Technion – Israel Institute of Technology

Faculty of Mechanical Engineering

HW3

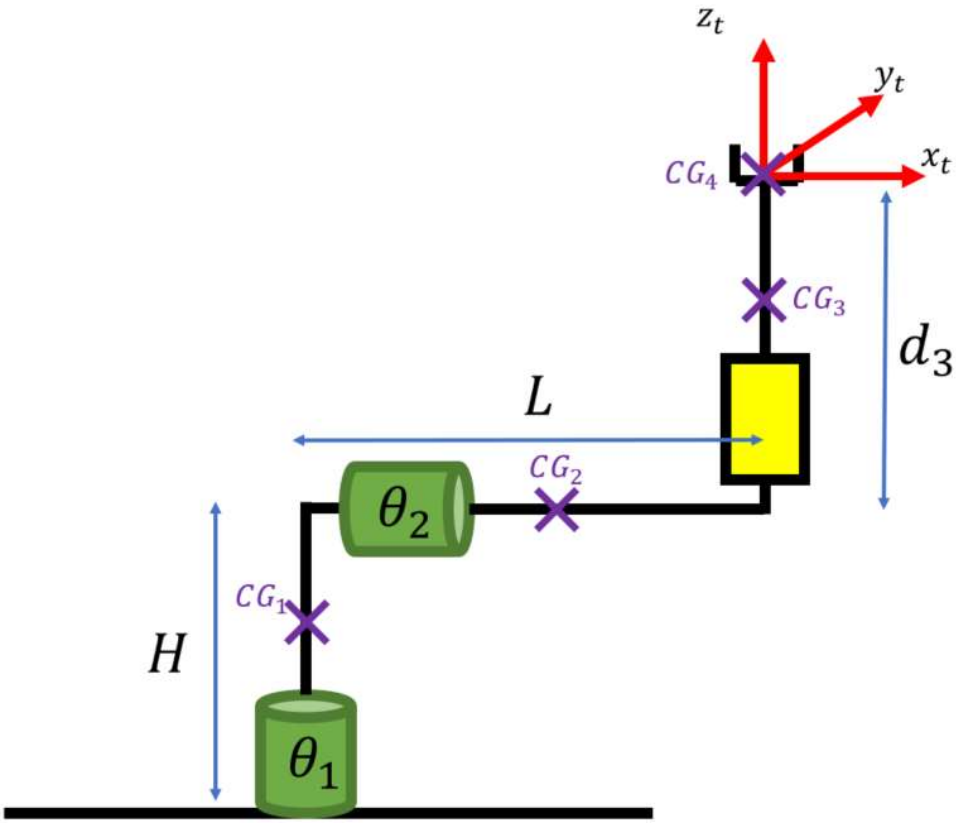
Kinematics, Dynamics, and Control of Robots

036026

Alon Spinner	305184335	alonspinner@gmail.com
Shahar Tsadok	203783519	Shahar507@gmail.com

January 7, 2021

1 Finding the Dynamic Equation of Motion



1.1 Computing Link's Jacobians in link's coordinate system

$$J_{1,t} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad J_{2,t} = \begin{bmatrix} 0 & 0 & 0 \\ \frac{Lc_2}{2} & 0 & 0 \\ -\frac{Ls_2}{2} & 0 & 0 \\ 0 & 1 & 0 \\ s_2 & 0 & 0 \\ c_2 & 0 & 0 \end{bmatrix}$$

$$J_{3,t} = \begin{bmatrix} (d_3 - \frac{L_1}{2})s_2 & 0 & 0 \\ Lc_2 & \frac{L_1}{2} - d_3 & 0 \\ -Ls_2 & 0 & 1 \\ 0 & 1 & 0 \\ s_2 & 0 & 0 \\ c_2 & 0 & 0 \end{bmatrix} \quad J_{4,t} = \begin{bmatrix} d_3 s_2 & 0 & 0 \\ Lc_2 & -d_3 & 0 \\ -Ls_2 & 0 & 1 \\ 0 & 1 & 0 \\ s_2 & 0 & 0 \\ c_2 & 0 & 0 \end{bmatrix}$$

1.2 Computing System Inertias

$$\begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{bmatrix} = \begin{bmatrix} \frac{\pi D^2}{4}\rho H \\ \frac{\pi D^2}{4}\rho L \\ \frac{\pi D^2}{4}\rho L_1 \\ M \end{bmatrix} = \begin{bmatrix} 0.2757 \\ 0.1378 \\ 0.1378 \\ 0.5 \end{bmatrix} [kg]$$

The Inertial matrices are written in the link's coordinate system

$$I_1 = \frac{m_1 H^2}{12}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} = 0.9189 \cdot 10^{-3}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} [kg \cdot m^2]$$

$$I_2 = \frac{m_2 L^2}{12}\begin{bmatrix} 0 & & \\ & 1 & \\ & & 1 \end{bmatrix} = 0.1149 \cdot 10^{-3}\begin{bmatrix} 0 & & \\ & 1 & \\ & & 1 \end{bmatrix} [kg \cdot m^2]$$

$$I_3 = \frac{m_3 L_1^2}{12}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} = 0.1149 \cdot 10^{-3}\begin{bmatrix} 1 & & \\ & 1 & \\ & & 0 \end{bmatrix} [kg \cdot m^2]$$

$$I_4 = \begin{bmatrix} 0 & & \\ & 0 & \\ & & 0 \end{bmatrix} [kg \cdot m^2]$$

1.3 Computing H

Note: H is a symmetric matrix

$$\boldsymbol{H}(\boldsymbol{q}) = \sum_{i=1}^{n} m_i J_{L_i}^T J_{L_i} + J_{A_i}^T I_i J_{A_i}$$

Where $J_{L_i}, J_{A_i} I_i$ are written in the link's coordinates system.

$$\boldsymbol{H}(1,1) = \frac{L^2 m_2}{3} + L^2 m_3 + L^2 m_4 + \frac{L_1^2 m_3 s_2^2}{3} + d_3^2 m_3 s_2^2 + d_3^2 m_4 s_2^2 - L_1 d_3 m_3 s_2^2$$

$$\boldsymbol{H}(1,2) = \boldsymbol{H}(2,1) = -L d_3 c_2 (m_3 + m_4) + \frac{L c_2 L_1 m_3}{2}$$

$$\boldsymbol{H}(1,3) = \boldsymbol{H}(3,1) = -L s_2 (m_3 + m_4)$$

$$\boldsymbol{H}(2,2) = \frac{L_1^2 m_3}{3} + d_3^2 m_3 + d_3^2 m_4 - L_1 d_3 m_3$$

$$\boldsymbol{H}(2,3) = \boldsymbol{H}(3,2) = 0$$

$$\boldsymbol{H}(3,3) = m_3 + m_4$$

1.4 Computing C

$$C_{ij}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \sum_{k=1}^{n}\left(\frac{\partial H_{ij}}{\partial \theta_k} + \frac{\partial H_{ik}}{\partial \theta_j} - \frac{\partial H_{kj}}{\partial \theta_i}\right)\dot{\theta}_k$$

