Variable Force Control On 7dof Robotic Sanding Toolpaths

Niculae Mihai [1], Ashraf Uddin Chowdhury Rafat[2]
[1]Technoaccord Inc.; mniculae@technoaccord.com
[2]Concordia University; aurafat23@gmail.com

Abstract: This work introduces a sophisticated robotic sanding technique that combines database-driven toolpath generation, adaptive force modulation, and impedance-based force control for accurate 3D surface sanding. Variable robot sanding toolpaths are calculated using a STP 3D model file, and real-time force adjustments guarantee uniform sanding quality on intricate surfaces. To maximize precision and effectiveness, the system uses impedance control to dynamically adjust contact pressure. Sanding is carried out by a seven degree of freedom (DOF) robotic arm that is outfitted with specialized tools and a force-torque sensor. The technique maintains optimal force application, especially in edge and frame sanding tasks, by combining inverse kinematics, Lagrange equations, and real-time feedback. Over-sanding at edges is avoided by using a sigmoid smoothing function, which guarantees smooth force transitions along the toolpath. The system demonstrates the potential of fusing data-driven precision with sophisticated robotic control for industrial applications by achieving robust and consistent surface finishing through the integration of mathematical modeling, sensor feedback, and adaptive control.
Keywords: robot; sanding; toolpath; force control; kinematics

1. Introduction

Automatic surface finishing, especially 3D surfaces, automatically, with the help of robots, regardless of the material the part is made of, is a challenge in the world of robotics. After much experimental research, we discovered that without automatic control of the pressing force on the contact surface, this is not possible. But this force must always be perpendicular to the surface, therefore both the forces on each axis Fx, Fy and Fz, as well as the torque around each axis Tx, Ty, Tz, must be taken into account. Controlling these variables on spatial trajectories is quite difficult because surfaces are complex. We need a good definition of them, a precise mathematical modeling to be able to calculate the trajectories of the finishing tools, then a rigorous control of the variations of the forces and torques, correlated with the speed and accelerations of movement, in order to obtain uniformly finished surfaces.

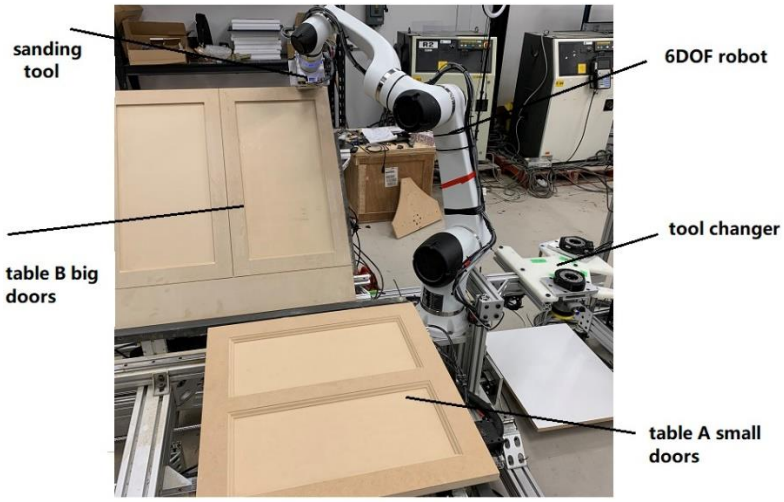

Fig 1 Description of experimental robotic cell

For a better finish we need to create special tools for the robot arm, with precisely determined weight, for a better optimization of forces and torques on spatial trajectories. These special tools are 4 in number and are adapted for specific surfaces: front, sides, edges, three-dimensional.(see fig.2)

