A Bevel-Tip Needle Path Tracking: A Model Predictive Approach

Haitao Jiang, Lambert Ren,

***Abstract*—This paper focuses on a real-world issue that arises in medical scenarios. Specifically, we explore the application of model predictive control (MPC) to the problem of tracking the path of a bevel-tip needle model. Through simulations on different tracking paths, we compare the performance of the MPC controller with various parameter sets. To demonstrate the stability of the MPC controller, we conduct a stability analysis of the model. Finally, we present a 3D simulation created by Python that further proves the controller can be used to achieve our desired outcome.**

I. Introduction

Percutaneous punctures are minimally invasive surgical procedures that are commonly utilized for diagnosing and treating lesions located in hard-to-reach areas. They are frequently employed for a variety of medical purposes, such as biopsy, injection, radiofrequency ablation, drug delivery, and more [1]. Despite the benefits, there are several factors that can cause the needle tip to deviate from its intended target. These factors may include tissue inhomogeneity, tissue deformation, as well as the surgeon's hand-eye incoordination and fatigue [2]. To enhance the controllability of puncture needles, a new approach involving the use of a bevel-tip flexible needle has been proposed [3]. The accuracy is improved via precise planning and controlling due to the flexibility and controllable of the bevel-tip needle. The common approach of controlling the bevel-tip needle is the PID controller [4]. However, the use of this type of controller may result in limitations to the accuracy of the surgical procedure.

This paper introduces a novel approach, a model predictive control (MPC) system, to control the bevel-tip needle during surgical procedures. This system is designed to enable the needle to accurately follow predetermined paths.

A. Bevel-tip needle dynamics

As stated by Huo [5], the bevel-tip needle can be characterized as a nonlinear dynamic system, with the following equations governing the motion:

$$\mathbf{q_{m+1}} = \mathbf{q_m} + \dot{\mathbf{q}} * \triangle dt \tag{1}$$

$$\dot{\mathbf{q}} = \mathbf{A}(\mathbf{q}) \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{2}$$

$$\mathbf{q} = \begin{bmatrix} x & y & z & \alpha & \beta & \gamma \end{bmatrix}^T \tag{3}$$

Haitao Jiang and Lambert Ren are master students at TU Delft, The Netherlands. E-mail addresses: {n.surname1, z.ren-4}@student.tudelft.nl.

$$\mathbf{V^b} = v\mathbf{V_1} + \omega\mathbf{V_2} \tag{4}$$

$$\mathbf{V^b} = \mathbf{J}\mathbf{V^w} \tag{5}$$

As shown in Fig. 1, the state of the needle is represented by a vector $\mathbf{q}$, which includes the position and attitude of its tip. Specifically, $\mathbf{q}$ is composed of the coordinates x, y, z in 3D space, as well as the attitude angles α, β, γ. The needle's motion is determined by an insertion speed v, as well as a rotational speed ω, which corresponds to the rotation around the z-axis.

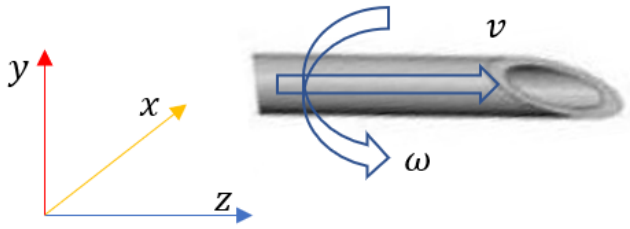

Fig. 1. Dynamics model of the bevel-tip needle.

Moreover, in the case of the bevel-tip needle model, there is typically a parameter, curvature, that also determines the path followed by the needle tip, as shown in Fig. 2. With the help of curvature and rotation speed, the tip can rotate further flexibly. As a part of the shape of the needle, curvature is generally set to a specific number initially.

Assuming that the needle tip is moving in its own body coordinate system, its velocity can be represented as $\mathbf{V^b}$ in Equation 4. Where,

$$\mathbf{V_1} = \begin{bmatrix} 0 & 0 & 1 & \kappa & 0 & 0 \end{bmatrix}^T \tag{6}$$

$$\mathbf{V_2} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^T. \tag{7}$$

The translation between the velocity of the needle tip in the world coordinate system and itself body coordinate system can be described in the Equation 5. And this translation matrix is $\mathbf{J} = \begin{bmatrix} \mathbf{R}^T & 0_{3\times3} \\ 0_{3\times3} & \mathbf{R}_\Omega \end{bmatrix}$. $\mathbf{R}^T$ is the rotation matrix between the world coordinate system and needle body coordinate system, $\mathbf{R}_\Omega$ is the coefficient matrix of Euler kinematics equations of rigid body motion [6].

Therefore, integrating all the formulas and parameters given previously, we can finally reach to the kinematics model:

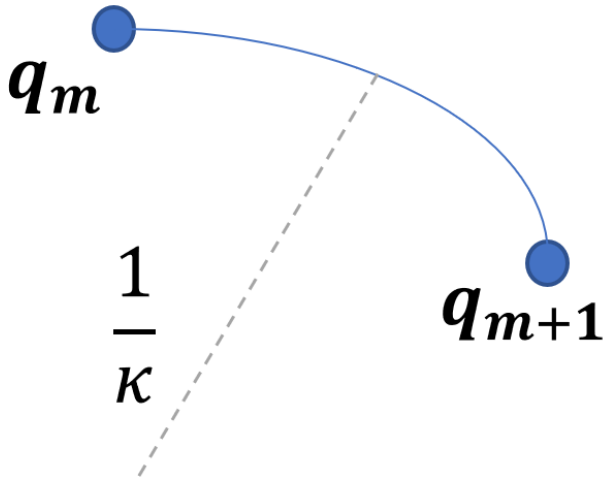

Fig. 2. The curvature of the arc followed by bevel-tip needle.

$$\dot{\mathbf{q}} = \mathbf{A}(\mathbf{q}) \begin{bmatrix} v \\ \omega \end{bmatrix} = \begin{bmatrix} sin\beta & 0 \\ -cos\beta sin\alpha & 0 \\ cos\alpha cos\beta & 0 \\ \kappa cos\gamma sec\beta & 0 \\ \kappa sin\gamma & 0 \\ -\kappa cos\gamma tan\beta & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \tag{8}$$

B. Discretized system dynamics

To discretize the dynamic model, we need to derive the equation based on two assumptions. First, we assume that the needle is stiffness in axial direction. Therefore, the needle tip will rotate exactly the same as the needle tail, where the power of rotation generates. We also assume that the needle is a rigid body model. In this case, the needle body moves exactly according to the trajectory traveled by the needle tip. Under these two assumptions, we can only take the needle tip as the robot model in our MPC system and only track the position and orientation of the needle tip during the experiment.