Note: We used a different notation for $sine$ and $cosine$ showing matrix C **only**.

$$In\ C\ matrix: s_{\theta_2} = \sin(\theta_2)$$

$$Usually: s_2 = \sin(\theta_2)$$

$$\boldsymbol{C}(1,1) = \dot{d}_3\left(d_3 m_3 s_{\theta_2}^2 + d_3 m_4 s_{\theta_2}^2 - \frac{L_1 m_3 s_{\theta_2}^2}{2}\right) + \dot{\theta}_2\left(\frac{L_1^2 m_3 s_{2\theta_2}}{6} + \frac{d_3^2 m_3 s_{2\theta_2}}{2} + \frac{d_3^2 m_4 s_{2\theta_2}}{2} - \frac{L_1 d_3 s_{2\theta_2}}{2}\right)$$

$$\boldsymbol{C}(1,2) = \dot{\theta}_1\left(\frac{L_1^2 m_3 s_{2\theta_2}}{6} + \frac{d_3^2 m_3 s_{2\theta_2}}{2} + \frac{d_3^2 m_4 s_{2\theta_2}}{2} - \frac{L_1 m_3 d_3 s_{2\theta_2}}{2}\right) - \dot{d}_3\left(Lc_{\theta_2}(m_3+m_4)\right) + \dot{\theta}_2\left(Ld_3 s_{\theta_2}(m_3+m_4) - \frac{Ls_{\theta_2} L_1 m_3}{2}\right)$$

$$C(1,3) = \dot{\theta}_1\left(d_3 s_{\theta_2}^2(m_3+m_4) - \frac{s_{\theta_2}^2 L_1 m_3}{2}\right) - \dot{\theta}_2\left(Lc_{\theta_2}(m_3+m_4)\right)$$

$$C(2,1) = -\dot{\theta}_1\left(s_{2\theta_2}\left(\frac{L_1^2 m_3}{6} + \frac{d_3^2 m_3}{2} + \frac{d_3^2 m_4}{2} - \frac{L_1 d_3 m_3}{2}\right)\right)$$

$$C(2,2) = \dot{d}_3\left(d_3(m_3+m_4) - \frac{L_1 m_3}{2}\right)$$

$$C(2,3) = \dot{\theta}_2\left(d_3(m_3+m_4) - \frac{L_1 m_3}{2}\right)$$

$$C(3,1) = -\dot{\theta}_1\left(d_3 s_{\theta_2}^2(m_3+m_4) - \frac{s_{\theta_2}^2 L_1 m_3}{2}\right)$$

$$C(3,2) = -\dot{\theta}_2\left(d_3(m_3+m_4) - \frac{L_1 m_3}{2}\right)$$

$$C(3,3) = 0$$

We tested our result by checking that $\dot{H} - 2C$ is a skew symmetric matrix [1]

1.5 Computing G

$$\boldsymbol{G}(\boldsymbol{q}) = -\sum_{k=1}^{n} m_i J_{L_i}^T g_i$$

Where J_{L_i}, g_i are written in the world coordinate system

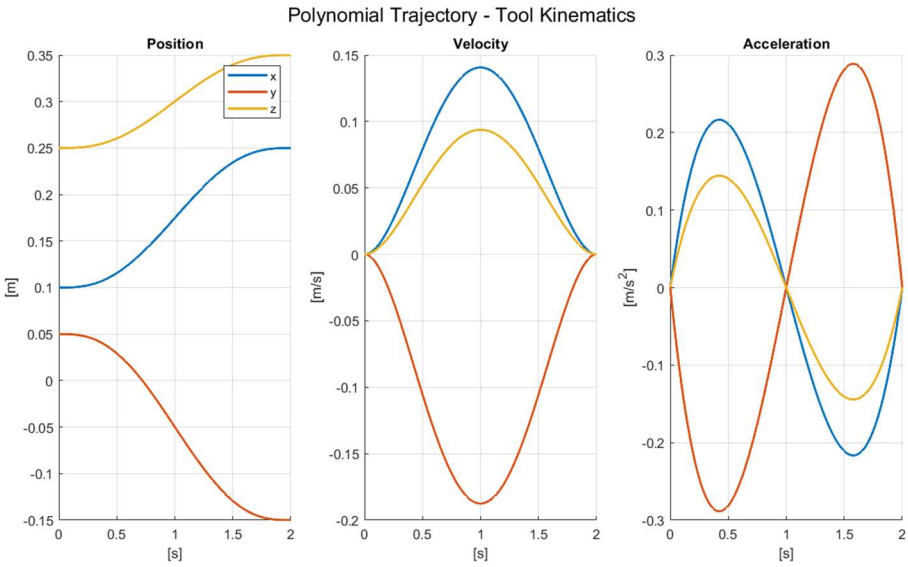
$$\boldsymbol{G} = \begin{bmatrix} 0 \\ -gs_2\left(d_3 m_3 + d_3 m_4 - \frac{L_1 m_3}{2}\right) \\ gc_2(m_3+m_4) \end{bmatrix}$$

2 Calculating $\tau(t)$ with Inverse Dynamics

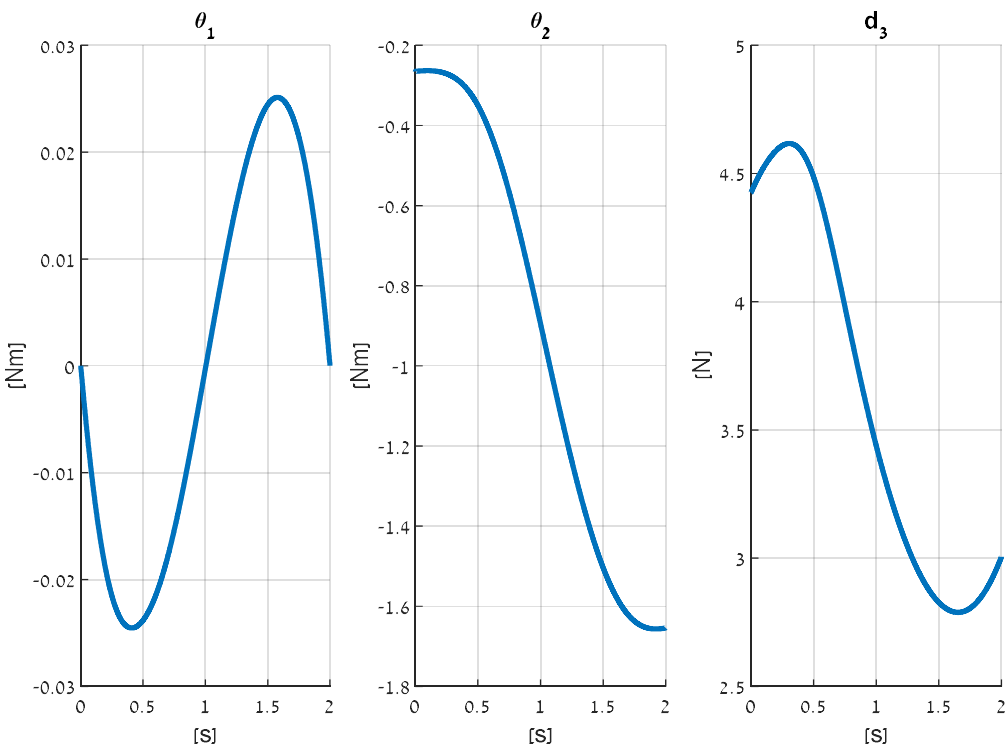
$$\boldsymbol{H\ddot{q}} + \boldsymbol{C\dot{q}} + \boldsymbol{G} = \boldsymbol{\tau} + \boldsymbol{J^T F_{ext}}$$