Fig 2. Robotic special tools

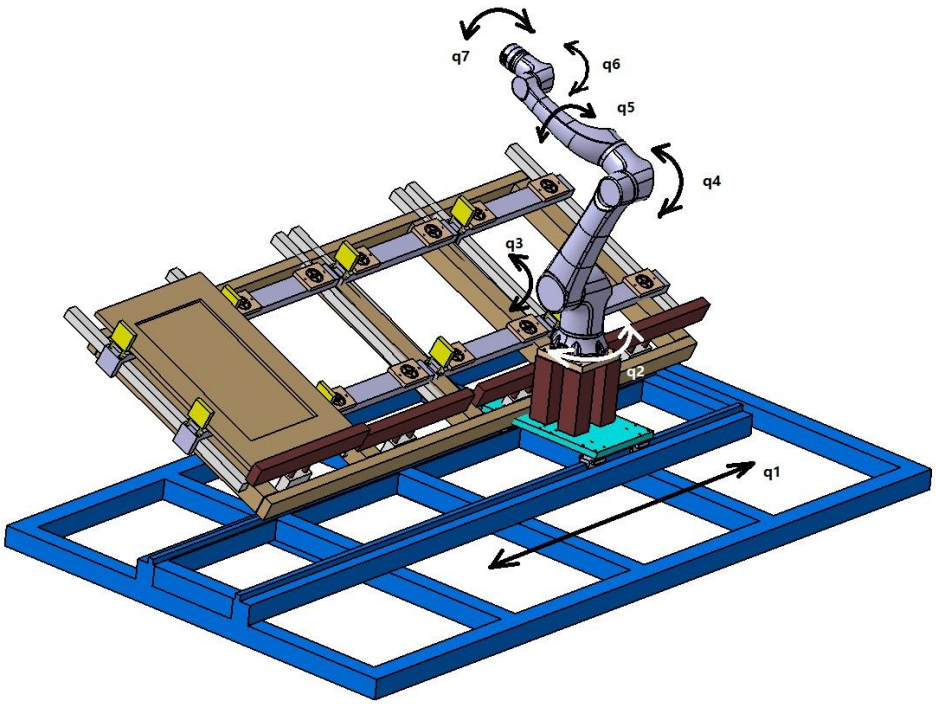

Fig. 3. Kinematic diagram of robotic cell

For grinding tasks where constant pressure is essential, a combination of both methods is useful:

Step 1: Use the DH parameters to calculate the initial inverse kinematics and basic positioning of the robot.

Step 2: Integrates the Lagrange equations to adjust the robot's movements in real-time based on feedback from the force sensor, ensuring that constant and optimal pressure is applied to the surfaces to be sanded.

To calculate the Lagrangian and derive the Lagrange equations for a robot with 6 rotational degrees of freedom (DOF) and an additional translational axis at the base (fig.3), we will follow a systematic approach using classical mechanics. The Lagrangian L is defined as L=T−V, where T is the kinetic energy of the system, and V is the potential energy of the system.

2. Determine the position and orientation

The robot consists of 7 joints:

- q: Translation along the base axis.
- q2,q3,q4,q5,q6,q7: Rotation of each of the 6 subsequent robot joints

The overall transformation matrix Tr from the base to the end-effector frame will be:

Tr=A1*A2*A3*A4*A5*A6*A7

3. Determine Lagrange equation for the robot

a. Kinetic energy T

- **translational kinetic energy:**

$T_{trans} = \frac{1}{2}mv^2$, where m is the mass of the link and v is the velocity of the center of mass of the link;

- **rotational kinetic energy:**
  $T_{rot} = \frac{1}{2}\omega^T I \omega$, where omega is the angular velocity vector and I is the inertia matrix of the link;

b. Potential energy

V=mgh, where h is the height of the center of mass of the link above reference level;

The Lagrangian is given by L=T-V, and Lagrange equations of motion are derivated from the principle of least action:

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}_i}\right) - \frac{\partial L}{\partial q_i} = Q_i$$

,

where *qi* represents each generalized coordinate, and *dqi* is the derivative of *qi* with respect to time, and Qi represents non-conservative forces/torques.

4. Robot Sanding Toolpath:

Our robot sanding code demonstrates a rigorous method of gathering data from a relational database in addition to the creation of exact geometric paths for robotic operations. Following a 2D scan and interpretation of a 3D STP file of doors, the data gathered in coordinate form is stored in an SQLite database. Specifically, the system gets coordinates for a set of predefined points from this database, separates these points into individual "pockets," and then computes offset boundaries and a zigzag path for the selected pockets. Our program begins by importing the necessary modules:

- A module providing the function get_points_coordinates_as_arrays, which is in charge of communication with the SQLite database.
- The standard library Python module used for numerical mathematics, in this instance to calculate a dynamic step size of the zigzag pattern.

A list of point labels is declared, with each label referencing a database entry. When function get_points_coordinates_as_arrays(db_path, point_labels) is invoked, it returns a dictionary where each label maps to its associated coordinate triplet. This dictionary is then used throughout the program for diagnostic reporting purposes as well as path computation.

