dVRK Kinematics Modeling Using Product of Exponentials

Zachary Serocki, Evan Carmody, Wyatt Harris, Natanel Pinkhasov

Robotics Engineering
Worcester Polytechnic Institute (WPI), Worcester, Massachusetts, USA
{zdserocki, epcarmody, wwharris, npinkhasov}@wpi.edu

***Abstract*—This paper presents the implementation of Kinematic Modeling with the Product of Exponentials (PoE) for the da Vinci Research Kit's (dVRK) Patient-Side Manipulator (PSM). Prior work modeled the arms using Denavit-Hartenberg (DH) parameters. In this project, we developed forward (FK) and inverse kinematics (IK) algorithms for the PSM robot component in MATLAB, making use of the Product of Exponentials and the assignment of screw axes. We implemented the analytical Jacobian and Gradient Descent models for IK and validated these results against the existing DH results presented in the reference paper [1].**

I. Paper Review and Analysis

The Convex Optimization-Base Dynamic Model Identification Package for the da Vinci Research Kit (dVRK) provides a open source framework for dynamic modeling and identification and includes resources for modeling the specific dVRK components, such as parallelograms, springs, and counterweights. The researchers present a convex optimization strategy for optimizing the accuracy of the dynamics for each component of the dVRK.

After incorporating link inertias, springs, joint frictions, cable force, tendon couplings, and closed-chains into their open-source package, the authors claim that the modeling performed is effective and is versatile enough to be applied to other robots outside of the dVRK package. They claim to implement a convex optimization-based method for obtaining the dynamic parameters and that the experimental results have improved modeling and package robustness. Post-implementation, while identification performance is improved relative to another paper, deviations between the measured and predicted torques are still prevalent. They also conclude that non-linear friction models that consider pre-sliding hysteresis are not compatible with the package they developed, which poses an obstacle for more accurate modeling.

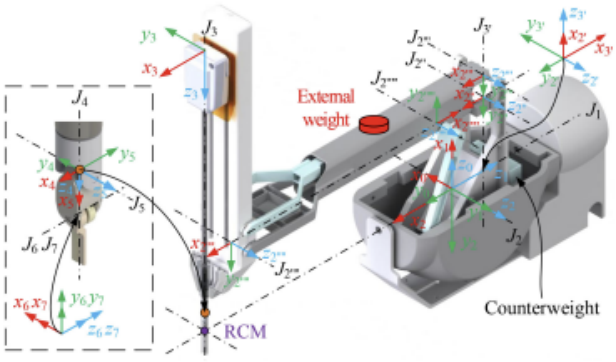
II. Materials and Methods

A. Overview

The researchers used a variety of hardware and software to accomplish and evaluate performance. The robots used were from the da Vinci Research Kit–a kit provided to universities without the software to model the kinematics or the dynamics. From this kit, the team focused on two components, the Master Tool Manipulator (MTM) and Patient Side Manipulator (PSM) models.

B. Kinematics Modeling

To define the kinematics for both the MTM and PSM models, the authors expressed the several joint coordinate types, namely q^d (the joint coordinates in the dVRK-ROS package) and q (the joint coordinates used in this paper's kinematic modeling), where q contains q^a and q^b, the additional joint coordinates and basis joint coordinates, respectively. q^c is also defined to represent the complete joint coordinates. The authors also note that the dimensions are taken from the dVRK user guide and otherwise manually measured. For the MTM, there are left and right models that can be similarly modeled. The MTM kinematics are described by the q^b. The authors were able to model the first five of seven joints of the PSM the same way as the dVRK-ROS package, while the last two joints were modeled as q^{d_6}. Note that all frame definitions for both models are based on Denavit-Hartenberg (DH) conventions.



(a) Frame definition of the PSM using modified DH convention.

Fig. 1. Assigned DH Frames of the PSM [1].

C. Dynamics Modeling

To create dynamic parameters for each link k, the researchers calculate the inertia tensor, L_k. Using this inertia tensor as well as the mass of the link, m_k, they construct a subset of linkage base parameters for joint k called $\delta_{Lk} \in \mathbb{R}^{10}$.