$$\boldsymbol{\tau} = \boldsymbol{H\ddot{q}} + \boldsymbol{C\dot{q}} + \boldsymbol{G} - \boldsymbol{J^T F_{ext}}$$

Provided no external forces and the polynomial trajectory presented below, we can calculate $\boldsymbol{q}, \boldsymbol{\dot{q}}, \boldsymbol{\ddot{q}}$ by applying the inverse kinematics (HW1), and sample $\boldsymbol{\tau}$.



Sampling at $dt = 1[ms]$, and deciding on joint $elbows = [1,1]$ (see HW1), we obtained the following required torque/forces from the joints:



3 Newton-Euler method for Inverse Dynamics and Reaction Forces.

3.1 Methods and Parameters

Unlike Lagrange-based methods, this method allows us to compute internal forces in the manipulator as it is based on applying Newton's second law for each link.

This method consists of two parts:

a) Front recursion, computed first, where we calculate the kinematics quantities
b) Back recursion, computed second, where we calculate the general forces between the links.

The computation process is provided in the references and is brought from the course's tutorial.

Parameters of importance are provided below for each link i

$$u_1 = [0 \quad 0 \quad 1]^T$$

$$u_2 = [1 \quad 0 \quad 0]^T$$

$$u_3 = [0 \quad 0 \quad 1]^T$$

$$u_3 = [0 \quad 0 \quad 0]^T$$

Vectors to centers of links from their origin

$$r_{1,c1} = \left[0 \quad 0 \quad \frac{H}{2}\right]^T$$

$$r_{2,c2} = \left[\frac{L}{2} \quad 0 \quad 0\right]^T$$

$$r_{3,c3} = \left[0 \quad 0 \quad d_3 - \frac{L_1}{2}\right]^T$$

$$r_{4,c4} = [0 \quad 0 \quad 0]^T$$

Vectors to end of links from their origin

$$r_{1,e1} = [0 \quad 0 \quad H]^T$$

$$r_{2,e2} = [L \quad 0 \quad 0]^T$$

$$r_{3,e3} = [0 \quad 0 \quad d_3]^T$$

$$r_{4,e4} = [0 \quad 0 \quad 0]^T$$

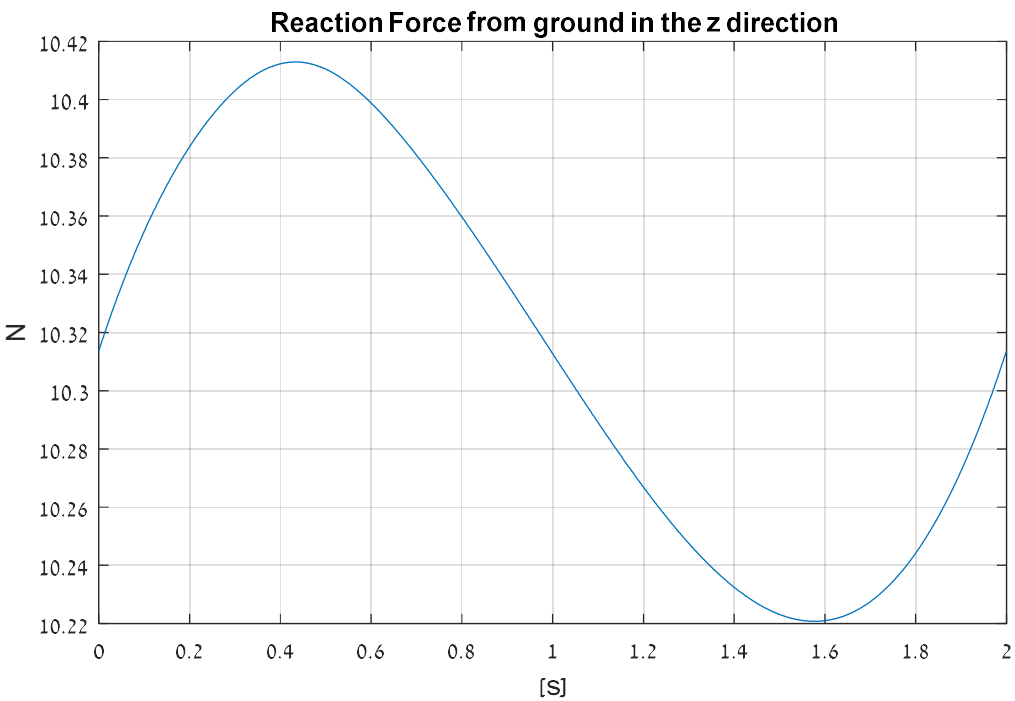
Rotation matrices between link systems

$${}^0R_1 = \begin{bmatrix} c_1 & -s_1 & 0 \\ s_1 & c_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad {}^1R_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_2 & -s_2 \\ 0 & s_2 & c_2 \end{bmatrix}$$

$${}^2R_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad {}^3R_4 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