With it receiving coordinate dictionary back, the code cycles through the initial list of point labels and prints out each corresponding value. This validation process guarantees that point data required has been properly captured and facilitates real-time data integrity checks. Four pockets are defined as logical groups of points. Each pocket is defined by a set of four point labels, thus defining a rectangular or polygonal boundary. In manufacturing terminology, these areas might be interpreted as locations on an item work where machining, milling, or sanding activities take place. Parameters for the tool are established. Tool offsets position the limit in or out for purposes of having clearance space or to adapt for the real geometry of a cutter or a sanding medium. A margin as a user-supplied internal offset is used within the bounds in addition to these. The inner sanding offset defines the control step size to control spacing between passes along the zigzag path. Frame offsets are employed to adjust interior vertical size of the pocket, typically for hardware or fixture constraints.

For boundary point retrieval for each pocket, the code retrieves valid coordinates from the dictionary. These coordinates are also printed to reflect the exact edge of each pocket, acting as

secondary verification of space data and corner or edge information that delineate each pocket region. Secondarily, for pockets that are set to continue a zigzag (or serpentine) pattern, our code also calculates an offset boundary. The corner points both use the tool offsets as well as the inner offset to ensure that the resulting toolpath is within the specified machining area and provides enough clearance for the tool. The program next determines the vertical range between two corners, then subtracts the frame offsets to compute the usable interior height to construct a zigzag path (Fig.4). If yinner > 0, the algorithm calculates how many passes must be made. The program also constructs a list of row endings, reversing their order by the use of a "toggle" variable to create an old-fashioned back-and-forth motion. On even-numbered rows, the instrument moves from left to right; on odd-numbered rows, the motion is reversed to fill the area in its last areas. As the zigzag path is constructed, the endpoints of each row are added to a master list. Lastly, these calculated coordinates are output, giving a clear graphical depiction of the planned toolpath for sanding.

This method streamlines boundary detection, offsetting, and zigzag path creation, all of which are important to precise toolpath construction. Its dynamic calculation of overlapping steps makes it capable of adjusting to pockets of any size, thus optimizing efficiency and accuracy. By merging careful database queries with geometric transformation and strict path creation, our code shows a straightforward pipeline in robotic sanding. The output provides valuable insights regarding the robot motion while sanding, and how to compute the offsets for implementing the zigzag path talks about flexibility that is characteristic of real-robot sanding operations. The flow chart visualizes robot sanding tool path whole algorithm (Fig. 5). the overall process for the Generally speaking, this automated sanding robot path solution highlights the significance of merging data handling, mathematical precision, and algorithmic innovation in developing robust robot sanding systems.

