our NLP models, specifically with the use of Alexa and Vosk in control of the dVRK. For our test set, we executed three trials consisting of three different individuals (authors). Each trial consisted of ten runs where the nine commands were incrementally spoken through. Rather than repeating the same command consecutively, we only voiced each command once per run. Repeating the commands over ten runs provided the potential for them to be misspoken and allowed us to assess how easily they can be articulated. Simultaneously, this can show how natural it is to use the voice interface when faced with multiple commands.

There were two primary variables we were interested in capturing for data collection. The first is the registration of the time when the voice recognition systems respond back with a sound indicating that our command has triggered the subsequent algorithmic action. The second is the percentage of which Alexa or Vosk correctly triggers the voiced command. Over the course of the three trials, we recorded each of the ten runs and used the recording along with the time provided by the application to analyze the accuracy and exact response time. We then measured the time it took for the skill to be registered by the software. Table 1 shows the accuracy comparison of commands spoken to both Alexa and Vosk. For each person's ten runs the commands that were correctly identified are represented as a

percentage. Similarly, the totals for all three trials are also represented as a percentage in the last column of the chart.

Table 1. Comparison of accuracy for each command over three trials and ten runs per trial. Each trial is a different person saying each of the commands ten times.

Command	Trial 1 Accuracy (%)		Trial 2 Accuracy (%)		Trial 3 Accuracy (%)		Total Accuracy (%)	
	Alexa	Vosk	Alexa	Vosk	Alexa	Vosk	Alexa	Vosk
start autocamera	100	100	100	90	100	100	100	96.67
stop autocamera	90	60	100	70	90	40	93.33	56.67
track right	90	90	90	80	100	90	93.33	86.67
track left	100	100	90	100	80	100	90	100
track middle	100	80	80	100	90	80	90	86.67
keep left	100	100	90	60	80	80	90	80
keep right	100	100	90	100	100	100	96.67	100
keep off	80	80	90	90	100	90	90	86.67
Find my tools	100	100	90	70	80	90	90	86.67

Table 2 shows the overall timed averages and accuracy of the 30 test runs. Of the 270 commands voiced relative to Alexa, only 20 were not recognized or misinterpreted. This produces an interpretation accuracy of 94.07%. Of the 270 commands voiced relative to Vosk, 36 were not recognized or misinterpreted, producing an accuracy of 86.67%. This accuracy can even be improved by creating more synonyms of natural commands to control the autocamera's algorithm and with increased fine tuning of the offline model. The average time for Alexa to complete the requested change was 1.51 s, whereas the average time for Vosk to complete the same request was 0.60 s.

Table 2. Total accuracy and total response time of all commands over the three trials.

Percent Command Accuracy		Total Response Average [1]	
Alexa	Vosk	Alexa	Vosk
94.07%	86.67%	1.51s	0.60s

[1] Of all commands understood and requests completed.

The analysis of the phrases with the highest rate of accuracy is presented in Figure 11. It is observed that the online system provides the most balanced set of accuracy with no noticeable issues with any particular command. The Vosk system, however, shows particular difficulty in recognizing certain commands. Future work should choose phrases that have the highest accuracy and finetune models to create a more balanced system.

(**a**) (**b**)

Figure 11. This graph shows the distribution of accuracy amongst all commands over the course of the three trails. (**a**) Shows the percent accuracy of the 270 commands voiced to the online-based Alexa system. (**b**) Shows the percent accuracy of the 270 commands voice to the offline-based Vosk system.

Voice recognition technology, especially that of offline based systems, is still an active research area. In our current work we notice a tradeoff between accuracy and time between the online and offline systems. Furthermore, Alexa customization is limited by what is allowed by Amazon, including the implementation of only a few hot words, off-site processing of voice commands, a microphone that can only be on for a limited amount of time, and the need for extra phrases to trigger commands. Vosk, however, can overcome some of those nuances of Alexa and Amazon's usage requirements by allowing better customization and implementation of commands and hot words, which are less tedious for the surgeon.

4.1.3. Safety Concerns for Use in Operating Room

We acknowledge the safety concerns surrounding voice recognition in the operating room. Based on our initial testing, we can improve accuracy with further speech recognition training and tuning, especially with a limited number of commands. Over the course of multiple surgeries, the system can improve as systematic data of commands are used as feedback for further training of the NLP. For the OR, we foresee an offline implementation for patient privacy concerns and a directional microphone with appropriate filtering to only allow the surgeon's voice commands. Additionally, we would further optimize and limit the commands to the ones specifically needed for endoscope control. There should be immediate feedback for the recognized commands to indicate that they will be executed. The system should also ask for clarification to repeat commands when unsure. Our approach is geared towards camera manipulation, which is inherently safer than tool manipulation since the camera is distant from the patient's tissue. However, a physical cut-off switch for any inadvertent endoscope movement should also be added.