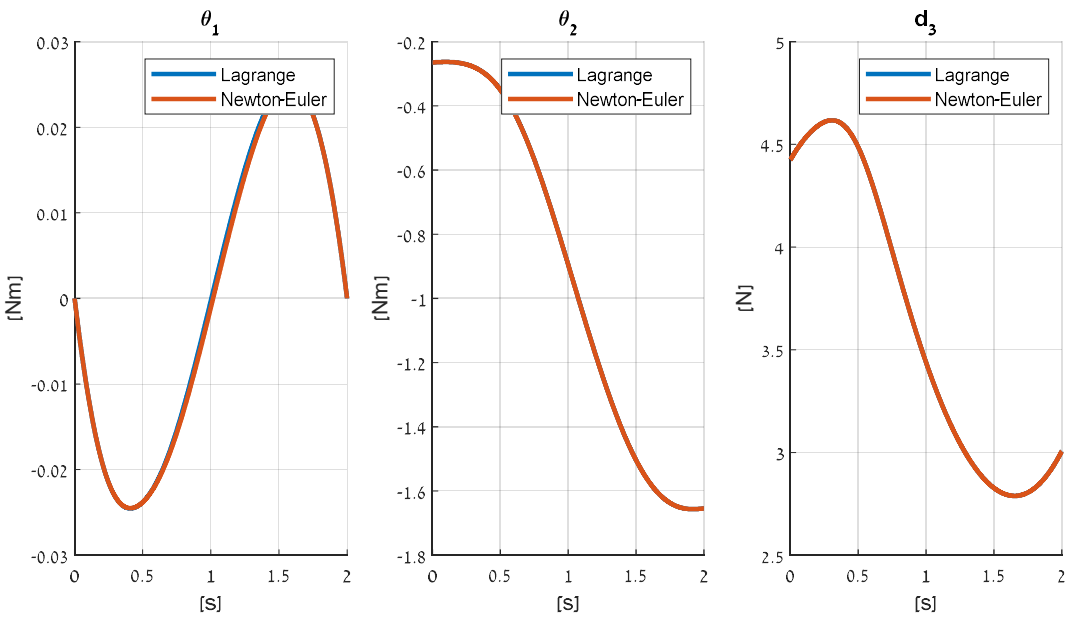
3.2 Solution Results – Reaction Force in the $\hat{z}$ direction

After running through the front and back recursive computations we received the reaction force applied on link 1 from the ground, in the $\hat{z}$ direction.



The force starts and ends at the same point - $10.31[N]$, which is the sum of all weights of the robot. Reasoning is that the robot is at rest in the start and end of the motion profile. Furthermore, the increase in the reaction force shows the acceleration of the mass away from the ground, while its decrease shows the opposite.

To validate our result, we compared the $\boldsymbol{\tau}$ computed in this section by the Newton-Euler method, with the one computed in the previous one with the Langrage approach:



4 Direct Dynamics

In the Direct Dynamics, we are to find the motion profile of the robot given torque/force profiles in the joints $\boldsymbol{\tau}$.

We set the state vector $\boldsymbol{X} = [\boldsymbol{q}; \dot{\boldsymbol{q}}]$.

$$\dot{\boldsymbol{X}} = [\dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}]$$

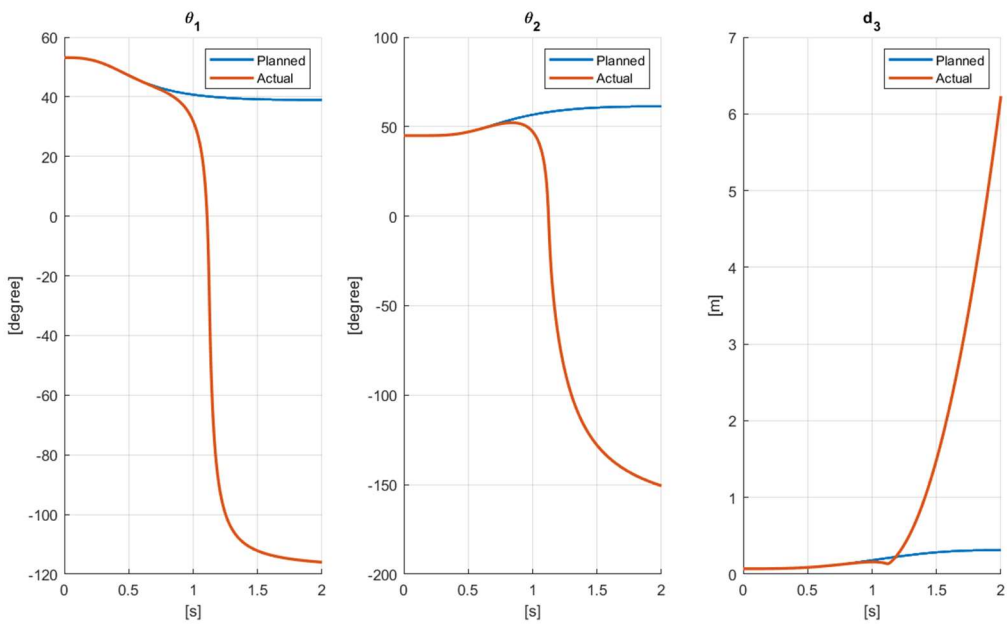
$$\dot{\boldsymbol{X}}(1:3) = \boldsymbol{X}(4:6)$$

$$\dot{\boldsymbol{X}}(4:6) = \ddot{\boldsymbol{q}} = \boldsymbol{H}^{-1}(\boldsymbol{\tau} + \boldsymbol{J}^{T}\boldsymbol{F}_{ext} - \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{G})$$

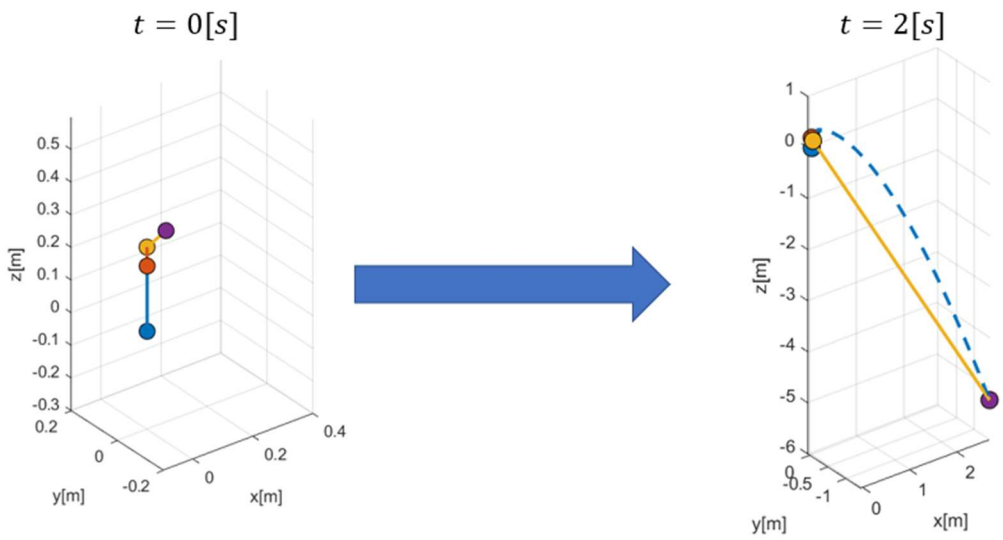
Creating a function *state_eq* such that $\dot{\boldsymbol{X}} = state\_eq(t, \boldsymbol{X})$, we attempted to solve for $\boldsymbol{X}$ using Matlab's ODE45 function. We have set the integration time of the simulation to $\frac{1}{10}$ of the joint torque/force computation sampling time (used in question 2)