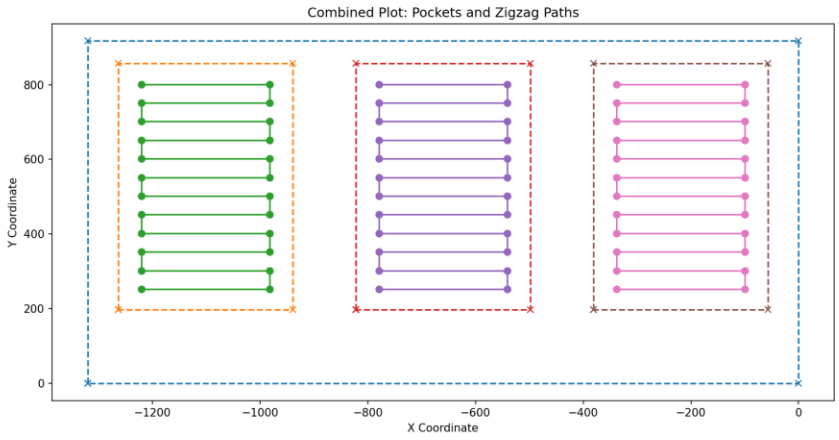

Fig 4: Robot Sanding Toolpath

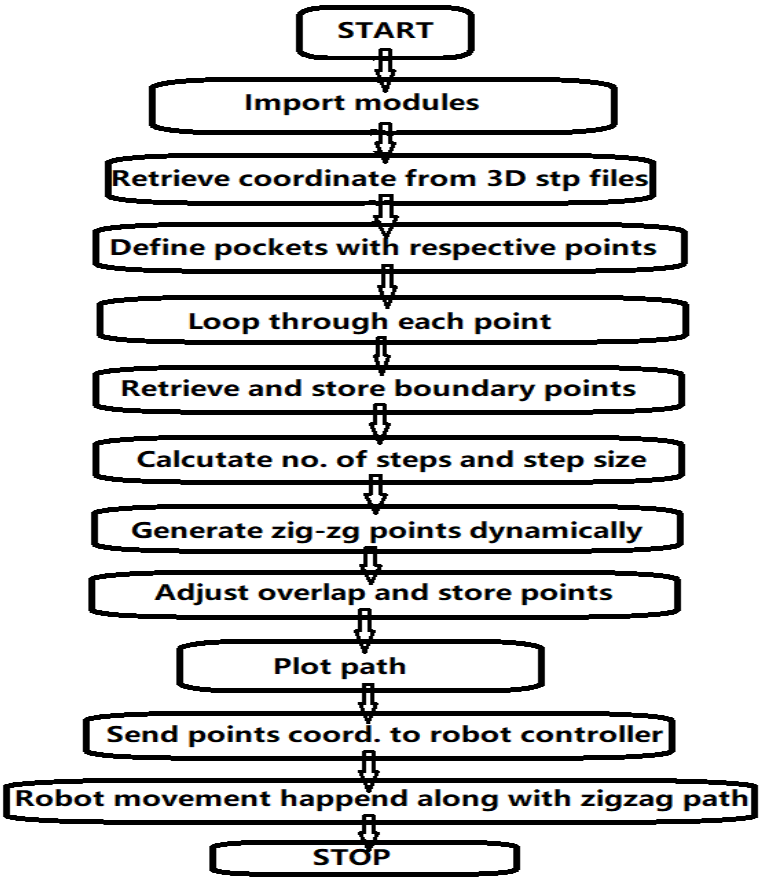

Fig 5: Flow chart for robot sanding toolpath

5. Force Control at the time of Sanding:

Generally, for the sanding operation our robot work in highly contact-intensive operations that involved the highly precise regulation of applied forces in order to deal with workpiece shape irregularities, changing tool condition, and varying material properties. This alone puts the robot at a disadvantage given the compliant interventions and to-ensure even applications of pressure can not be ensured in unpredictable environments where sudden changes in parameters like position and orientation occur suddenly and thus needs to be modified according to their need. The impedance-control architecture addresses this issue, defining how the robot end effector needs to dynamically interact with its surroundings [5]. Through the independent control of forces along the Cartesian coordinate-axis (x, y, and z), this method ensures consistency of the normal force (z-axis) used for material removal while minimizing any unexpected lateral loads (x/y-axis) resulting from misalignment.

Our Robot has a mass spring damper system. The external force Fext generally relates here the position and velociry deviations [6].

$$Fext= Md(x-xd)+ Bd(x-xd)+Kd\ (x-xd)$$

where, Fext is the external force measured having three components (Fx, Fy, Fz) and Md, Bd, and Kd denote matrices that are diagonal and intended to give the desired inertia, damping, and stiffness along each axis. As well as, x and xd are the position of the end-effector at the current

and desired positions, respectively. By careful choice of Kd and Bd, for each axis, the controller can be tuned in a way that Z-axis is relatively high stiffness at its limits-due to keep a stable contact force about 1N-15 N for sanding. For X and Y are lower stiffness is used in the lateral directions so that greater compliance allows to soften lateral friction and therefore avoids the tool from skidding. A blend of impedance control and motion planning is used. For Z-axis the focus is basically on a force control system to ensure the specified normal pressure exerted on the work surface is maintained. The x and y axes are on the other hand focused on accurately following a path, using impedance with lateral-less disturbance characteristics. These force control is essential for the zigzag, outer edge sanding. With the wrist-mounted six-degree-of-freedom force-torque sensor, there is a continuous measurement of the interaction forces. This data are combined with joint torque feedback, enhancing the noise tolerance of the controller and enabling it to compensate automatically for dynamic uncertainties [6].

For the side edge sanding of the frames requires precise force for not only the normal direction z axis but also the direction of the x and y axis. Here our robot used impedance control to regulate forces in all the three cartesian axes with the adaptive edge tracking and uniform sanding for all of the surface. For the edge sanding the impedance control law is migrated into the specific dynamic axis.