As shown in Equation 8, the bevel-tip needle model is a nonlinear model and the model states at different state is strongly related to the input v and ω. Considering that our task is to tracking a given reference path and either angle of the orientation will have a large influence on the final path of needle tip, we will not linearize our dynamic model. Therefore, we simply discretize the needle model. and, the discrete needle model derived from Equation 8 is given as:

$$x_p(m+1) = x_p(m) + sin(\beta(m))v(m) * dt, \tag{9}$$

$$y_p(m+1) = y_p(m) - cos(\beta(m))sin(\alpha(m))v(m) * dt, \tag{10}$$

$$z_p(m+1) = z_p(m) + (cos(\alpha(m))cos(\beta(m)))v(m) * dt, \tag{11}$$

$$\alpha(m+1) = \alpha(m) + (\kappa cos(\gamma(m))/cos(\beta(m)))v(m) * dt, \tag{12}$$

$$\beta(m+1) = \beta(m) + (\kappa sin(\gamma(m)))v(m) * dt, \tag{13}$$

$$\gamma(m+1) = \gamma(m) - \kappa cos(\gamma(m))tan(\beta(m))v(m) * dt + \theta, \tag{14}$$

$$v(m+1) = v(m) + a * dt, \tag{15}$$

where x_p, y_p, z_p are the needle tip's position in the world coordinate system relative to the beginning point of the robot model, α, β, γ are the needle tip's total rotation angle in three orientation in the world coordinate system accumulated from the initial point, a is the acceleration of the axial velocity, θ is the rotation angle with respect to the z-axis every time step, κ is the curvature of the needle model and dt is the time step between two states in time series.

C. System dynamics

After the discretization performed in the previous section, the system can be transformed into the nonlinear time invariant system:

$$x_{m+1} = f(x, u) = \mathbf{A}x + \mathbf{B}u, \tag{16}$$

where the state in every time step can be described by

$$x = \begin{bmatrix} x_p & y_p & z_p & \alpha & \beta & \gamma & v \end{bmatrix}^T, \tag{17}$$

and the input to the state space system can be described by

$$u = \begin{bmatrix} a & \theta \end{bmatrix}^T. \tag{18}$$

Moreover, the sparse matrix used in the state space model is given by:

$$\mathbf{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & sin(\beta) * dt \\ 0 & 1 & 0 & 0 & 0 & 0 & -cos(\beta)sin(\alpha) * dt \\ 0 & 0 & 1 & 0 & 0 & 0 & cos(\alpha)cos(\beta) * dt \\ 0 & 0 & 0 & 1 & 0 & 0 & \kappa cos(\gamma)/cos(\beta) * dt \\ 0 & 0 & 0 & 0 & 1 & 0 & \kappa sin(\gamma) * dt \\ 0 & 0 & 0 & 0 & 0 & 1 & -\kappa cos(\gamma)tan(\beta) * dt \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{19}$$

$$\mathbf{B} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & dt \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^T \tag{20}$$

A non-linear model predictive controller for purpose of the path tracking will be developed based on this discretized system dynamic.

II. Model Predictive Control Design

In this section, we will introduce a model predictive controller (MPC) designed to control the motion of the bevel-tip needle model based on the system dynamics model presented in Equation 16. Moreover, the MPC controller varies depends on setting the different scenarios when setting the curvature κ and the state input to controller:

(i) When setting the curvature $\kappa = 0$, the rotation angle θ actually makes no difference to the general motion of the needle tip if we just treat it as a mass point, the needle will only rotate around it self. Therefore, the input angel θ to the controller can be set to 0 when in this situation. According to the Equation 16, when setting $k = 0$ and $\theta = 0$, and $\alpha, \beta, \gamma = 0$, the system dynamics will be much more simplified, and the system can be treated as

a linear model. Therefore, under this scenario, the MPC controller is a linear MPC controller.

(ii) For all the other situations, even if setting $\theta = 0$ while $k \neq 0$, α will change between stages according to the Equation 12. The system is highly nonlinear. Therefore, for all the scenarios other the one illustrated in (i), our implemented MPC controller is a non-linear controller.

As illustrated before, we build our system dynamics based on the assumptions that the needle is stiff in axial direction and the it is also a rigid body model. Therefore, we choose to track only the position, orientation and velocity of the needle tip.