$$t_{integration} = \frac{1}{10} t_{sample} = 0.1[ms].$$

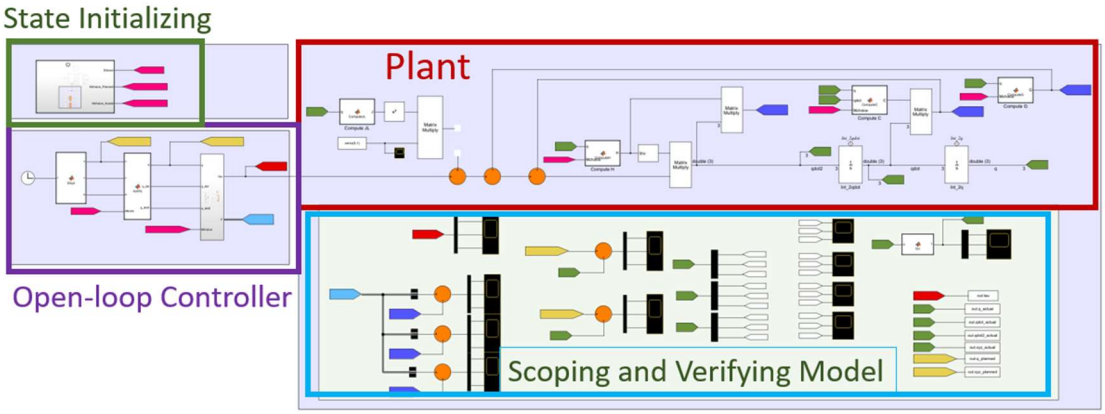
Between torque/force sample times we implemented linear interpolation for $\boldsymbol{\tau}$.



The open loop control we tried to implement failed us completely in the simulation.
To understand the problem, we drew the robot's state through time.

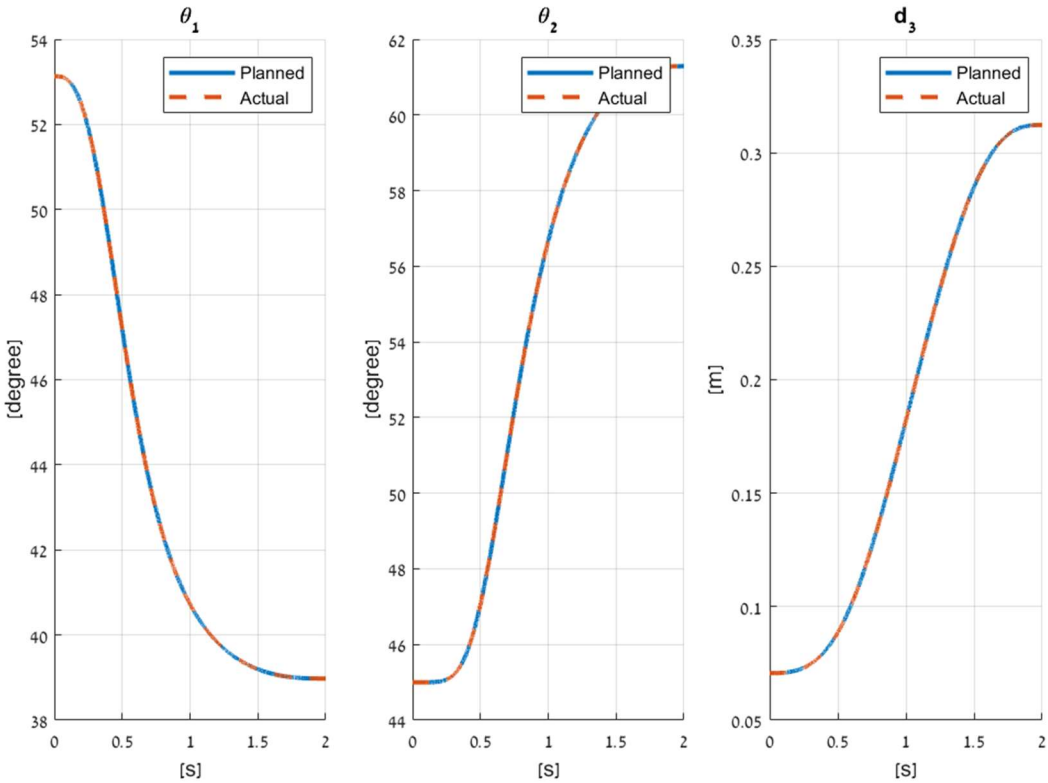


The second joint, θ_2 goes closer and closer to $-180°$ where the mass M can free-fall via the unconstraint joint d_3. We understood the system is very unstable and thus sensitive to small integration errors.
To overcome this problem, we created a detailed *Simulink* simulation of the entire system where $\boldsymbol{\tau}$ is computed on each time step



With harsh simulation settings, we came to successful results:

Solver	$ODE45$
Max Step Size	1^{-6}
Relative Tolerance	1^{-10}
Zero-Crossing Time Tolerance	eps

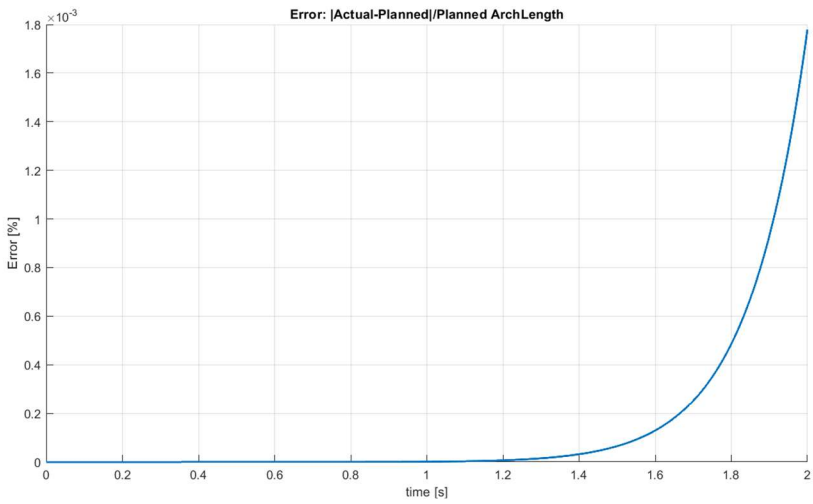


The joints follow their ordained paths computed in $Question\ 2$ via the open-loop control. To evaluate our performance, we computed the error in cartesian space as follows:

$$Err = \frac{\left|xyz_{planned}(t) - xyz_{actual}(t)\right|}{Total\ Arclength} \cdot 100\%$$

