Haptic Virtual Fixture for Teleoperation of a CNC Welding Machine

Jaffer Ali　　Aidan Rosenbaum

***Abstract*—This paper implements a haptic virtual fixture for a teleoperated CNC welding machine. A 2-degree-of-freedom pantograph device is used to translate user position to machine movement. The pantograph applies kinesthetic force feedback to maintain the users cursor position along a predefined virtual fixture. Two optical encoders are used to track user position while two DC motors are used to apply force feedback. Sensors and actuators are controlled through a single Arduino which communicates with the CNC machine over serial. The results of our paper demonstrate the potential effectiveness of virtual fixtures for teleoperated robotics.**

I. Introduction

Integrating force feedback in teleoperated robotic systems provides important information to the operator when performing certain tasks. For example, teleoperated surgical robots implement force feedback systems to ensure that operators do not apply excessive force in certain areas. Traditional force feedback systems uses force sensors to detect the robots contact with objects and provide feedback to the user. This feedback can improve operator performance but does not prevent an operator from moving into certain areas completely. Virtual fixtures are one solution that can help solve this problem. A virtual fixture renders haptic feedback to ensure that a user stays within certain bounds or is following a predefined path. For robotic surgery, certain areas may be restricted and if the operator attempts to move into them they are met with a force feedback pushing them away. For something like teleoperated robotic welding, a virtual fixture can ensure that the operator is welding along the correct path. In this paper we use an open source 2-degree-of-freedom pantograph device called the Graphkit to teleoperate a CNC machine. Using forward and inverse kinematics along with the Jacobian we were able to teleoperate a 3-axis CNC Welding machine in 2D space and render a virtual fixture of a potential weld path.

II. Background

Our pantograph is based off of the 2-degree-of-freedom Graphkit developed in [**1**]. Previous work has been done with different types of virtual fixtures in robotic surgery applications [**2**].

III. Methods

A. Hardware Design and Implementation

The 2-degree-of-freedom Graphkit was built using 2 1-degree-of-freedom hapkits. The pantograph is a 5 bar linkage with link lengths a_1 = 5, a_2 = 6, a_3 = 6, a_4 = 5, a_5 = 2.5. The linkage drives two capstan drives which are connected to corresponding motors and optical encoders which provide the force feedback and rotary position data respectively. An Arduino 101 is used to calculate xy-positions of the pantograph end effector using forward kinematics. The Arduino calculates the necessary forces using the Jacobian and provides force feedback to the DC motors using pulse-width-modulation. To connect to the CNC welding machine, the Arduino writes xy-positions over serial to a python script on a separate laptop. The python script scales the position data and converts the position data to G-Code which is sent over serial to the CNC machine. For live demonstration purposes, a pen was used instead of a welder as the end effector on the CNC machine.

B. Control

Forward kinematics were implemented to derive the end-effector position (x_3, y_3) given angles θ and α from the Hapkit encoders [**3**]. The values of these equations were then tested against a Solidworks model of our system. An image of the geometry of our pantograph robot is shown below:

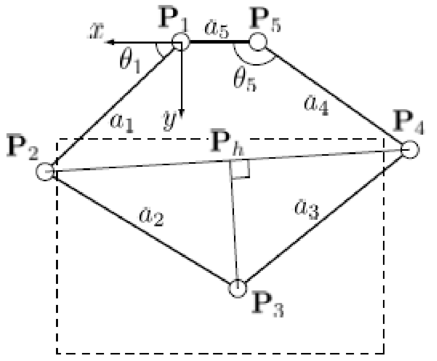

Figure 1: Pantograph Model

The corresponds forward kinematic equations are given below:

$$P_2 = (x_2, y_2) = (a_1 cos(\theta), a_1 sin(\theta)) \tag{1}$$

$$P_4 = (x_4, y_4) = (a_4 cos(\alpha), a_4 sin(\alpha)) \tag{2}$$

$$P_{24} = \|P_4 - P_2\| \tag{3}$$

$$P_{2h} = \frac{(a_2^2 - a_3^2 + P_{24}^2)}{2P_{24}} \tag{4}$$

$$P_{3h} = \|L_2 - P_{2h}\| \tag{5}$$

$$x_h = x_2 + (\frac{P_{2h}}{P_{24}})(x_4 - x_2) \tag{6}$$

$$y_h = y_2 + (\frac{P_{2h}}{P_{24}})(y_4 - y_2) \tag{7}$$

$$x_3 = x_h + (\frac{P_{3h}}{P_{24}})(y_4 - y_2) \tag{8}$$

$$y_3 = y_h - (\frac{P_{3h}}{P_{24}})(x_4 - x_2) \tag{9}$$

We then calculated the Jacobian to translate reaction forces in the xy-space to our motors.

$$\tau = J^T F \tag{10}$$

Supplemental to these calculations, we also derived the velocity every 30ms with a filter. From experimentation, we decided a 90/10 filter would work best.

On the machine-side, position scaling was also implemented to increase the available range of our system:

$$x_{machine} = k_{scale} * (x_3 - x_0) \tag{11}$$

$$y_{machine} = k_{scale} * (y_3 - y_0) \tag{12}$$

Where k_{scale} is the scaling factor, and (x_0, y_0) is the zero position.

$$v_x = \frac{dx}{dt} = \frac{(x - x_{last})}{.030} \tag{13}$$

$$v_y = \frac{dy}{dt} = \frac{(y - y_{last})}{.030} \tag{14}$$

$$v_{filtered} = v * .9 + v_{last} * .1 \tag{15}$$

Where v_{last}, x_{last}, and y_{last} are the previous sampled values in the loop.