We consider a receding horizon MPC strategy with control horizon $N = 6$. The state constraints in our MPC controller can be divided into four parts. The first part is related to the system dynamics between two stages, which is illustrated in the Equation 16. The second part is about the limitations on the velocity of the needle tip, as we do not expect the needle to enter the body of patient too quickly, which we think is rather painful. Moreover, as shown in Equation 12, there is a trigonometric function appearing as the denominator in the equation between stages for α. So there is another limitation on the angle rotated for β is that the numerical value can not be equal to $(k+\frac{1}{2})\pi$, where $k \in \mathbb{N}$. Finally, the first stage for the needle model is set as the initial position to which all the other stages are compared. And also the initial speed v is set to 0 to simulate the situation that the needle tip just reach to the skin surface of the patient. Therefore, in general, the state constraints $x \in \mathbb{X}$ are:

$$\begin{aligned} x_{m+1} &= \mathbf{A}x + \mathbf{B}u \\ v &\leq v_{max} \\ \beta &\neq (k+\frac{1}{2})\pi \\ x_0 &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \end{aligned} \tag{21}$$

The control input constraints in our MPC controller is mostly related to the limitation of the power generated for the acceleration of the needle tip and the torque for the angle rotated around z-axis. Therefore, there are upper bounds and lower bounds for both the two control input parameters. In general, the control input constraints $u \in \mathbb{U}$ are:

$$\begin{aligned} a &\leq a_{max} \\ a &\geq a_{min} \\ \theta &\leq \theta_{max} \\ \theta &\geq \theta_{min} \end{aligned} \tag{22}$$

The cost function of this MPC model can be generally described as [7]:

$$\begin{aligned} J(x_0, u) &= \sum_{k=0}^{N-1} l(x(k), u(k)) + V_f(x(N)) \\ s.t. u &\in \mathbb{U}, x \in \mathbb{X} \end{aligned} \tag{23}$$

The stage cost $l(x,u)$ and the terminal cost $V_f(x)$ are defined as:

$$\begin{aligned} l(x,u) &= \frac{1}{2}(x^T\mathbf{Q}x + u^T\mathbf{R}u) \\ V_f(x) &= \frac{1}{2}x^T\mathbf{P}x \end{aligned} \tag{24}$$

where $\mathbf{Q} = diag(\begin{bmatrix} 10 & 10 & 10 & 0.1 & 0.1 & 0.1 & 10 \end{bmatrix})$, $\mathbf{R} = diag(\begin{bmatrix} 1 & 1 \end{bmatrix})$ and $\mathbf{P}$ is the solution of discrete algebric Riccati equation (DARE) used to solve the unconstrained infinite-horizon LQR problem for our linear MPC controller part when setting $\kappa = 0$ and $\theta = 0$. The cost for the α, β, γ are set to 0 because that the angle rotated in all three orientations for this scenario is 0 for all stages. While for our nonlinear MPC model in other scenarios, the terminal cost is selected to be $\mathbf{P} == diag(\begin{bmatrix} 100 & 100 & 100 & 0 & 0 & 0 & 100 \end{bmatrix})$ to avoid solving the DARE at each iteration. The selection of the cost matrix are obtained after tuning while inspecting the performance of our MPC controller.

Taking everything discussed before into account, the model predictive control problem is finally formalized as follows:

$$\begin{aligned} \min_{x,u} J(x_0, u) &= \sum_{k=0}^{N} \frac{1}{2}(x^T\mathbf{Q}x + u^T\mathbf{R}u + x^T\mathbf{P}x) \\ s.t. x_{m+1} &= \mathbf{A}x + \mathbf{B}u \\ v &\leq v_{max} \\ \beta &\neq (k+\frac{1}{2})\pi \\ x_0 &= \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \\ a &\leq a_{max} \\ a &\geq a_{min} \\ \theta &\leq \theta_{max} \\ \theta &\geq \theta_{min} \end{aligned} \tag{25}$$

III. Asymptotic stability

In this section, we show that the designed MPC asymptotically stabilized the closed-loop system.

A. Linearized system stability analysis

As discussed above, when setting the curvature κ, θ = 0, and the initial system states α,β, γ = 0, the nonlinear system can be largely simplified and the dynamics system can be linearized, the system dynamics tracks the trajectory in 1-D. Then the system can be formulated as:

$$x^+ = \mathbf{A}x + \mathbf{B}u \tag{26}$$

Where $\mathbf{A}$ is reduced to a 2-by2 matrix and $\mathbf{B}$ is also reduced to a 2-by-2 matrix. They represent the system dynamics in 1-D space. The linearized system can be verified as a controllable system. We calculate the controllable matrix $\mathbf{W_c} = [\mathbf{BAB}]$ and the matrix is full rank. As the Matrix $\mathbf{Q}$, $\mathbf{R}$ are positive definite and the terminal cost matrix $\mathbf{P}$ is the solution of Discrete Algebraic Riccati Equation(DARE), the costs of the problem is positive definite.

B. Proof of the system stability

As described in the **Section 2.6**[7], the **Theorem 2.41** proves the MPC stability without terminal constraints, which needs to satisfy **Assumption 2.2** and **Assumption 2.3** in [7].

(i) **Assumption 2.2** (Continuity of system and cost) It's obvious that the function f , the stage cost $l(x,u)$ and the terminal cost $V_f(X(N))$ is continuous. Since all the cost part has quadratic form and all of them are positive definite, also $f(0,0) = 0$, $\ell(0,0) = 0$, $V_f(0) = 0$. Therefore, the assumption is satisfied.

(ii) **Assumption 2.2** (Properties of constraint sets): According to the equation 22, all state constraints $x \in \mathbb{X}, u \in \mathbb{U}$ are closed and compact. The terminal region $\mathbb{X}_f \in \mathbb{X}$ is also compact. And $0 \in int(\mathbb{X}_f)$ is satisfied. Therefore this assumption is satisfied.

(iii) **Assumption 2.23** (Modified basic stability assumption): According to the **section 2.6** in [7], To approve the assumption, $\mathbb{X}_f(\cdot)$, $\ell(\cdot)$, $V_f(\cdot)$ have the following properties

- For all $x \in \mathbb{X}_f$, there exists a $u \in \mathbb{U}$ satisfying $V_f(f(x,u)) - V_f(x) \leq -\ell(x,u)$, $f(x,u) \in \mathbb{X}_f$
- For all $x \in \mathbb{X}$ and $u \in \mathbb{U}$ satisfying $\ell(x,u) \geq \alpha_1(|(x,u)|)$
- For all $x \in X_f$ satisfying $V_f(x) \leq \alpha_f(|x|)$, $x \in \mathbb{X}_f$

Constraints 22 are satisfied because the stage cost and terminal cost have the following quadratic forms:

$$\ell(x,u) = \frac{1}{2}(x^T Q x + u^T R u) \geq \frac{1}{2}(x^T Q x) \geq \frac{1}{2}\lambda(Q)|x|^2$$
$$V_f(x) = \frac{1}{2}x^T P x \leq \frac{1}{2}\lambda_{\max}(P)|x|^2 \tag{27}$$

Since the **Assumption 2.2**, **Assumption 2.3**, **Assumption 2.23** are all satisfied, the origin is exponentially stable for the closed loop system.