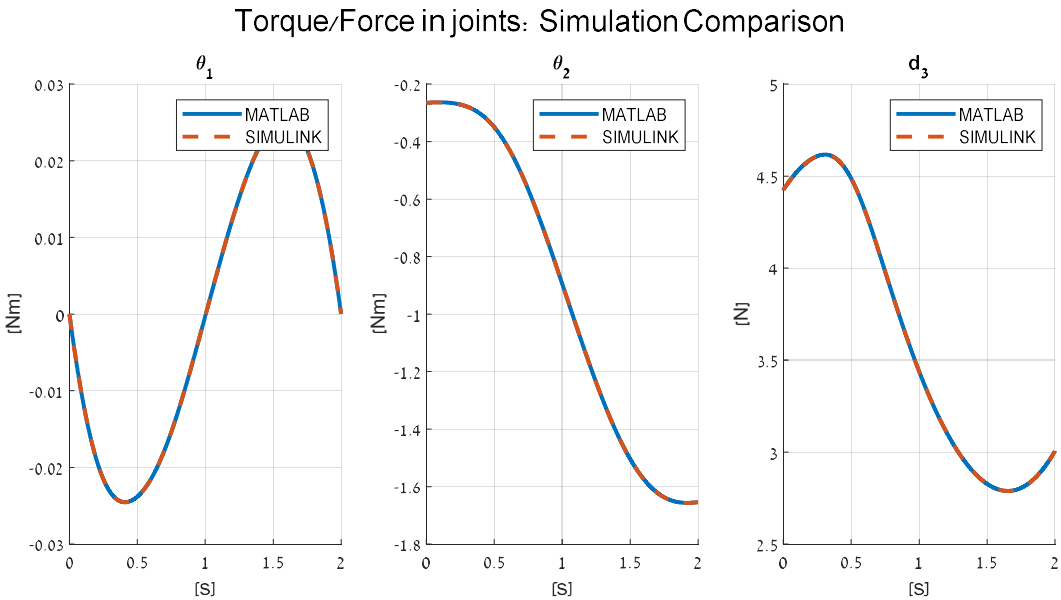
The maneuver is planned to be a straight line, hence:

$$Total\ Arclength = \sqrt{\left(x_f - x_0\right)^2 + \left(y_f - y_0\right) + \left(z_f - z_0\right)^2} = 0.2693[m]$$



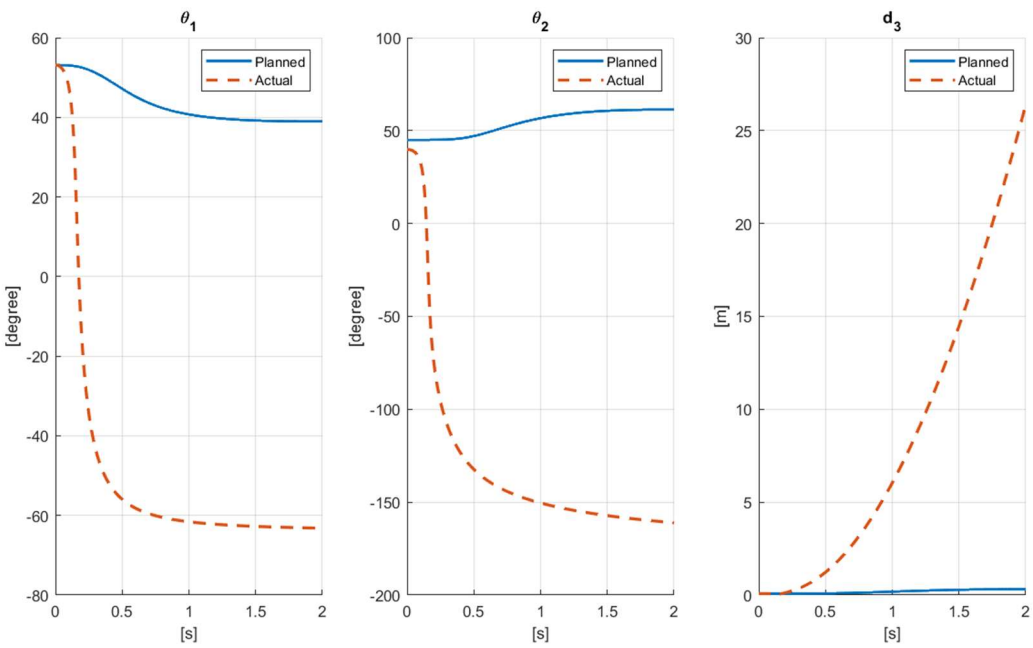
The cartesian error does not grow to be anything significant but is exponentially rising as expected from simulation of an unbalance system controlled by open-loop controller.

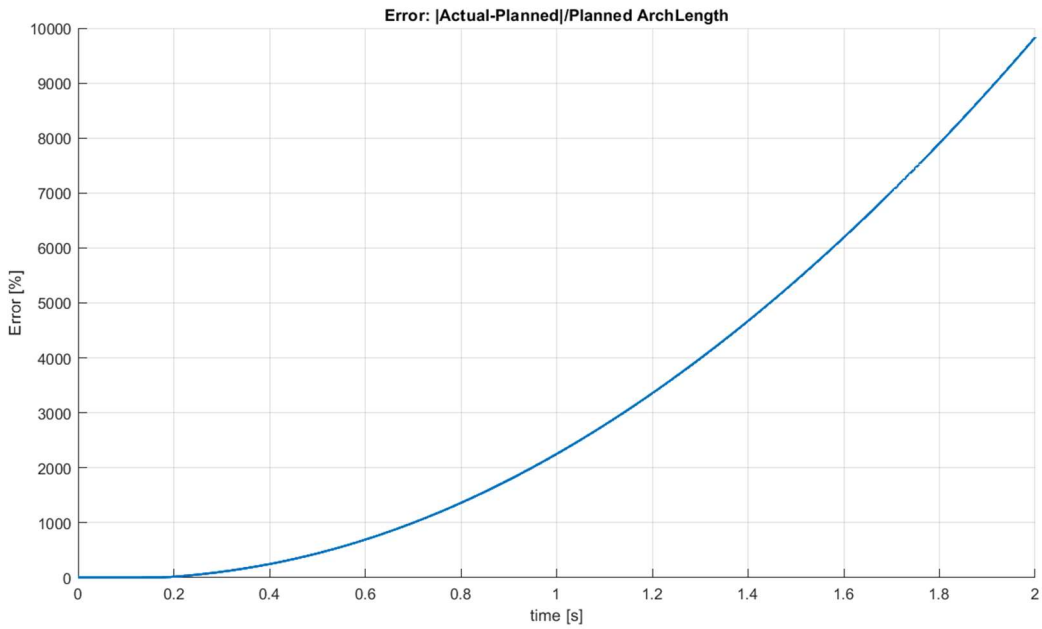
Lastly, we compared the two $\boldsymbol{\tau}$ vectors computed in the MATLAB's ODE45 simulation against the Simulink simulation to ensure that integration errors were indeed the cause of the first simulation failure.