Using these velocity values, we are able to apply a damping force to the x and y axis of the machine. During forcefielding, this damping is removed.

$$F_x = b * v_x \tag{16}$$

$$F_y = b * v_y \tag{17}$$

Where b is the damping coefficient of the machine.

Position clutching was also implemented to increase the available workspace of the machine. This was done by resetting (x_0, y_0) to the current position once the user releases the clutching button.

C. Virtual Fixture Rendering

The virtual fixture that we rendered was a horizontal line. We defined the line with two xy coordinate pairs in the pantograph workspace. We calculated the shortest distance from the line to the current end effector position using Equation 18 which allowed us to enable the virtual fixture when the user got close to the line. If the user was far enough away from the line, $F_x = 0$ and $F_y = 0$. The constants a, b and c are coefficients for the given line equation. The force applied to the user was given by Equations 19 and 20 where k was a constant multiplied by the difference in position between the current end effector position and the closest point on the line.

$$d = \frac{|ax_0 + by_0 + c|}{\sqrt{a^2 + b^2}} \tag{18}$$

$$x_{closest} = \frac{b(bx_0 - ay_0) - ac}{a^2 + b^2} \tag{19}$$

$$y_{closest} = \frac{a(-bx_0 + ay_0) - bc}{a^2 + b^2} \tag{20}$$

$$F_x = -k(x_{current} - x_{closest}) \tag{21}$$

$$F_y = -k(y_{current} - y_{closest}) \tag{22}$$

IV. Results and Discussion

We were able to successfully implement position tracking as well as virtual fixturing. In implementing position damping, we ran into limitations with the processing capabilities of our hardware, resulting in instability at damping levels adequate for our machine. A plot of our full workspace as measured on the machine is in Figure [**2**]. We found a scale of 10 was the limit before issues with quantization occurred.

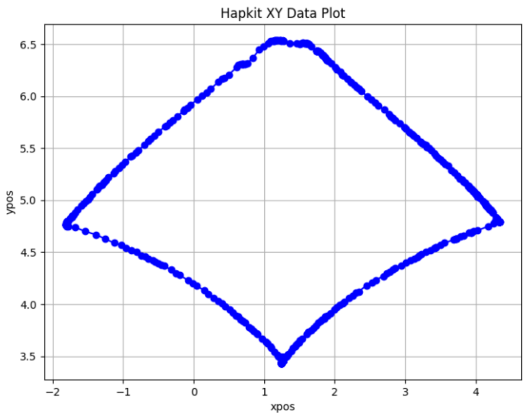

Figure 2: The full workspace measured

Users were able to distinguish the virtual fixture in the environment, and often mistaken it for an actual physical limitation of the device. a rendered plot of our virtual fixture is shown in Figure [**3**].

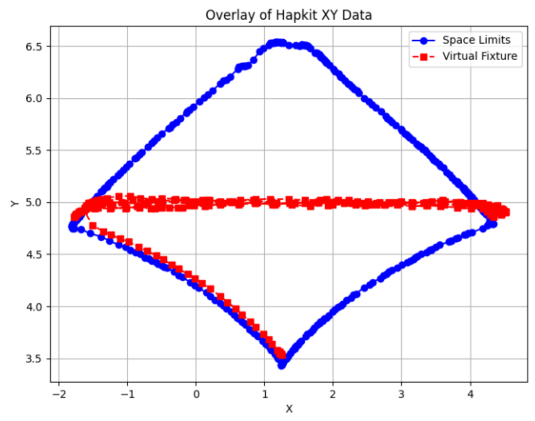

Figure 3: Rendered virtual environment, virtual wall.

In creating this device, we also set out to implement bilateral position coupling with the machine. We were unable to implement this however because of limitations of the CNC software. The GRBL interface could only support communicating up to 10 times per second, which was not adequate enough for position coupling.

The nature of CNC-machine control also meant that new target positions would be added into a stack of previous target positions. This created problems when the user would move at high speeds, as the machine would move to previous target points before reaching the current position.

Position clutching was also implemented to increase the available workspace. Since the control of it was limited to the keyboard, it was not included in the demo.

Our GitHub for this project can be found here, and a Youtube video of it can be found here.

V. Conclusion

In this work we present a 2-dof Hapkit device that is used to control a CNC-welder. We implemented force-fielding to assist in seam-awareness, as well as clutching, position scaling, and damping. All resulted in stable control except for damping, which introduced instability to our system. Our interactive demos showed that the force-fielding produced a convincing effect for the user, however working with the limitations of serial and machine-communication limited our ability to effectively control the machine.

References

[1] M. O. Martinez, J. Campion, T. Gholami, M. K. Rittikaidachar, A. C. Barron, and A. M. Okamura, "Open source, modular, customizable, 3-d printed kinesthetic haptic devices," in IEEE World Haptics Conference. IEEE, 2017, pp. 142–147.

[2] J. Abbott, P. Marayong, and A. Okamura, "Haptic Virtual Fixtures for Robot-Assisted Manipulation." Accessed: Apr. 29, 2024. [Online]. https://www.telerobotics.utah.edu/uploads/Main/Abbott_RoboticsResearch07.pdf.

[3] "Design and Control of a Pantograph Robot." Design and Control of a Pantograph Robot - Northwestern Mechatronics Wiki, 17 July 2010, hades.mech.northwestern.edu/index.php/Design_and_Control_of_a_Pantograph_Robot.