C. Estimating terminal region sets $\mathbb{X}_f$

Since we have proved the the stability for the linear system. Our goal is to find the invariant constraint admissble set $\mathbb{X}_f$ for the linearized closed loop system $x^+ = \mathbf{A}_K x$, where $\mathbf{A}_K = \mathbf{A} + \mathbf{BK}$ and $\mathbf{K}$ is the optimal control gain calculated by the unconstrained infinite-horizon Linear Quadratic Regulator problem. By applying the Algorithm 3.2 mentioned in [7], an invariant constraint admissble set $\mathbb{X}_f$ can be founded.

D. Estimating feasible set $\mathcal{X}_N$

The feasible set $\mathcal{X}_N$ can be deductive backward estimated with the estimation of terminal region sets $\mathbb{X}_f$. The basic idea is to check the feasibility of MPC problems with a set of initial states x_0 if the terminal stage $x(N)$ satisfy the terminal constraints $\mathbb{X}_f$. Once $x(N) \in \mathbf{X}_f$, then the x_0 should be included in the feasible set $\mathcal{X}_N$.

IV. Numerical Simulations

In this section, we run several numerical simulations in Python Axes3D environment, which is shown as Fig. 3. And in the following part, we are going to discuss and prove the effectiveness of our MPC model with various predetermined paths. Moreover, we would also like to choose a path as an example to discuss the different effects brought by changing various parameters of the model. Finally, we will compare the performance of the MPC model and the LQR model for the linear dynamic system when setting $\kappa = 0$ and $\theta = 0$.

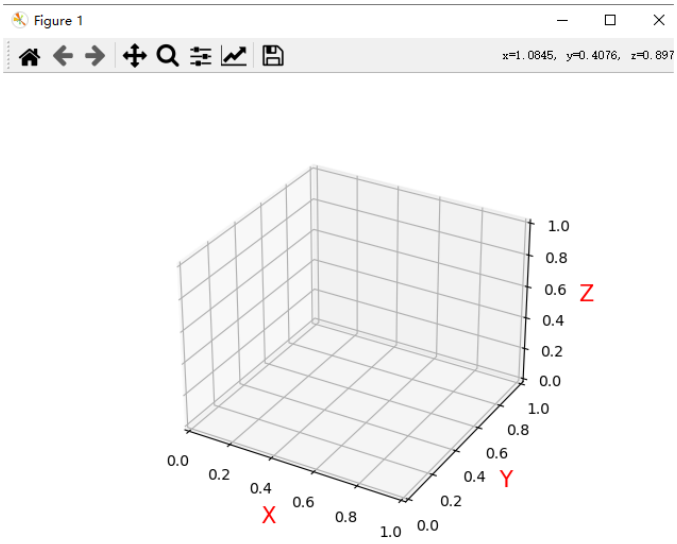

Fig. 3. Environments in Python used to run the simulations

A. Different paths to track

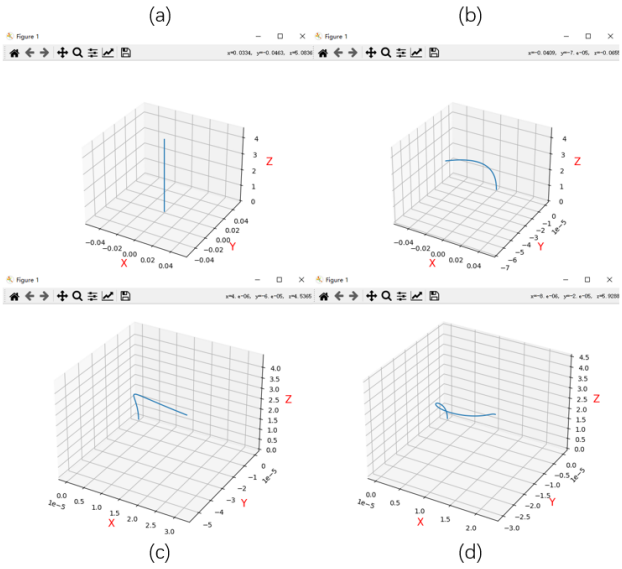

Fig. 4. The four paths with different shapes generated for the model to track, (a): the path when setting curvature $\kappa = 0$; (b): the path when setting curvature $\kappa = 1/150mm^{-1}$ and the rotational speed $\omega = 0rad/s$; (c): the path when setting curvature $\kappa = 1/150mm^{-1}$ and the rotational speed $\omega = 0.175rad/s$; (d): the path when setting curvature $\kappa = 1/150mm^{-1}$ and a varying rotational speed ω.

As illustrated previously in the report, our model is developed based on the application of a bevel-tip needle model.

There are generally several different paths of injection in actual clinical applications. And these paths can be generated from needle models with different curvatures κ [8]. Specifically, when $\kappa = 0$, this strongly nonlinear dynamic model can be simplified into a linear model, whose motion is only in one dimension. Moreover, the shape of paths also vary a lot even if the curvature κ is set to constant when the rotation angle ω changes. In this experiment, we create four paths of different shapes for our model to track, which is shown in Fig. 4.