$$Fext = \begin{bmatrix} Kdx & 0 & 0 \\ 0 & Kdy & 0 \\ 0 & 0 & Kdz \end{bmatrix}(x - xd) + \begin{bmatrix} Bdx & 0 & 0 \\ 0 & Bdy & 0 \\ 0 & 0 & Bdz \end{bmatrix}(x - xd)$$

where, Fext is the external force vector with components Fx,Fy and Fz, Kd is the stiffness matrix(Kdx,Kdy and Kdz are the elements for the x,y,z),Bd is the damping matrix(Bdx,Bdy and Bdz elements for the x,y and z axis,x is the current position of the effector and xd is the desired position of the effector.Here, for the z axis high stiffness ensures steady normal for at the time of sanding. At x axis and y axis lower stiffness allows to follow the edges for sanding while maintain controlled force (Fig.6). In the process of inner frame sanding, the tangential force of friction can be, approximately, given as

$$F_{friction} = \mu F_z,$$

where, μ is the friction coefficient between the abrasive surface and the workpiece. To solve this disturbance, the controller corrects the desired lateral force (Fd,xy) thereby compensating for the deviations in x and y induced by friction. For the inner side frame sanding our robot is programmed with multi axis force regulation. The sanding tool's TCP is aligned with our user co-ordinate system by using our program. So, we can say that active force control in the x and y axes for the inner frame sanding operation ensures the robustness of sanding. This approach is also essential for the door frame finishing and door edge processing.

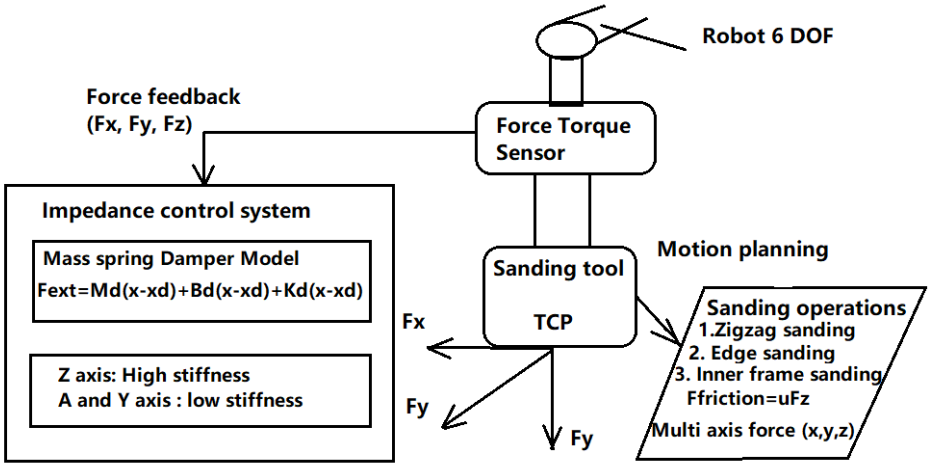

Fig 6: Force Control at the time of Sanding

6. Variable Force Control on Toolpath:

In this approach, the force along the toolpath is minimized near the lateral boundaries and held at a constant level elsewhere. To achieve the desired force distribution, a mathematical model is used to define how the force profile varies with the position along the toolpath. Let s denote the position along the path. The desired force, Fd is then expressed as a function of s. Consequently, the force profile can be formulated as follows:

$$\mathrm{Fd}(s) = \begin{cases} \mathrm{F\,low} & \text{if } s \in \text{side edges}, \\ \mathrm{Fconst} & \text{otherwise}, \end{cases}$$

In this formulation, Flow represents the reduced force applied at the lateral boundaries, while Fconst indicates the uniform force level maintained throughout the remaining segments. Moreover, to ensure a seamless transition from one boundary to the softer regions thus avoiding abrupt shifts in the force profile a method known as the sigmoid smoothing function is employed [4].

$$\mathrm{Fd}(s)=\mathrm{Fconst}-(\mathrm{Fconst}-\mathrm{Flow})\ *\sigma(s;s1,s2)$$

where, σ(s;s1,s2) is a sigmoid function that transitions smoothly between 0 and 1. s1 and s2 define the start and end of the side edge regions. sigmoid function is defined as

$$\sigma(s;s1,s2) = \frac{1}{1+e^{-k(s-s1)}} \times \frac{1}{1+e^{k(s-s2)}}$$

Here, the parameter k governs how sharply the transition occurs. The desired force, Fd is subsequently incorporated into the control law, with the control input u reflecting this relationship [1].

$$u = j^T(SFd(s)n + (I - S)Kp(Xd - X)$$

Here, s denotes the selection matrix for the force-controlled directions, n is the surface normal vector, and Kp is the position gain matrix. The variables xd and x represent the desired and actual positions, respectively. Along the lateral edges, the desired force is set to Flow. Consequently, if excessive sanding is detected, the robot can decrease the applied force. This adjustment can be realized by lowering the stiffness Kd and employing an adaptive control strategy that modulates the force based on sensor feedback [2].