TABLE II
MODELING DESCRIPTION OF THE PSM

i	$a(i)$	a_{i-1}	α_{i-1}	d_i	θ_i	$\boldsymbol{\delta}_{Li}$	I_{mi}	$\boldsymbol{F}_i$	K_{si}
1	0	0	$\frac{\pi}{2}$	0	$q_1 + \frac{\pi}{2}$	✓	×	✓	×
2	1	0	$\frac{\pi}{2}$	0	$q_2 \frac{\pi}{2}$	✓	×	✓	×
$2'$	2	l_{2L3}	0	0	$\frac{\pi}{2}$	×	×	×	×
$2''$	$2'$	l_{2H1}	0	0	$\frac{\pi}{2} q_2$	✓	×	×	×
$2'''$	$2'$	l_{c1}	0	0	$\frac{\pi}{2} q_2$	✓	×	×	×
$2''''$	$2''$	l_{2L2}	0	0	q_2	✓	×	×	×
$2'''''$	$2''$	l_{2L1}	0	0	$q_2 + \pi$	✓	×	×	×
3	$2''''$	l_3	$\frac{\pi}{2}$	$q_3 + l_{c2}$	0	✓	×	✓	×
$3'$	2	l_{2L3}	$\frac{\pi}{2}$	q_3	0	✓	×	×	×
4	3	0	0	l_{tool}	q_4	×	✓	✓	✓
5	4	0	$\frac{\pi}{2}$	0	$q_5 + \frac{\pi}{2}$	×	✓	✓	×
6	5	l_{p2y}	$\frac{\pi}{2}$	0	$q_6 + \frac{\pi}{2}$	×	×	✓	×
7	5	l_{p2y}	$\frac{\pi}{2}$	0	$q_7 + \frac{\pi}{2}$	×	×	✓	×
M_6	-	0	0	0	q_6^m	×	✓	✓	×
M_7	-	0	0	0	q_7^m	×	✓	✓	×
F_{67}	-	0	0	0	$q_6 \quad q_7$	×	×	✓	×

Note: Links 1 to 7 correspond to the links sescribed in Fig. 4 a. M_6 and M_7 correspond to the modeling of motors 6 and 7, respectively. F_{67} corresponds to the modeling of the relative motion between links 6 and 7. The dimensions are shown in Fig. 4 a. $l_{c1} = l_{2H1} + l_{2H2}, l_{c2} = -l_{RCC} + l_{2H1}$.

Fig. 2. Defined DH Table of the PSM [1].

Additionally, they construct a subset of base parameters from the properties of the joint, including friction, inertia, and spring stiffness, which is labeled $\delta_{Ak} \in \mathbb{R}^5$. The total base parameter variable δ combines all the base parameters for each joint into one large vector.

To formulate an inverse dynamic model for the dVRK, the authors use the Euler- Lagrangian equation. They ensure that they properly identify inertia, friction, and spring stiffness in their calculations. This results in the development of a function $\boldsymbol{\tau}^m = \boldsymbol{H}(\boldsymbol{q}^m, \dot{\boldsymbol{q}}^m, \ddot{\boldsymbol{q}}^m)\boldsymbol{\delta}$. To calculate the dynamics of the MTM, additional terms are calculated and added to counteract the torques of an electrical wire and a spring that counteracts gravitational force. For the PSM calculations, the inertia of the gripper and wrist are ignored because they are insignificant compared to that of the modeled motors. There are many joints where friction can be ignored as well. They also account for a torsional spring that applies torque resetting joint 4 back to it's home position.

D. Trajectory Optimization and Data Collection

To collect data that can be used to find the base parameters, the authors of [**?**] developed a trajectory optimized for the task. They generate trajectories based on the Fourier series given by

$$q_k^m(t) = q_{ok}^m + \sum_{l=1}^{n_{\mathrm{H}}} \frac{a_{lk}}{\omega_f l} \sin(\omega_f l t) - \frac{b_{lk}}{\omega_f l} \cos(\omega_f l t) \qquad (1)$$

In addition, hard bounds are placed on the trajectory to prevent potential problems. Joint position limits are enforced, as well as a joint speed limit and confinement of the end effector to the workspace. They then selected frequency parameters to produce trajectories within the given bounds for both the MTM and PSM, and data from the real dVRK as it attempted to follow the trajectory. They recorded information on the joint positions and velocities as well as motor torques at 200 hz using a low-pass filter to remove noise.