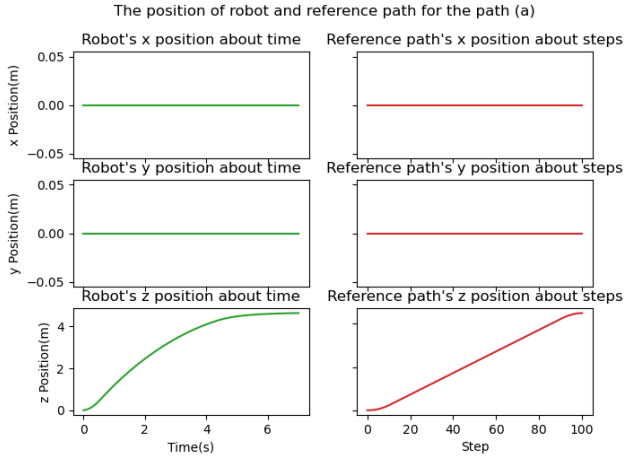

Fig. 5. The comparison between the robot's position and reference path (a)

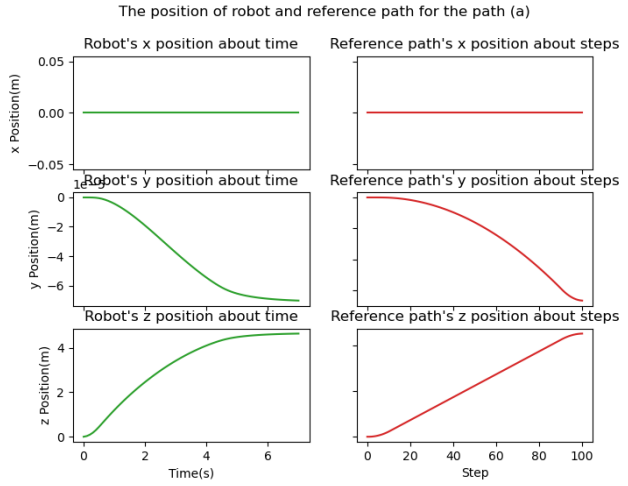

Fig. 6. The comparison between the robot's position and reference path (b)

According to Swaney, the patient will suffer form reduced tissue damage with a larger curvature κ [9]. And we believe that the model will be much more difficult to track for a higher κ, as the influence of rotating or stretching out to another orientation will be faster with a higher κ. Therefore, in our following experiment, we choose to set the curvature $\kappa = 1/150mm^{-1}$ for all the other paths except the one from going directly along the z-axis. The results of our experiment is separately shown in Fig. 5, Fig. 6, Fig. 7 and Fig. 8.

From the plots, we can see that our model successfully follow all the four predetermined paths within in a rather short

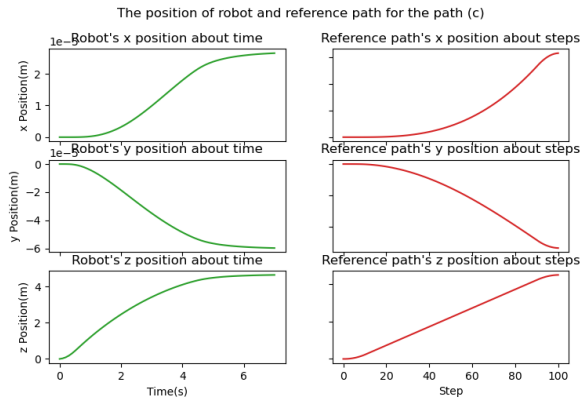

Fig. 7. The comparison between the robot's position and reference path (c)

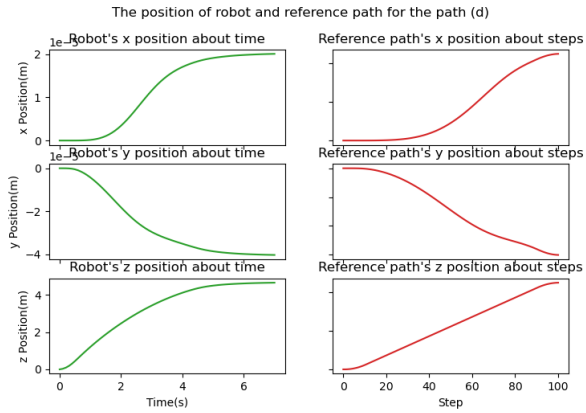

Fig. 8. The comparison between the robot's position and reference path (d)

time and it tends to stay stably near the end of the paths around 5 seconds after starting.

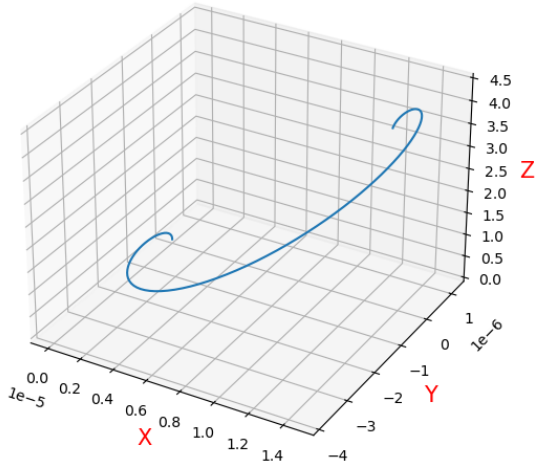

Fig. 9. The path failed to follow for our MPC model

However, these four paths are all generated with a low rotation speed, which is much more similar to the application in real world. However, when setting the rotation speed a little larger, which is only numerical applicable, for example, $\omega = 0.8rad/s$, our model fails to follow the path, which is

shown as Fig. 9. And the difference between reference path and the robot's motion is shown in Fig. 10. We think this is because that there is an error solved for the control input θ at the beginning of the experiment, which makes the model really difficult to make up this mistake. In the meantime, there are only small difference between other parameters in state, for example, position of x, velocity and angel rotated. Therefore, the MPC controller will give up making this mistake on y-axis.