5. Conclusions

The current clinical practice paradigms are to have either a separate camera operator (for traditional laparoscopic surgery) or a surgeon-guided camera arm (for fully robotic surgery). As stated in our previous work [19], there are several issues with these two methods of camera control. Our previous work performed a quantitative human test comparison with respect to a separate camera operator, a surgeon-guided camera, and our autonomous camera systems. It showed that the autonomous algorithm outperformed the traditional clutch and move mode of control. Our proposed project seeks to shift clinical practice by introducing a form of autonomous and customizable robotics. Unlike existing autonomous camera systems, our system operates with surgeon input/direction, which may improve performance and creates a true partnership between a robotic camera system and the human operator. At the same time, a camera system has little direct interaction with the patient; thus, it represents a safer avenue for the introduction of autonomous robotics to surgery. Given that this work is an extension of our autocamera algorithm, we expect to see improvements in user performance in a future subject study.

This work is novel and will improve clinical practice in several ways. First, it improves the interaction between the robot(s), autonomous camera system(s), and the human to produce efficient, fault-tolerant, and safer systems. There is no current research that studies the interaction of an automated camera system and the human in the loop. Second, it was designed using guidance from experts. We leveraged this knowledge to provide a framework for intelligent autonomous camera control and robot/tool guidance. Alleviating the physical and cognitive burden of camera control will allow telerobotic operators to focus on tasks that better use uniquely human capabilities or specialized skills. This will allow tasks to be completed in a safer and more efficient fashion. Thus, our research will enable cooperative robots to effectively support and partner with human operators to enable more robust robotic surgeries.

The natural language enhanced automated systems will supplement the technical capabilities of surgeons (in both fully robotic and traditional laparoscopic procedures) by providing camera views that help them operate more accurately and with less mental

workload, potentially leading to fewer errors. The effect on clinical practice will be safer procedures, lowered costs, and a consistent, automated experience for surgeons.

In the future, Natural Language Processing can be extended beyond camera control. For instance, using our previous work on bleeding detection and prediction [30,31], an overwatch system can be created to verbally warn the surgeon about unsafe tool movements or even attenuate robot movements. In addition, using recording capability [32], the surgeon could easily ask to record videos or even movements for later use. Moreover, annotations during surgery for teaching and documentation purposes could be easily achieved with voice interaction. The software for the system implementation is available online [33–35]. A video of the system in operation is also available [36].

Author Contributions: Conceptualization, A.P.; methodology, M.E. and A.P.; software, M.E., A.P. and L.J.; validation, M.E., A.P., L.J. and M.H.; formal analysis, M.E., A.P., L.J. and M.H.; investigation, M.E., A.P., L.J. and M.H.; resources, A.P.; data curation, M.E., A.P., L.J. and M.H.; writing—original draft preparation, M.E., A.P., L.J. and M.H.; writing—review and editing M.E., A.P. and L.J.; visualization, M.E., A.P. and L.J.; supervision, A.P.; project administration, M.E. and A.P.; funding acquisition, A.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported with funding from the Michigan Translational Research and Commercialization (MTRAC), grant number 380137/23R343 and 192748/117E37; and from the US Department of Veterans Affairs National Center for Patient Safety under grant "NCPS Robotic Operations Task Excursion Analysis" (VA701-15-Q-O179/2VHF).