E. Parameter Identification

To identify parameters, the researchers used their function $\boldsymbol{\tau}^m = \boldsymbol{H}(\boldsymbol{q}^m, \dot{\boldsymbol{q}}^m, \ddot{\boldsymbol{q}}^m)\boldsymbol{\delta}$, where δ is the set of base parameters that is being identified. Applying this formula at each timestep, we find the regression matrix

$$\boldsymbol{W}_b = \begin{bmatrix} \boldsymbol{H}_b(\boldsymbol{q}_1^m, \dot{\boldsymbol{q}}_1^m, \ddot{\boldsymbol{q}}_1^m) \\ \boldsymbol{H}_b(\boldsymbol{q}_2^m, \dot{\boldsymbol{q}}_2^m, \ddot{\boldsymbol{q}}_2^m) \\ \vdots \\ \boldsymbol{H}_b(\boldsymbol{q}_S^m, \dot{\boldsymbol{q}}_S^m, \ddot{\boldsymbol{q}}_S^m) \end{bmatrix} \qquad (2)$$

Multiplying this matrix W_b by δ finds ω, where

$$\boldsymbol{\omega} = \begin{bmatrix} \boldsymbol{\tau}_1^m \\ \boldsymbol{\tau}_2^m \\ \vdots \\ \boldsymbol{\tau}_S^m \end{bmatrix} \qquad (3)$$

Finally, they solve for the base parameters using convex optimization, as well as ordinary and weighted least squares. Weights for each joint error were calculated using the formula $w_i = \frac{1}{\max\{\tau_{mi}\} - \min\{\tau_{mi}\}}$. Where i is the joint number, and the max and min function analyze all recorded time steps.

F. Validation

To assess the validity of their predicted base parameters $\hat{\delta}$, torque predictions were generated using the identified parameters on recorded test data that was set aside and not used for training. The relative root mean squared error was found for different trajectories and individual motors. They then repeated the entire data collection and parameter identification process on the PSM with a 200 g weight attached to link 2" to ensure that the new weight would be recognized.

III. REPLICATION

We initially implemented the forward kinematics of the PSM by replicating the DH table used by Wang et al. in MATLAB. After replicating the results using the provided parameters, a set of functions was developed using the Product of Exponentials (PoE) and was implemented in hopes of obtaining similar forward kinematics to those produced using the Denavit-Hartenberg (DH) conventions [1].

The challenging aspect is that the Patient Side Manipulator (PSM) has a different form of joint than what was studied in class. With joint two being a linkage with an additional rotation at the end means that it can't be generalized by pure rotation or linear displacement. Our implementation uses planar joints to model the kinematic constraints of the linkage [5]. This form of joints can be generalized to any form of linkage and the column of which can be thought of stepping

through a process of rotation then linear movements through the rotated frame. The screw axis of the robot follow the form:

$$\mathcal{S} = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & -0.1445 & 0.0401 \\ 1 & 0 & 0 & 0 & 0.1445 & 0.4759 \\ 0 & 0 & 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0.4358 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0.5589 & -0.4358 \\ 0 & 1 & 0 & 0.5598 & 0 & 0 \\ 0 & 0 & -1 & 0.4358 & -0.0009 & 0 \end{bmatrix}^T \tag{4}$$

Where the first column is the screw axis of the first joint, columns three, four and five are for the second joint and the rest of the columns correspond with the rest of the joints. The home configuration matrix stores the offset that previously lied in the intermediate transform in the Denavit-Hartenberg method and thus it takes the form

$$M = \begin{bmatrix} 0 & 1 & 0 & -0.0009 \\ 1 & 0 & 0 & -0.4358 \\ 0 & 0 & -1 & -0.5598 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

A. Forward Kinematics

In the forward kinematics functions, the program is able to run the standard forward kinetics algorithm $f(q) = (\prod_{i=1}^{7} e^{[\mathcal{S}_)]q_i})M$ with the displacement of joint two being applied to each column of the planar movement. This calculates the position and orientation of the end effector while minimizing the number of computations.