The force Fd(s) is determined in the constant-force regions by maintaining a force Fconst. The robot maintains this force by controlling the position error e=x−xd, and by compensating for external disturbances like surface variations. If, the side edge regions are not described, the robot can dynamically adjust the force depending on real-time sensor feedback. It uses a force threshold to utilize real-time feedback from the sensors to bypass the transmission feature dealing with the detection of the side edges and thus lowering the force when the measured force Fext exceeds a safe limit. Here for the dynamic force adjustment, Reduce Fd(s) if Fext>Fthreshold [3]. Here, the toolpath is parameterized as xd(s). First we identified the side edge regions and constant force regions. After that we used a sigmoid function for the smooth transition between Flow and Fconst. In the loop,we measured the external force Fext, computed the desired force Fd(s). At last, applied the control law to achieve Fd(s). Also, the robot used sensor feedback to dynamically adjust the force in real time. By the implementation of this mathematical represented program, our robot achieved precise force control along the toolpath and it ensures consistent sanding quality without doing more sanding at the side edges.In the constant-force regions, Fd(s) is specified to remain at Fconst. The robot upholds this force by regulating the position error, e=x−xd and compensating for external disturbances, such as variations in the work surface. If the boundaries along the toolpath are not explicitly defined, the robot can adapt the applied force in real time by referencing sensor data. Through a predefined force threshold, the control system can circumvent the transmission functionality for edge detection and instead reduce the force whenever the measured external force Fext surpasses a safe limit. Concretely, for real-time force adjustment, the desired force Fd(s) is lowered if Fext>Fthreshold.The toolpath is parametrized as xd(s). First, the side edge sections and constant-force sections are identified. A sigmoid function is then introduced to provide a smooth shift between Flow and Fconst. During each iteration of the control loop, the external force Fext is measured, the desired force Fd(s) is calculated, and the control law is executed to achieve Fd(s) (Fig.7). Additionally, the robot uses sensor feedback to adjust the force in real time. By implementing this mathematically formulated strategy, the robot maintains accurate force control along the toolpath and ensures uniform sanding quality without over-sanding at the edges.

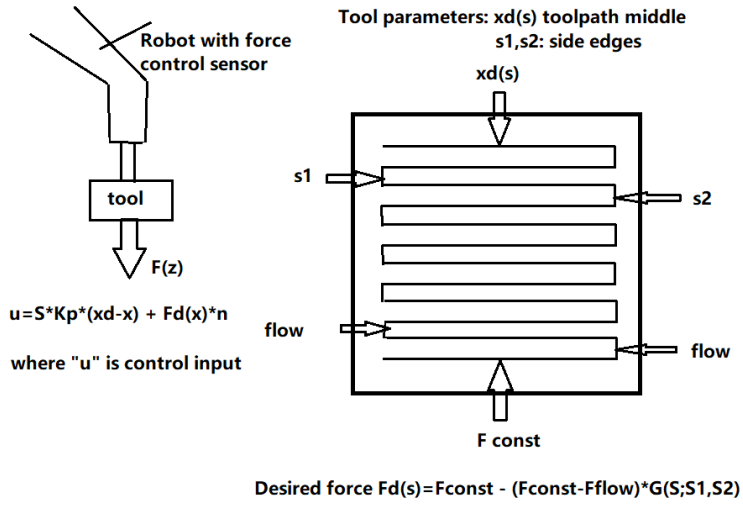

Fig 7: Variable force Control on toolpath

References:

1. Raibert, M. H. and Craig, J. J. "Hybrid Position/Force Control of Manipulators."ASME Journal of Dynamic Systems, Measurement, and Control, vol. 103, no. 2, pp. 126–133.

2. Hogan, N."Impedance Control: An Approach to Manipulation."ASME Journal of Dynamic Systems, Measurement, and Control, vol. 107, no. 1, pp. 1–24

3.Starr, G. P."Sensor-Based Robotic Deburring and Finishing Methods."IEEE Control Systems Magazine, vol. 8, no. 4, pp. 33–38.

4.Gasparetto, A. and Zanotto, V."A new method for smooth trajectory planning of robot manipulators."Mechanism and Machine Theory, vol. 42, no. 4, pp. 455–471.

5. N. Hogan, "Impedance Control: An Approach to Manipulation," *Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, pp. 1–24.

6. Han's Robot, "Elfin-Pro Collaborative Robot: Technical Manual," 2020.