Robot Control of a Surgical Laser Waveguide

Aishwary Jagetia, Alka Choudhary, Amey Sutavani, Animesh Nema,
Himanshu Raghuvanshi, Jighjigh Tersoo-Ivase, Nishant Doshi

Robotics Engineering Program, Worcester Polytechnic Institute, MA

***Abstract*— Currently various surgeries are performed using surgical laser sources and stereo-microscopes. Automated tissue ablation offers many advantages over free hand surgical procedures as it is faster, more accurate, much efficient and offers great dexterity, particularly for complex ablations. This paper discusses the dynamic modeling and control of the ABB IRB 120 robot that has a surgical laser waveguide, attached to its end effector. A graphical user interface has been designed that utilizes the camera feed to generate the trajectory points for the robot to move in the work space. The trajectory points are then sent to the controller via ROS communication. As a result, the laser waveguide follows a desired trajectory with a predetermined velocity to accomplish the task.**

I. INTRODUCTION

For a very long time, robots and surgery existed as two independent paths and has been growing as well. During the late 1980s and early 1990s, endoscopic techniques were booming, and limitations were being reached as well. Subsequently, the potential capability of tele-robotics in MIS was well recognized. However, robots and surgery only reached a safe enough stage for their combination via tele-manipulation for surgical innovation in the last few years. The robotic surgical system is truly an information system rather than a machine, and it can be simply divided into input, analysis and output. A human is interposed between the input and output instead of a computer in case there are any unexpected events or anatomy during surgery, and these components serve as a tele-operation system. The input side consists of several chemical and biologic sensors and images, and there are various devices on the output side, such as manipulators and lasers, to contact organs and tissues.

Throughout the twenty-first century, robotic surgery has been used in multiple oral surgical procedures for the treatment of head and neck tumors and non-malignant diseases. With the assistance of robotic surgical systems, these surgeries are performed with less blood loss, fewer complications, shorter hospitalization and better cosmetic results than standard open surgery.

This paper focuses on developing an automated surgical procedure aimed at increasing safety, accuracy and efficiency. A graphical user interface (GUI) is designed that uses the camera feed to display the target space. The laser waveguide attached robot arm can then be made to follow a desired trajectory that is generated by the user through the GUI.

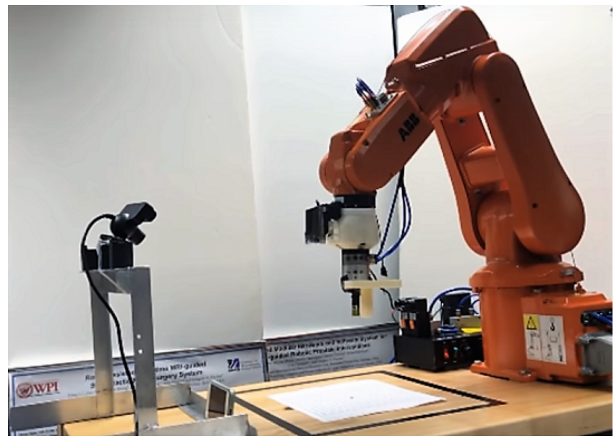

Fig. 1. Workspace Setup

II. LITERATURE REVIEW

In[2], laser cutting was performed by using fiber optical tips with the help of laser beam manipulators which could be controlled either with a joystick by the surgeon or computer controlled by determining pre defined trajectories. This was mainly implemented on osteotomy (surgical cutting or removal of bones). The point however, is that a non-contact surgery can be done with great precision by using laser beams and a set of predefined trajectories can be programmed into the system so that the robot arm (ABB IRB120 in our case) can execute these trajectories.

In[3], the author has used an ultrasound imaging and control the depth of the cut by altering the power of the laser. The focus of the paper was mainly on pruning/decimating unwanted internal structures such as operating on the prostrate gland. However, we found this paper useful for our purposes regarding alternate ways for imaging and controlling depth of tissue ablation.