Algorithm 1: Forward Kinematics

Require: q,S,M
$q^* \leftarrow \begin{pmatrix} q_1 & q_2 & q_2 & q_2 & q_3 & q_4 & q_5 & q_6 & q_7 \end{pmatrix}$
$T \leftarrow I_{4x4}$
for $i = 1\ n$ **do**
$T \leftarrow T \times exp([\mathcal{S}_i]q_i^*)$
end
$T \leftarrow T \times M$

B. Differential Kinematics

The differential kinematics with a planar joint requires a similar expression of joint variables to match the displacements with the corresponding screw axis. The differential kinetics poses a unique issue in that Jacobian that is computed is a 6x10 matrix since it is computed with respect to q^* to convert the Jacobian to be with respect to q the corresponding columns need to be added together to, this is adding up the contribution to the velocity from the joint. Let

$$\dot{q}^* = \begin{bmatrix} \vdots \\ \dot{q}_i \\ \dot{q}_i \\ \dot{q}_i \\ \vdots \end{bmatrix}, \quad J^* = \begin{bmatrix} \cdots & J_i & J_{i+1} & J_{i+2} & \cdots \end{bmatrix}$$

where the planner joint is located at joint i. Then the velocity is given by

$$V = J^* q^*$$

$$V = \begin{bmatrix} \cdots + J_i\dot{q}_i + J_{i+1}\dot{q}_i + J_{i+2}\dot{q}_i + \cdots \end{bmatrix}$$

$$V = \begin{bmatrix} \cdots & J_i + J_{i+1} + J_{i+2} & \cdots \end{bmatrix} \begin{pmatrix} \vdots \\ \dot{q}_i \\ \vdots \end{pmatrix}$$

The algorithm to perform this looks like

Algorithm 2: Differential Kinematics

Require: q,S
$q^* \leftarrow \begin{pmatrix} q_1 & q_2 & q_2 & q_2 & q_3 & q_4 & q_5 & q_6 & q_7 \end{pmatrix}$
$T \leftarrow I_{4x4}$
for $i = 1\ n$ **do**
$J_i \leftarrow Ad_T(S_i)\ T \leftarrow T \times exp([\mathcal{S}_i]q_i^*)$
end
$J \leftarrow \begin{bmatrix} J_1 & J_2 + J_3 + J_4 & J_5 & J_6 & J_7 & J_8 & J_9 & J_{10} \end{bmatrix}$

C. Inverse Kinematics

The original paper did not implement inverse kinematics as a part of their library, but we were interested in implementing full kinematic control of the robot. We decided to implement the Gradient Descent method for inverse kinematics, using our previously written PoE forward kinematics. Our Ikine function takes a 3x1 target pose x in the space frame, a 6x7 space frame screw axes matrix S, a 4x4 homogeneous transformation matrix M, and a 7x1 vector of initial guess joint displacements q0. It then outputs the 7x1 vector of joint displacements needed for the desired position. The function employed the use of the jacoba function produced for other assignments in this course. Collectively this approach proved to be effective and worked with the PSM manipulator seamlessly. To test the validity, a file called testIkine was developed that ensured our Ikine function could find valid joint configurations for Space Frame coordinates randomly generated within the workspace of the robot. In order to ensure coordinates were within the workspace of the PSM, we generated positions based on joint variables created between the joint limits described by Fontanelli et al. [7].

D. Simulation

To determine whether the formulation of the Product of Exponentials yields the same results as the DH method outlined in the paper, simulations were conducted in which the same joint angles were passed into the functions of both methods. The outputs from both methods were considered data that were then compared. We subsequently implemented our PoE approach in MATLAB and ran the DH approach using the Python code provided by the researchers. To gather results for the DH robot joint angles were fed in a similar form where there were repeats of joint two and there was a 0 injected into the sequence to account for the static transform. To find the

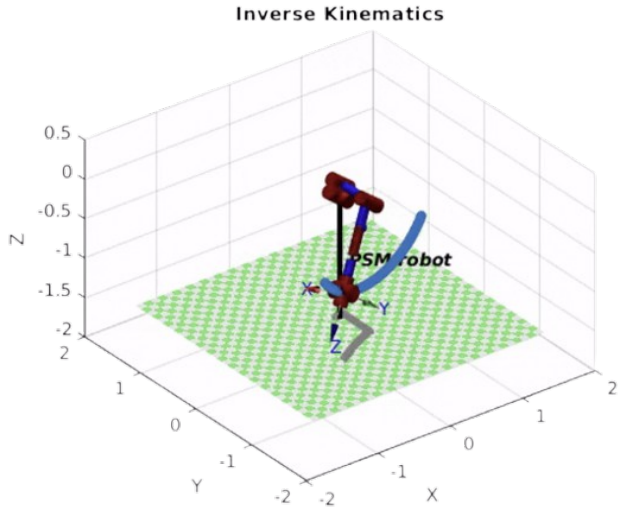


Fig. 3. MATLAB Simulation of Inverse Kinematics for specified path

Jacobian from the convoluted joint input, it was required that it be built column-by-column, setting each joint to one and constructing the Jacobian as such.