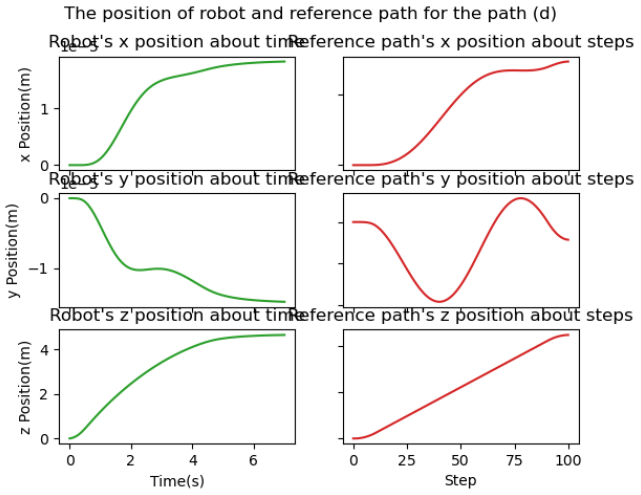

Fig. 10. The comparison between the robot's position and reference path for the path failed to follow for our MPC model

In conclusion, we generate five different paths with various shapes in this section. And for the following part in this report, we are going to take the path (c) in Fig. 4 as the example to analyze the difference brought by the varying parameters. This is because this path is the one with all the parameters not setting to zero and in the meantime, it is also the one path much more similar to the application in real life of a bevel-tip needle during surgery.

B. Different prediction horizon N

In this section, we perform experiments to see how the different prediction horizons (N) affect the result of our MPC model. As illustrated before, we will try to test the result with the path (c). And hereby we separately select the prediction horizon as 2, 4, 6 and 20.

The comparison result between different prediction horizon for the needle's tip position in the world coordinate system is shown in Fig. 11. The difference of the velocity of the needle's tip is shown in Fig. 12. It is obvious that there are great difference for the MPC controller between the prediction horizons. We can tell that when setting N to be larger, the model will react to follow the path more rapidly. It is much more obvious in Fig. 12 that the model will have a larger velocity for $N = 6$ and quickly decelerate to $v = 0$ before reaching to the end of the path. However, when setting $N = 20$ for our model, it can be seen that the maximum of velocity of our model is even slower than that of the $N = 4$, which further proves that there is a limitation on the selection of prediction horizon. We also found that there is only a slight improvement

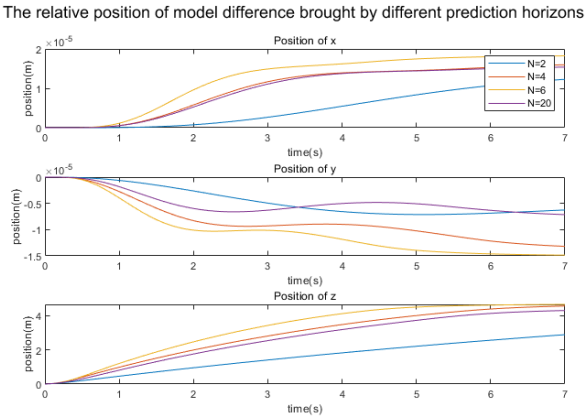

Fig. 11. The performance of position with different horizons

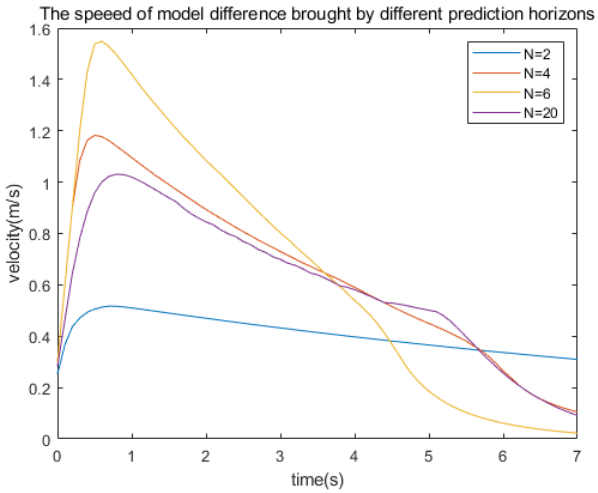

Fig. 12. The performance of velocity with different horizons

on the reaction time when setting $N = 8$, but result in a lower running time of the code. Therefore, we finally choose $N = 6$ as our parameter for prediction horizon.

C. Different stage cost $\mathbf{Q}$

In this part, we compare the result of MPC performance brought by different choices of cost matrix $\mathbf{Q}$ for stages. In our model, we select the cost matrix as $\mathbf{Q} = diag(\begin{bmatrix}10 & 10 & 10 & 0.1 & 0.1 & 0.1 & 10\end{bmatrix})$, based on our trials and understanding of the model. In this part, we tried few more stage cost matrix $\mathbf{Q}$, which is shown in Fig .13 and Fig. 14.

In the figures, we can see that the main difference of varying stage cost matrix $\mathbf{Q}$ is the maximum velocity of the model. When the number is smaller, the maximum velocity of our model is larger. Actually, the blue line and the purple line overlap with each other in the figure of z position and velocity. However, by comparing the final stable position of the model for x and y position, it is obvious that our Q outperforms as it is the one most similar to that of the reference path. This is why we select this matrix as our stage cost. And we can also tell that we can not set the number to be too large as the green line performs really bad on no matter the velocity or the