In[9], they have presented a robotic vision system which automatically retrieves and positions of the surgical instrument safely from an unknown position. LEDs are attached to the tip of the instrument along with optical fibers to project laser dots on the surface of the organs. Using the endoscopic image to detect these optical markers, the instrument is centered by means of a visual servoing algorithm. This provides a relative position between the instrument and the pointed organ. This system can also be used to move the instrument at a position specified by the surgeon with the touch screen or mouse-type device. With the use of optical galvano-scanner they estimate the 3-D surface of the scanned organ. They have also presented the control scheme used to position the instrument by automatic vision feedback and the method for estimating the distance from the instrument to the organ in their paper.

In[4], the authors discuss the impacts of laser surgery (thermal damage) on the types of tissues. They carried a number of experiments to reduce the impact of thermal damage on the non-ablated tissue by modifications in laser wavelength, irradiance and exposure duration. We could use their findings to optimize the ablations performed by our robotic arm.

In[10], they presented the potential and difficulties of mathematical trajectory optimization methods. Derivation of the dynamic model, limitation of the angle of rotation, online robot controller, and internal path planning methods are all very well documented in the paper. The refined Direct Transcription (DT) method is used for optimizing point to point trajectory. With the constrained optimal control and the optimal trajectory, best possible robot trajectories can be obtained.

In[11], they have developed ABBY a prototype of an Industrial Mobile Manipulator platform composed of ABB IRB-120 industrial robotic manipulator. To accelerate system development they have used Robot Operating System (ROS) and ROS Industrial facilitated development and drives for ABB industrial manipulator. They have used cRIO for mobile base control and IRC5 robot controller for ABB robotic arm. Two ROS nodes where used to communicate with the server, the first node subscribes to ROS trajectory message and the other connects to the IRC5 controller and listens to state information from the controller. Based on these messages other ROS nodes determines the position of the robot's arm and adds feedback to the arm planning nodes. With the help of ROS's arm navigation slack they have performed forward and inverse kinematics with collision checking for Arm trajectory planning. They have also used LIDAR and Kinect camera for obstacle avoidance manipulable object detection, which was integrated with the help of open source drives compatible with ROS.

III. Problem Statement

- Formulate the kinematic model for IRB-120 and implement trajectory tracking in Gazebo.
- Enable robotic control of laser positioning.
  1) Implement a Communication channel via middleware (ROS)
  2) Robot position control from a computer
- Realize a library of basic motion primitives (laser scanning patterns)

IV. Setup

A. The ABB IRB120

The ABB IRB120 robot arm has been used that can either be operated with a teach Pendant (manually guiding the robot arm at set speeds) or via ROS communication channels. The nature of our task requires the control of the ABB IRB120 via ROS communication channels. *Figure 2* shows an image of the IRB120.

Fig. 2. ABB IRB 120 6 DOF arm

B. The Sharplan 30W Surgical Laser

The Sharplan 30W Surgical Laser setup consists of a portable laser generator *Figure 3* and has a control panel for handling the laser output, such as the laser frequency and the power. An optical fiber *Figure 4* is used to channel the laser from the generator. The optical fiber has been attached to the end effector tool of the robot.

Fig. 3. Sharplan CO2 30W Surgical Laser

Fig. 4. Laser Fiber

C. The End Effector Tool

The end effector tool *Figure 5* has been prototyped via additive manufacturing and holds the optical fiber in its place.