IV. Results

Tests were conducted by running the DH forward kinematics and the screw axis formulation on a variety of inputs to validate our methods. The inverse kinematics was used to calculate the joint displacements required to follow the path of an arc. Comparing the output product of the exposure formulation with the DH method yields the same results in a multitude of tests, allowing us to be certain that the formulation is effective at estimating and determining the mapping from the joint space to the task space. The inverse kinematics was able to converge rapidly and worked no differently than any other formulation; this provides further validation that the algorithms work as intended. Furthermore, testing was conducted on various forms of the Jacobian, ensuring that the algorithm holds true for a variety of processes that can be preformed to a standard Product of Exponentials formulation such as determining the body frame Jacobian and the Cartesian velocity analytic Jacobian. The use of black box testing, verifying the output of the function for a given input, allows us to be certain that the algorithms developed can be applicable for a variety of systems dependent on them and a variety of manipulators.

V. Discussion

Overall, our team successfully validated a kinematic model using the Product of Exponentials (PoE) for the da Vinci Research Kit's Patient Side Manipulator device (dVRK PSM). Through replication of the results outlined in "Convex Optimization-Base Dynamic Model Identification Package for the da Vinci Research Kit", this project has demonstrated that it is possible to implement PoE-based methods rather than relying on Denavit-Hartenberg (DH) conventions to solve for the kinematic solutions of a dVRK PSM. We learned that it is achievable to model the same framework defined by the research team from Wang et al. [1] in MATLAB using the Peter Corke Robotics Toolkit. Additionally, in this process, we learned how to implement a planar joint using this MATLAB framework to more accurately model the PSM linkage. Furthermore, our approach is generalizable to linkages that are not just the PSM and due to its simplicity the functionality can be abstracted, allowing it to be scalable to multiple joints. Thus, this method could be extended to other general manipulators.

References

[1] Y. Wang, R. Gondokaryono, A. Munawar, and G. S. Fischer, "A convex optimization-based dynamic model identification package for the da Vinci Research Kit," *IEEE Robotics and Automation Letters*, vol. 4, pp. 3657–3664, Oct. 2019.

[2] Intuitive Surgical Inc., "da Vinci Research Kit (dVRK)," Intuitive Surgical, Sep. 21, 2023. [Online]. Available: https://research.intusurg.com/index.php/Main_Page

[3] "Patient side manipulators — dVRK devel," Read the Docs, 2025. [Online]. Available: https://dvrk.readthedocs.io/devel/pages/kit/from_ISI/Classic/PSM.html

[4] G. Chrysilla, N. Eusman, A. Deguet, and P. Kazanzides, "A compliance model to improve the accuracy of the da Vinci Research Kit (dVRK)," *Acta Polytechnica Hungarica*, vol. 16, no. 8, 2019. [Online]. Available: https://acta.uni-obuda.hu/Chrysilla_Eusman_Deguet_Kazanzides_95.pdf

[5] R. Featherstone, "Rigid Body Dynamics Algorithms," SpringerLink, 2020, doi: https://doi.org/10.1007-978-1-4899-7560-7

[6] "WPI-AIM: Dynamic model identification of the DVRK," GitHub, 2025. https://github.com/WPI-AIM/dvrk_dynamics_identification

[7] G. Fontanelli, F. Ficuciello, L. Villani, and B. Siciliano, "Modelling and identification of the da Vinci Research Kit robotic arms," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, Sep. 2017, pp. 1464–1469. doi: https://doi.org/10.1109/IROS.2017.820594810.1109/IROS.2017.8205948.

Github: https://github.com/Zac-BB/RBE501dVRKModeling/releases/tag/final