Acknowledgments: The authors of this paper would like to thank Mohammed Alawad for his expertise on voice recognition and language processing; Moaz Elazzazi for editing together the video used for demonstration of our work; Meghan Martin and Dayna Green, co-founders of Gals & Ghouls, for their artistic work put into the realistic testbed seen in the resultant images; and both David Edelman and Michael Klein for their expert medical insight and consultation.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. D'Ettorre, C.; Mariani, A.; Stilli, A.; Rodriguez y Baena, F.; Valdastri, P.; Deguet, A.; Kazanzides, P.; Taylor, R.H.; Fischer, G.S.; DiMaio, S.P. Accelerating Surgical Robotics Research: A Review of 10 Years with the da Vinci Research Kit. *IEEE Robot. Autom. Mag.* **2021**, *28*, 56–78. [CrossRef]
2. Pandya, A.; Reisner, L.A.; King, B.; Lucas, N.; Composto, A.; Klein, M.; Ellis, R.D. A Review of Camera Viewpoint Automation in Robotic and Laparoscopic Surgery. *Robotics* **2014**, *3*, 310–329. [CrossRef]
3. Ellis, R.D.; Munaco, A.J.; Reisner, L.A.; Klein, M.D.; Composto, A.M.; Pandya, A.K.; King, B.W. Task analysis of laparoscopic camera control schemes. *Int. J. Med. Robot. Comput. Assist. Surg.* **2016**, *12*, 576–584. [CrossRef] [PubMed]
4. Staub, C.; Can, S.; Knoll, A.; Nitsch, V.; Karl, I.; Färber, B. Implementation and evaluation of a gesture-based input method in robotic surgery. In Proceedings of the 2011 IEEE International Workshop on Haptic Audio Visual Environments and Games, Qinhuangdao, China, 14–17 October 2011; pp. 1–7.
5. Allaf, M.E.; Jackman, S.V.; Schulam, P.G.; Cadeddu, J.A.; Lee, B.R.; Moore, R.G.; Kavoussi, L.R. Laparoscopic visual field: Voice vs foot pedal interfaces for control of the AESOP robot. *Surg. Endosc.* **1998**, *12*, 1415–1418. [CrossRef] [PubMed]
6. El-Shallaly, G.E.; Mohammed, B.; Muhtaseb, M.S.; Hamouda, A.H.; Nassar, A.H. Voice recognition interfaces (VRI) optimize the utilization of theatre staff and time during laparoscopic cholecystectomy. *Minim. Invasive Ther. Allied Technol.* **2005**, *14*, 369–371. [CrossRef] [PubMed]
7. Kraft, B.M.; Jäger, C.; Kraft, K.; Leibl, B.J.; Bittner, R. The AESOP robot system in laparoscopic surgery: Increased risk or advantage for surgeon and patient? *Surg. Endosc.* **2004**, *18*, 1216–1223. [CrossRef] [PubMed]
8. Mellia, J.A.; Basta, M.N.; Toyoda, Y.; Othman, S.; Elfanagely, O.; Morris, M.P.; Torre-Healy, L.; Ungar, L.H.; Fischer, J.P. Natural Language Processing in Surgery: A Systematic Review and Meta-analysis. *Ann. Surg.* **2021**, *273*, 900–908. [CrossRef] [PubMed]
9. Mettler, L.; Ibrahim, M.; Jonat, W. One year of experience working with the aid of a robotic assistant (the voice-controlled optic holder AESOP) in gynaecological endoscopic surgery. *Hum. Reprod.* **1998**, *13*, 2748–2750. [CrossRef] [PubMed]
10. Mewes, A.; Hensen, B.; Wacker, F.; Hansen, C. Touchless interaction with software in interventional radiology and surgery: A systematic literature review. *Int. J. Comput. Assist. Radiol. Surg.* **2016**, *12*, 291–305. [CrossRef] [PubMed]
11. Nathan, C.O.; Chakradeo, V.; Malhotra, K.; D'Agostino, H.; Patwardhan, R. The voice-controlled robotic assist scope holder AESOP for the endoscopic approach to the sella. *Skull Base* **2006**, *16*, 123–131. [CrossRef] [PubMed]
12. Perrakis, A.; Hohenberger, W.; Horbach, T. Integrated operation systems and voice recognition in minimally invasive surgery: Comparison of two systems. *Surg. Endosc.* **2013**, *27*, 575–579. [CrossRef] [PubMed]