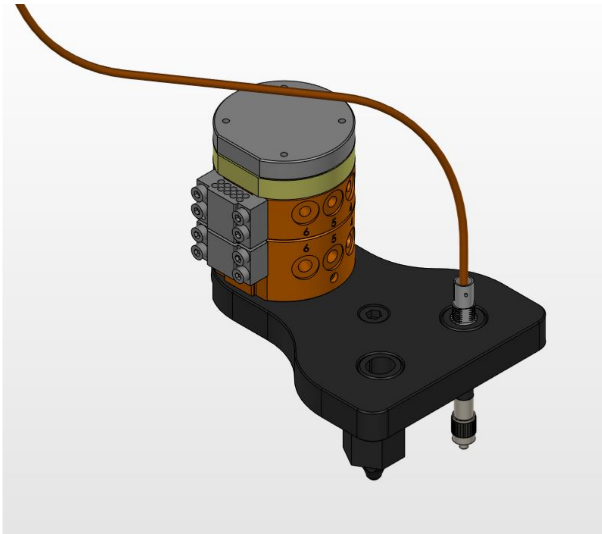

Fig. 5. End effector Design

D. Camera

The camera has been setup at a distance from the robot arm and captures the work space the arm has to work on. The camera feed is then converted to perspective view and utilized by the Graphical user interface to generate trajectory points.

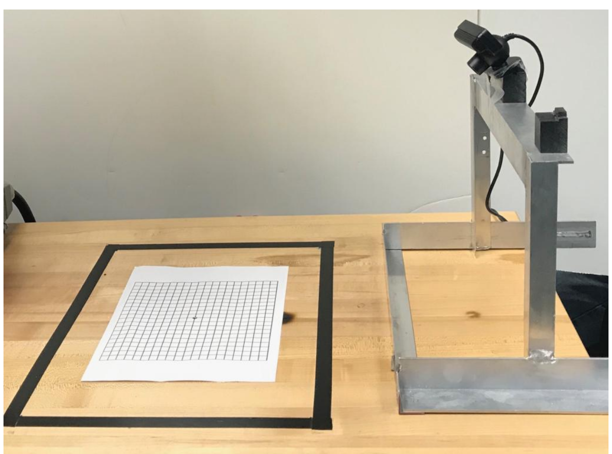

Fig. 6. Camera and Workspace Setup

V. IMPLEMENTATION

A. Graphical User Interface

Since the surgical plan that the robot follows is given by an experienced operator, the user interface available to them is an important part of the whole system. We have developed an intuitive user interface where the operator can draw the desired trajectory on a live video feed. In order to convert the video feed and the points drawn on it into usable task space points for the robot, there are a series of necessary calibration steps:

1) *Perspective shift:* Since the camera has been placed at the edge of the test setup, the raw video feed that we get has a perspective distortion. Therefore, the first calibration steps asks the operator to double click 4 points in the video feed which correspond to the rectangular area of where the surgery will take place. Using these points, the subsequent frames that the operator sees is similar to a top down view of the surgery area. This also eliminates the needs to calibration of the camera orientation itself, since the transformation can take place from any given angle as long as the operator marks the interested area correctly.