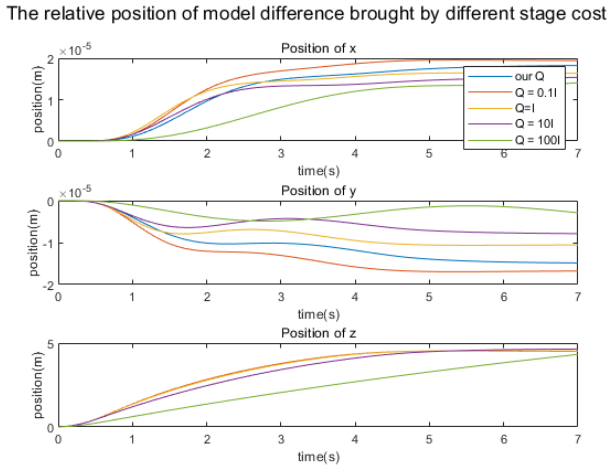

Fig. 13. The performance of position with different $\mathbf{Q}$ matrix

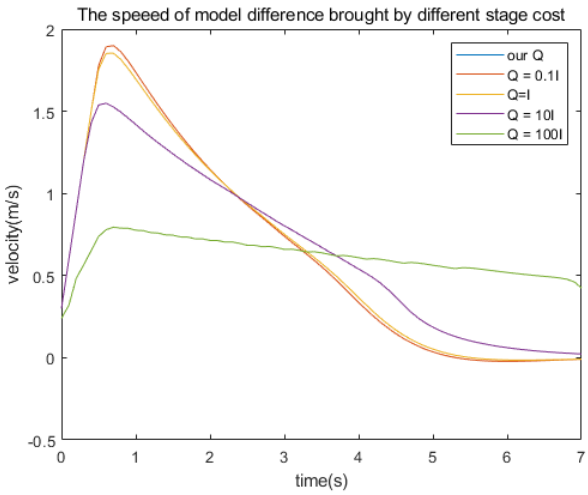

Fig. 14. The performance of velocity with different $\mathbf{Q}$ matrix

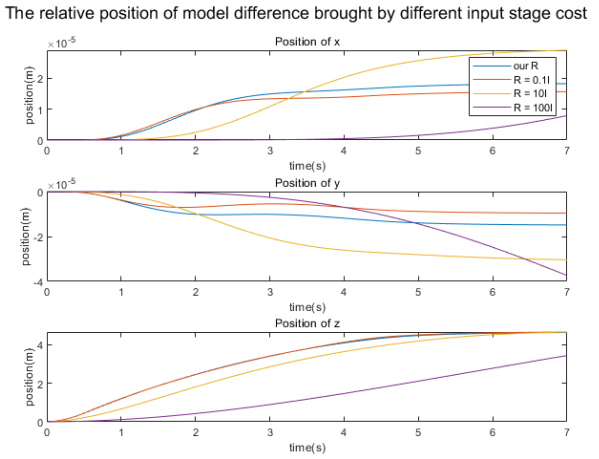

Fig. 15. The performance of position with different $\mathbf{R}$ matrix

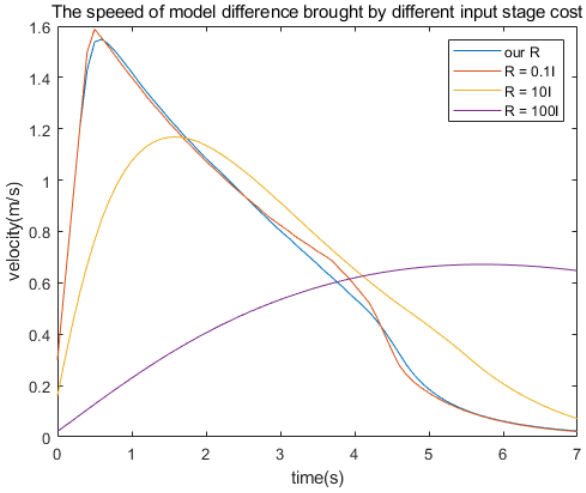

Fig. 16. The performance of velocity with different $\mathbf{R}$ matrix

position. Moreover, we select this matrix because we actually only need our needle tip to reach to a certain position while care little about the angle rotated with respect to the initial position. Thus, we only give a small number for the costs of the three angles rotated.

D. Different input stage cost $\mathbf{R}$

In this part, we perform the similar experiment as we do in the previous section, but this time the object is the input stage cost $\mathbf{R}$. In our model, we finally select $\mathbf{R} = diag(\begin{bmatrix}1 & 1\end{bmatrix})$ as our input stage cost matrix. The result between different cost matrix is shown in Fig. 15 nad Fig. 16.

From the figures, we can tell that when giving a larger cost matrix to the input stage, the velocity will be greatly punished as the optimization process will give a smaller acceleration a as the result. And there is only a really small difference between setting $\mathbf{R} = diag(\begin{bmatrix}1 & 1\end{bmatrix})$ and $\mathbf{R} = diag(\begin{bmatrix}0.1 & 0.1\end{bmatrix})$, the blue line is much more smoother than that of the red line. Moreover, it is obvious again that when setting the $\mathbf{R}$ to be the value used in our model, there is only slight difference for the position of x and y between the robot's model and reference path.

E. Different terminal stage cost $\mathbf{P}$

Again, the same experiment is performed for the terminal stage cost $\mathbf{P}$ in our MPC model. We initially set $\mathbf{P} = diag(\begin{bmatrix}100 & 100 & 100 & 0 & 0 & 0 & 100\end{bmatrix})$ in the model we used to avoid solving the DARE at each iteration for our non-linear MPC model. We think that the terminal stage cost need to be set larger than that of the stage stage cost $\mathbf{Q}$ to make it prone to stop at the exact position of end of the reference path, and again we think the relative position of the needle tip and the final velocity $v = 0$ is what we are really concerning about, we could just neglect the difference of the angle rotated. The result plot of performance are shown in Fig. 17 and Fig. 18.