13. Unger, S.; Unger, H.; Bass, R. AESOP robotic arm. *Surg. Endosc.* **1994**, *8*, 1131. [CrossRef] [PubMed]
14. Azizian, M.; Khoshnam, M.; Najmaei, N.; Patel, R.V. Visual servoing in medical robotics: A survey. Part I: Endoscopic and direct vision imaging—Techniques and applications. *Int. J. Med. Robot. Comput. Assist. Surg.* **2014**, *10*, 263–274. [CrossRef] [PubMed]
15. Wei, G.-Q.; Arbter, K.; Hirzinger, G. Real-time visual servoing for laparoscopic surgery. Controlling robot motion with color image segmentation. *Eng. Med. Biol. Mag. IEEE* **1997**, *16*, 40–45.
16. Bihlmaier, A.; Worn, H. Learning surgical know-how: Dexterity for a cognitive endoscope robot. In Proceedings of the Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM), 2015 IEEE 7th International Conference on Engineering Education (ICEED), Kanazawa, Japan, 17–18 November 2015; pp. 137–142.
17. Da Col, T.; Mariani, A.; Deguet, A.; Menciassi, A.; Kazanzides, P.; De Momi, E. Scan: System for camera autonomous navigation in robotic-assisted surgery. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 2996–3002.
18. Eslamian, S.; Reisner, L.A.; King, B.W.; Pandya, A.K. Towards the implementation of an autonomous camera algorithm on the da vinci platform. In *Medicine Meets Virtual Reality 22*; IOS Press: Amsterdam, The Netherlands, 2016; pp. 118–123.
19. Eslamian, S.; Reisner, L.A.; Pandya, A.K. Development and evaluation of an autonomous camera control algorithm on the da Vinci Surgical System. *Int. J. Med. Robot. Comput. Assist. Surg.* **2020**, *16*, e2036. [CrossRef]
20. Weede, O.; Bihlmaier, A.; Hutzl, J.; Müller-Stich, B.P.; Wörn, H. Towards cognitive medical robotics in minimal invasive surgery. In Proceedings of the Conference on Advances in Robotics, Pune, India, 4–6 July 2013; pp. 1–8.
21. Weede, O.; Monnich, H.; Muller, B.; Worn, H. An intelligent and autonomous endoscopic guidance system for minimally invasive surgery. In Proceedings of the IEEE International Confrence on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 5762–5768. Available online: https://go.exlibris.link/j6RcCL1h (accessed on 21 March 2022). [CrossRef]
22. Composto, A.M.; Reisner, L.A.; Pandya, A.K.; Edelman, D.A.; Jacobs, K.L.; Bagian, T.M. Methods to Characterize Operating Room Variables in Robotic Surgery to Enhance Patient Safety. In *Advances in Human Factors and Ergonomics in Healthcare*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 215–223.
23. Vosk Offline Speech Recognition API. Available online: https://alphacephei.com/vosk/ (accessed on 15 January 2022).
24. Chen, Z.; Deguet, A.; Taylor, R.; DiMaio, S.; Fischer, G.; Kazanzides, P. An Open-Source Hardware and Software Platform for Telesurgical Robotics Research. In Proceedings of the MICCAI Workshop on Systems and Architecture for Computer Assisted Interventions, Nagoya, Japan, 22–26 September 2013.
25. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the ICRA Workshop on Open Source Software, Kobe, Japan, 12–17 May 2009; p. 5.
26. Open Source Robotics Foundation. RViz. 2015. Available online: http://wiki.ros.org/rviz (accessed on 1 March 2016).
27. Alexa Skills Builder. Available online: https://developer.amazon.com/en-US/alexa (accessed on 24 January 2022).
28. Ngrok. Available online: https://ngrok.com/ (accessed on 2 March 2022).
29. Povey, D.; Ghoshal, A.; Boulianne, G.; Burget, L.; Glembek, O.; Goel, N.; Hannemann, M.; Motlicek, P.; Qian, Y.; Schwarz, P. The Kaldi speech recognition toolkit. In Proceedings of the IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, Waikoloa, HI, USA, 11–15 December 2011.
30. Rahbar, M.D.; Reisner, L.; Ying, H.; Pandya, A. An entropy-based approach to detect and localize intraoperative bleeding during minimally invasive surgery. *Int. J. Med. Robot. Comput. Assist. Surg.* **2020**, *16*, 1–9. [CrossRef]
31. Daneshgar Rahbar, M.; Ying, H.; Pandya, A. Visual Intelligence: Prediction of Unintentional Surgical-Tool-Induced Bleeding during Robotic and Laparoscopic Surgery. *Robotics* **2021**, *10*, 37. [CrossRef]
32. Pandya, A.; Eslamian, S.; Ying, H.; Nokleby, M.; Reisner, L.A. A Robotic Recording and Playback Platform for Training Surgeons and Learning Autonomous Behaviors Using the da Vinci Surgical System. *Robotics* **2019**, *8*, 9. [CrossRef]
33. Available online: https://github.com/careslab/dvrk_voice (accessed on 15 February 2022).
34. Available online: https://github.com/careslab/dvrk_autocamera (accessed on 15 February 2022).
35. Available online: https://github.com/careslab/dvrk_assistant_bridge (accessed on 15 February 2022).
36. Available online: https://youtu.be/UZa7xCtOYT0 (accessed on 15 February 2022).