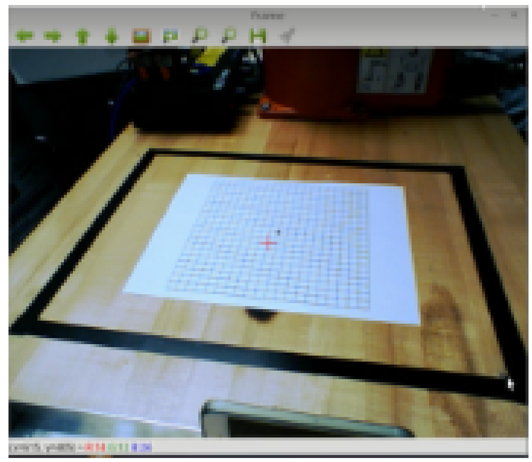

Fig. 7. Raw feed from Camera

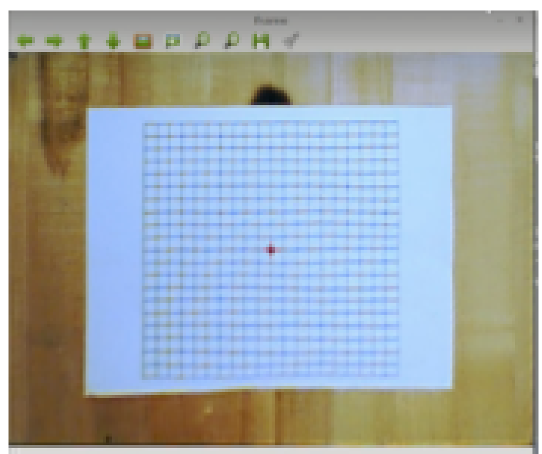

Fig. 8. Corrected Perspective view

2) *Origin of the visible area:* Once the operator can see the top down view of the test area, the whole frame consists of the test grid that we have placed on the setup table. This grid is stationary and the position of its origin is measured with respect to the origin of the robot base frame. This is necessary in order to translate the trajectory in the form of pixel value to a series of desired points with respect to the robot's frame.

After calibration, the operator can directly draw on the live feed to generate the trajectory that the robot should follow.

In the current implementation, the operator chooses the way-points by double clicking them onto the camera feed. They are allowed to choose as many points as required. Once all the way points are confirmed, the operator locks them by pressing the *'c'* key. An example user input is shown in Figure 9. The final trajectory is later generated by interpolating between the given way-points. This also allows the operator to achieve a simple form of velocity control over certain parts of the trajectory by selecting the spacing of the way-points as necessary. Apart from the double clicking every single way-point, the operator can also drag the mouse pointer on the camera feed, which will automatically generate the necessary way-points for following the given path. These way-points are then transformed into the robot's frame and published to the robot controller.

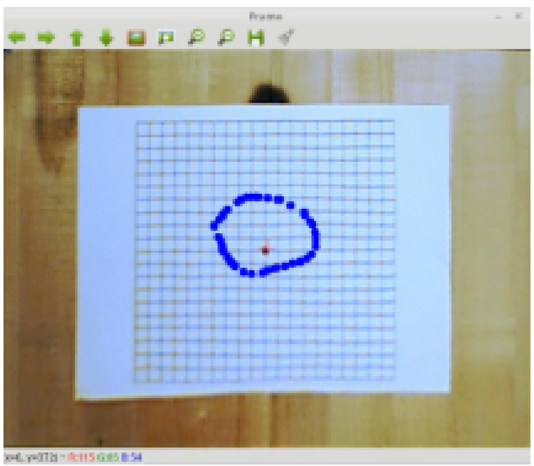

Fig. 9. Generating trajectory points

B. Controller

After receiving way-points from the operator, the software achieves the motion of the robot's joints such that the laser accurately follows the given trajectory. This is done in two stages: Converting the way-points from task space to joint space using inverse kinematics, and generating a trajectory in joint space using those way-points.

1) *Realizing Forward and Inverse Kinematics libraries*: The main package is written in Python at a higher level and features easy trajectory input interface, modules for computing robot kinematics and ROS communication wrappers. This maintains a clear abstraction of the back-end complexities from the end user while providing modularity and an easy-to-use API for accessing the package functionalities if required by a developer. As we needed inverse kinematics solution to be a part of the generic package, we explored different implementations based on Iterative Jacobian Inverse and the analytical solutions. We have implemented ROS packages for both methods. The Analytical solution of the Inverse Kinematics are available in the *ik_analytical* package. This package is implemented using C++. However, there are current bugs in this package, which we haven't isolated which result in giving an incorrect solution of th last 3 joints of of the robot which handle the orientation of the robot. We believe this is due to mismatch in the frames that we have assumed, and the frames that are chosen in the URDF file, because we are pretty sure about the equations that we are using. The Iterative Jacobian Inverse method has been implemented using Python scripts and is implemented in the *RobotUtils* package. The architecture of this package is illustrated in Figure **??** We were more inclined towards using this methods since the code doesn't have to change for a robot with different dimension. And since the final aim of this system is to be implemented on a different robot, we figured this would be a more flexible choice. This IK worked, but we chose not to deploy this on the real robot because we were getting erratic joint configurations in some configurations, which is undesirable when working with the real hardware. Therefore, in our implementation, we have used the inverse kinematics libraries available in ROS using the MoveIt! framework.