As shown in figures, a smaller terminal stage cost results in a longer reacting time to reach to the final destination. Besides, since the value of position x and y is much smaller comparing to the other values, a smaller terminal stage costs will also

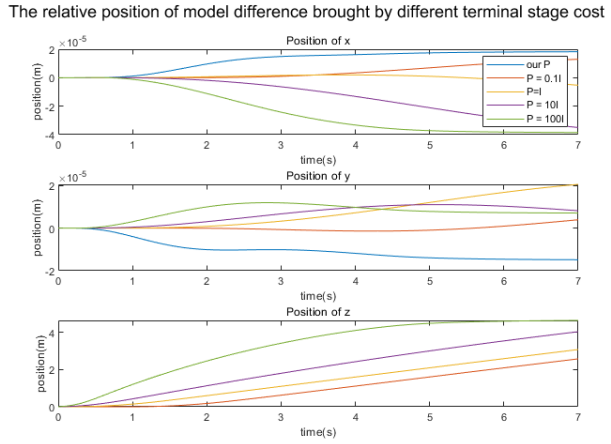

Fig. 17. The performance of position with different $\mathbf{P}$ matrix

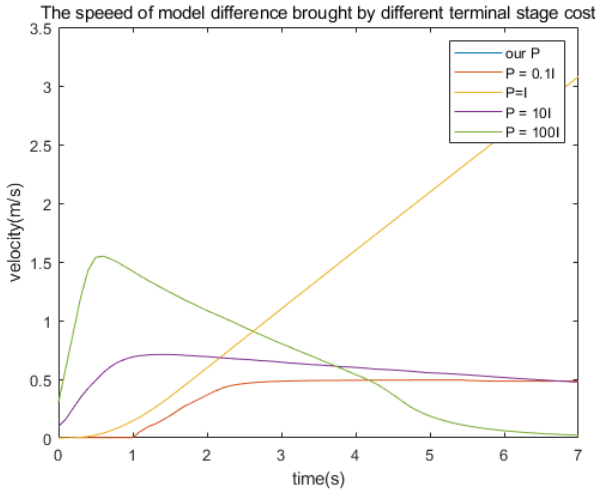

Fig. 18. The performance of velocity with different $\mathbf{P}$ matrix

result in a larger deviation in these two dimensions. And it is also really weird for selecting $\mathbf{P} = I$ that the velocity seems to go to infinity. And again, we can see that there is only slight changes between $\mathbf{P} = I$ and our value for $\mathbf{P}$, as the two matrix is closer to each other. By comparing the final position, our value for $\mathbf{P}$ outperforms than all the other parameters.

V. Discussion

While implementing this project, we actually do meet with few problems would like to share. The first problem is how to linearize a model with such high non-linearity. As shown in Equation 16, the kinematic model we are using is of rather high non-linearity and there are lots of parameters in state of the robot used to determine another single state. Moreover, it would really deviate a lot from the original path only if we delete one of the trigonometric functions shown in the equation. Moreover, the article we cited itself also preserve its nonlinearity and take the whole model into calculation when they perform their experiment. Therefore, we are really curious about if there is a specific method to linearize such a complicated dynamics model in 3 dimensions.

The other question remains unsolved is that we are not sure about how to solve the reference state for our robot if there is only reference position given. Since the orientation of the robot on a curve may be totally different from that of on a straight line especially when the velocity is high and the time between time step is long. Therefore, it is really hard to calculate the angle rotated and the reference speed out using the same way for a bicycle model if we are only given the two coordinates.

Moreover, just as illustrated in Section 4.1 of the report, when a larger rotation speed is given, it is really hard for our MPC model to follow the path even if we tried a lot with various parameters sets. And we think it is because that the optimization procedure could not successfully solve the θ at the very beginning and just give up making up the errors in the later process.

In the future, we are going to explore the linearization of the model and also try to read more papers to find a way to calculate the reference state between two given positions. We think the application of model predictive control in the bevel-tip needle can definitely be well developed to help patients relieve pain and reduce the workload of medical staff in the future.

VI. Conclusion

In this paper, we successfully build a model predictive control model for the bevel-tip needle model to track the given path. By illustrating the dynamic model and MPC design, we present our procedure to get to our final code. With the stability proof, we manage to prove the theory background for our MPC model. Finally, we use different reference path to compare the performance of our MPC model with different parameter sets to prove our solution is optimal and effective to solve the problem.

References

[1] H. Holm, J. K. Kristensen, S. N. Rasmussen, A. Northeved, and H. Barlebo, "Ultrasound as a guide in percutaneous puncture technique," *Ultrasonics*, vol. 10, no. 2, pp. 83–86, 1972.

[2] D. J. Kopacz and H. W. Allen, "Comparison of needle deviation during regional anesthetic techniques in a laboratory model," *Anesthesia & Analgesia*, vol. 81, no. 3, pp. 630–633, 1995.

[3] R. J. Webster III, J. S. Kim, N. J. Cowan, G. S. Chirikjian, and A. M. Okamura, "Nonholonomic modeling of needle steering," *The International Journal of Robotics Research*, vol. 25, no. 5-6, pp. 509–525, 2006.

[4] R. Secoli, F. Rodriguez *et al.*, "Experimental validation of curvature tracking with a programmable bevel-tip steerable needle," in *2018 International Symposium on Medical Robotics (ISMR)*. IEEE, 2018, pp. 1–6.

[5] B. Huo, X. Zhao, J. Han, and W. Xu, "Path-tracking control of bevel-tip needles using model predictive control," in *2016 IEEE 14th International Workshop on Advanced Motion Control (AMC)*. IEEE, 2016, pp. 197–202.

[6] J. C. Chou and M. Kamel, "Finding the position and orientation of a sensor on a robot manipulator using quaternions," *The international journal of robotics research*, vol. 10, no. 3, pp. 240–254, 1991.

[7] J. Rawlings and D. Mayne, *Model Predictive Control: Theory and Design*. Nob Hill Publishing, 2008.

[8] R. Alterovitz, K. Goldberg, and A. Okamura, "Planning for steerable bevel-tip needle insertion through 2d soft tissue with obstacles," in *Proceedings of the 2005 IEEE international conference on robotics and automation*. IEEE, 2005, pp. 1640–1645.

[9] P. J. Swaney, J. Burgner, H. B. Gilbert, and R. J. Webster, "A flexure-based steerable needle: high curvature with reduced tissue damage," *IEEE Transactions on Biomedical Engineering*, vol. 60, no. 4, pp. 906–909, 2012.