5 Direct Dynamics – Different Initial Position

We ran our simulation again, this time changing the Initial position such that $z_0 := z_0 + 0.01[m]$, <u>without changing the torque applied to the joints.</u>

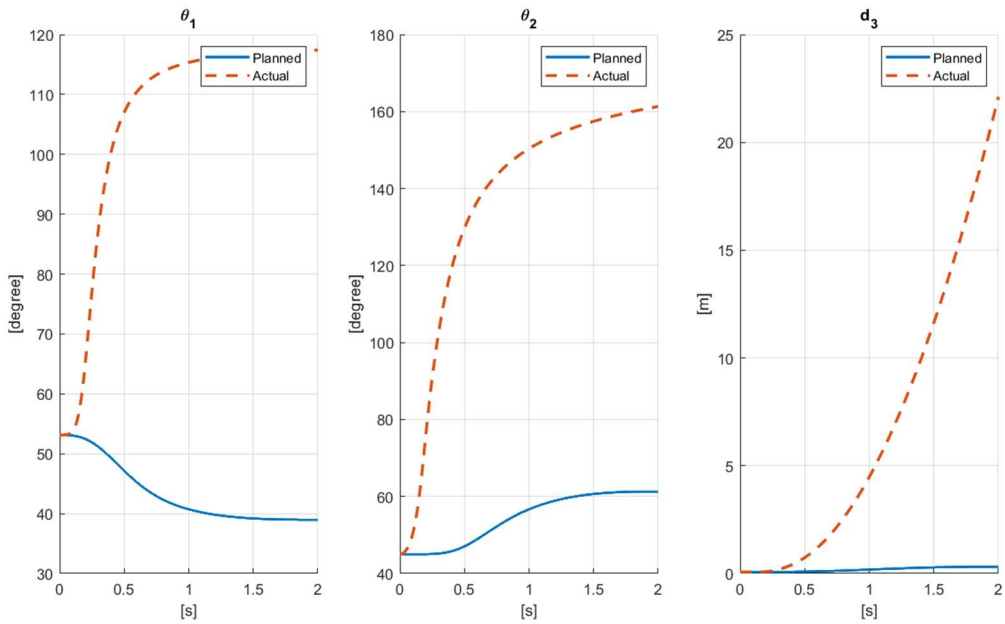


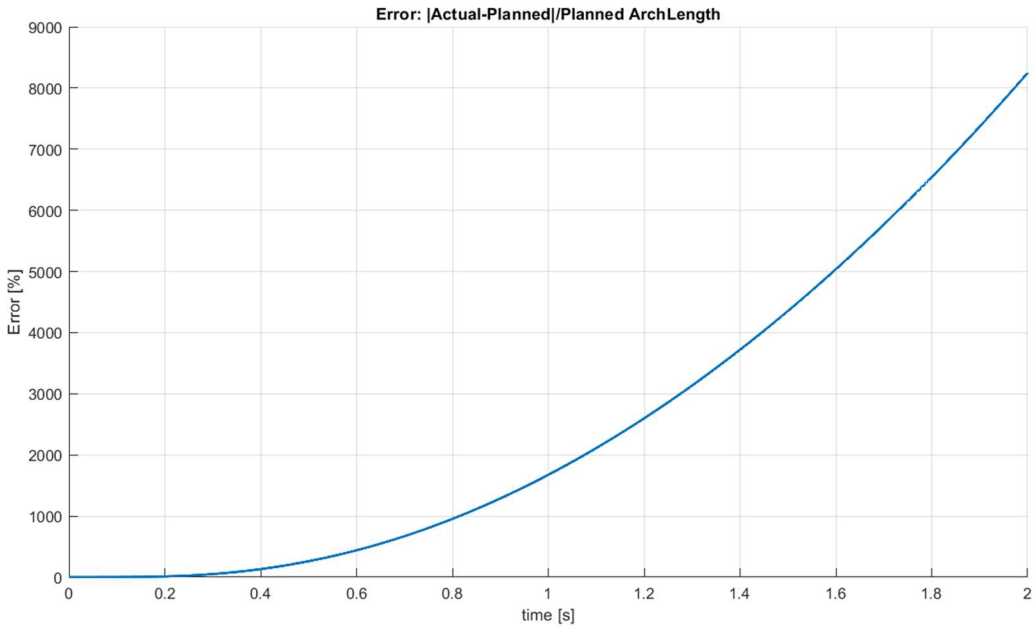


Just like the first simulation in $Question\ 4$, the second joint, θ_2 goes closer and closer to $-180°$ where the mass M can free-fall via the unconstraint joint d_3.
Our open loop controller <u>cannot</u> handle changes in initial position.

6 Direct Dynamics – Different Mass

Changing the mass held at the robot's tip to $M \coloneqq 0.6[Kg]$ (addition of $0.1[Kg]$), and <u>keeping the torque applied to the joints as they were in $Question4$:</u>





Unlike the previous case $Question$ 5, this time the second joint, θ_2, goes closer and closer to $180°$ from the positive side. The result, however, remains unchanged: The system diverges to a point where the mass M can free-fall via the unconstraint joint d_3. Our open loop controller cannot handle changes in tip's mass.

7 Conclusion

בתרגיל זה תיכננו את מומנטי הבקרה הדרושים להנעת הזרוע הרובוטית על פי מסלול ידוע מראש. בהתחלה פיתחנו את משוואות התנועה, למדנו בהרצאה שבאמצעות ניסוח משוואות לגרנג' ניתן לתכנן את מומנטי וכוחות המפרקים ללא ידיעת מכלול הכוחות הפועלים במערכת. למעשה, על ידי שימוש במטריצות היעקוביאן ניתן לפתור את המערכת בצורה קומפקטית ונוחה למדי.

קיבלנו את מומנטי הבקרה הנדרשים והרצנו עליהם סימולציות של דינמיקה ישירה – קרי, הזנו למערכת את המומנטים הנדרשים וציפינו לקבל את המסלול המתוכנן בתפסנית. ניסינו להריץ את הסימולציה באמצעות ODE45 אך הפתרונות התבדרו בכל פעם מחדש. נראה שתחילת ההתבדרות של הפתרון קורית לאחר שנייה אחת כאשר קצה התפסנית עובר מתאוצה לתאוטה ומפרק 1 משנה את כיוון תנועתו. ייתכן שנקודה זו יוצרת שגיאות נומריות בעייתיות. פתרנו זאת על ידי שימוש בסימולציה בסימולינק שם התוצאות לא סבלו מאותן שגיאות נומריות ואכן קיבלנו את תנועת התפסנית הרצויה.