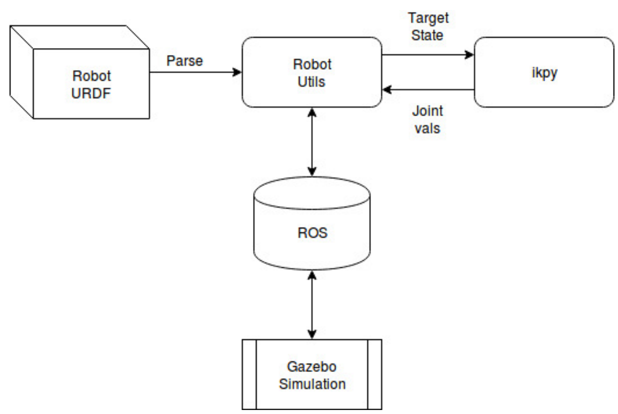

Fig. 10. Robot Utils: Flow Diagram

2) *Trajectory Generation*: For generating a feasible trajectory between way-points, we have used the MoveIt! framework available in ROS. It uses the Open Motion Planning Library (OMPL) which is an open source library using several state-of-the-art sampling based algorithms. We chose to use this because the trajectory messages that are used for communicating a particular trajectory are robustly handled by in the library. There also support of visualization using RViz. By giving the way-points of the desired trajectory, MoveIt! generates a continuous trajectory. This motion plan is then sent to the IRB120 controller using the ROS-Industrial packages which handle the communication between the robot and the operator system.

C. Hardware

The hardware design of this project consists of creating a feasible end effector assembly which allows easy assembly and a secure housing of the laser fiber. The hardware design

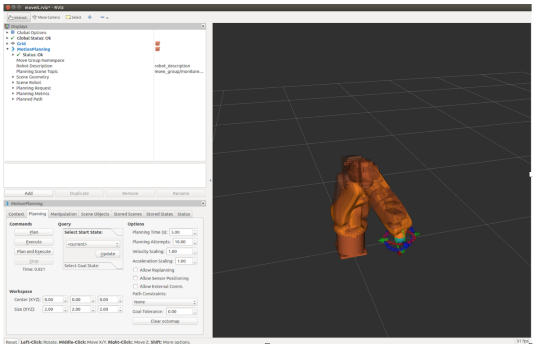

Fig. 11. MoveIt! Visualisation in RViz

has three components to it. The automatic tool changer, the interface plate and the fiber optic connector. The automatic tool changer was designed by a previous team which uses pneumatics to allow changing the end effectors as necessary. Figure 12 shows the complete assembly of the end effector. The tool changer is the orange part of the assembly. The black plate is the interface plate which houses the laser fiber connector and provides a stable connection to the tool changer. The fiber optic connector is a simple tube like structure which is attached to one of the holes of the interface plate and holds the end of the laser fiber from which the laser comes out. The tool changer is a custom made metal piece and the other two components are 3D printed.

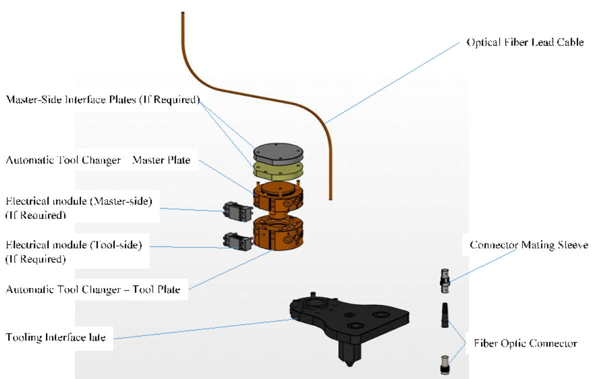

Fig. 12. End effector Design

VI. CONCLUSION

The software was able to successfully use the trajectory inputs in the form of way points, transform them into the robot's base frame, plan a path between way points and send over the desired joint trajectories to the robot hardware/simulation. The robot executes a calibration routine which ensures offset rejection and re calibrates the joint values. The current status of the project does complete two of the three objectives discussed during the proposal. The only part that was left untested was the velocity control of the end-effector as we found that the maximum end effector velocity depended directly on the arm configuration and was lower for configurations near singularity and joint limits. The velocity control is possible though the current framework as the messages being sent to the robot are joint trajectory messages and the velocity can be specified in them. The communication middleware is in place and the scanning patterns can be defined in a file which can be read by the software instead of freehand input from GUI. There are a few issues to be addressed though which are discussed in the next section.

VII. FUTURE WORK

This section discusses some problems that have been temporarily solved but could use a more formal solution, some suggestive improvements and extensions to the project. Firstly, the robot would develop an offset in spite of the initial calibration step which calls for a more robust workspace and robot calibration method to be used. Also, the current frame transformation involves perspective transform from a camera image which can be prone to errors. This can be avoided by freezing a camera up top the worktable with pre-calibration to concretely define the transform. Next, velocity control has to be well tried and tested to establish a relation between different joint configurations and the maximum possible linear end effector velocity possible at that configuration. Another concern is the nature of messages being sent: current setup uses each way-point as a messages and and uses pauses to avoid controller timeouts. But a more formal approach would be to pass planned trajectories instead of way-points which doesn't work as of now because of controller timeout and thus losses but future updates to MoveIt! might fix it.
Extensions include implementing a visual feedback based depth control along with trajectory tracking by processing camera images to estimate current depth of the ablation and adjust the lateral speed of the end effector accordingly. A long shot can be providing this depth as a haptic feedback to the surgeon to make it more intuitive and create an abstract separation between the inner engineering of the framework and the medical specifics.

REFERENCES

[1] Fernando Reyes, Rafael Kelly, Experimental evaluation of model-based controllers on a direct-drive robot arm,Mechatronics,Volume 11, Issue 3,2001,Pages 267-282,ISSN 0957-4158,https://doi.org/10.1016/S0957-4158(00)00008-8.

[2] Stibinger S. Landes C. Seitz. O., and Sader R. J. Periodontol. Er:YAG Laser Osteotomy for Intraoral Bone Grafting Procedures: A Case Series With a FiberOptic Delivery System 2007 December, 78(12):2389-94.

[3] Israel Barken, Apparatus and method for computer controlled laser surgery. Patented US4672963A

[4] Walsh JT Jr, Flotte TJ, Anderson RR, Deutsch TF, Pulsed CO2 laser tissue ablation: effect of tissue type and pulse duration on thermal damage. Lasers Surg Med. 1988;8(2):108-18.

[5] Morten Engell-Norregard, Inverse Kinematics The state of the art.

[6] Lukas Barinka, Roman Berka, Inverse Kinematics-Basic Methods.

[7] Jianmin Zhao, Norman Badler, Inverse Kinematics Positioning Using Nonlinear Programming for Highly Articulated Figures.

[8] Patrick Beeson, Barrett Ames, TRAC-IK: An Open-Source Library for Improved Solving of Generic Inverse Kinematics

[9] Krupa, A., Gangloff, J., Doignon, C., De Mathelin, M.F., Morel, G., Leroy, J., Soler, L. and Marescaux, J., 2003. Autonomous 3-D positioning of surgical instruments in robotized laparoscopic surgery using visual servoing. IEEE transactions on robotics and automation, 19(5), pp.842-853.

[10] Heim, A. and Von Stryk, O., 2000. Trajectory optimization of industrial robots with application to computer-aided robotics and robot controllers. Optimization, 47(3-4), pp.407-420.

[11] Venator, E., Lee, G.S. and Newman, W., 2013, August. Hardware and software architecture of ABBY: An industrial mobile manipulator. In Automation Science and Engineering (CASE), 2013 IEEE International Conference on (pp. 324-329). IEEE.