באמצע הדרך גם ביצענו חישוב לשאר הכוחות הפועלים על מפרקי הרובוט על ידי משוואות ניוטון-אוילר. משימה זו היתה מעט סזיפית אך כפי שראינו בתרגיל מתקבלות תוצאות טובות שאף תואמות את האינטואיציה שלנו.

לסיום בדקנו מה קורה כאשר תנאי ההתחלה אינם תואמים לאלו שלפיהם תכננו את הוקטור מומנטי המפרקים. ראינו בסעיפים 5 ו-6 שהרובוט כלל לא מבצע מסלול הדומה לתכנון שלנו, ומסתבר שהאופן הזה של בקרה בחוג פתוח איננו יעיל ואפילו לשגיאות קטנות במידול הבעיה יש השפעה דרמטית על התוצאות.

מובן אם כן שנצטרך לבקר את התהליך בחוג סגור כך שאות הכניסה יהיה תואם בכל רגע למצב הנוכחי של הרובוט ולא על סמך תכנון מראש שלא לוקח בחשבון סטיות מהערכים שחושבו.

8 References

[1] R. M. Murray, Z. Li, S. S. Sastry, R. M. Murray, Z. Li, and S. S. Sastry, "Robot Dynamics and Control," *A Math. Introd. to Robot. Manip.*, pp. 155–210, 2018, doi: 10.1201/9781315136370-4.

8.1 Newton-Euler method

<u>קינמטיקה – "רקורסיה קדמית"</u>

חשוב: עובדים במערכת צמודת חוליה.

1. **מהירות זוויתית:** $\underline{\omega}_i = \left({}^{i-1}R_i\right)^T \underline{\omega}_{i-1} + \hat{u}_i \dot{q}_i$
   כאשר $\hat{u}_i$ הוא ציר מפרק i מבוטא במערכת i.
2. **תאוצה זוויתית:** $\underline{\alpha}_i = \left({}^{i-1}R_i\right)^T \underline{\alpha}_{i-1} + \hat{u}_i \ddot{q}_i + \underline{\omega}_i \times \hat{u}_i \dot{q}_i$
   עבור מפרק קוי $\dot{q}_i = \ddot{q}_i = 0$.
3. **תאוצת מרכז הכובד של חוליה i:**
   $$\underline{a}_{ci} = \left({}^{i-1}R_i\right)^T \underline{a}_{e,i-1} + \underline{\alpha}_i \times \underline{r}_{i,ci} + \underline{\omega}_i \times \left(\underline{\omega}_i \times \underline{r}_{i,ci}\right) + \hat{u}_i \ddot{d}_i + 2(\underline{\omega}_i \times \hat{u}_i)\dot{d}_i$$
   עבור מפרק סיבובי $\dot{d}_i = \ddot{d}_i = 0$.
   $\underline{a}_{e,i-1}$ - תאוצת קצה חוליה $i-1$.

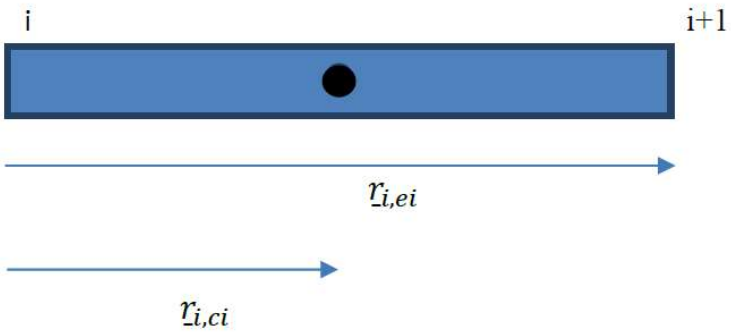
4. **תאוצת קצה חוליה i:**
   $$\underline{a}_{ei} = \left({}^{i-1}R_i\right)^T \underline{a}_{e,i-1} + \underline{\alpha}_i \times \underline{r}_{i,ei} + \underline{\omega}_i \times \left(\underline{\omega}_i \times \underline{r}_{i,ei}\right) + \hat{u}_i \ddot{d}_i + 2(\underline{\omega}_i \times \hat{u}_i)\dot{d}_i$$
   עבור מפרק סיבובי $\dot{d}_i = \ddot{d}_i = 0$.

- **תנאי התחלה** לחישוב רקורסיה קדמית: $\underline{\omega}_0 = \underline{\alpha}_0 = \underline{a}_{c,0} = \underline{a}_{e,0} = \vec{0}$
- $\underline{r}_{i,ci}$ - וקטור מבסיס חוליה i אל מרכז הכובד שלה מבוטא במערכת i.
- $\underline{r}_{i,ei} = \underline{r}_{i,i+1}$ - וקטור מבסיס חוליה i אל קצה החוליה מבוטא במערכת i.

<u>דינמיקה – "רקורסיה אחורית"</u>

1. **כוח שמפעילה חוליה $i+1$ על חוליה i:**
   $$\underline{f}_i = m_i \underline{a}_{ci} + {}^{i}R_{i+1} \underline{f}_{i+1} - m_i \underline{g}_i$$
   כאשר $\underline{g}_i = \left({}^{0}R_i\right)^T \underline{g}^{(0)}$.
2. **מומנטים שמפעילה חוליה $i+1$ על חוליה i:**
   $$\underline{M}_i = {}^{i}R_{i+1} \underline{M}_{i+1} + \underline{r}_{i,ci} \times \underline{f}_i + \left(\underline{r}_{i,ei} - \underline{r}_{i,ci}\right) \times \left({}^{i}R_{i+1} \underline{f}_{i+1}\right) + I_i \underline{\alpha}_i + \underline{\omega}_i \times (I_i \underline{\omega}_i)$$
   כאשר $\underline{r}_{i+1,ci} = \underline{r}_{i,ei} - \underline{r}_{i,ci}$ - וקטור מקצה חוליה i אל מרכז הכובד של חוליה i.

- **תנאי קצה:**
   $$ {}^{n}R_{n+1} \underline{f}_{n+1} = -\underline{F}_{ext}^{(n)}$$
   $$ {}^{n}R_{n+1} \underline{M}_{n+1} = -\underline{M}_{ext}^{(n)}$$



לפי הסימונים בהרצאה: $\left(\underline{r}_{i,ci} = \underline{r}_{c_i} - \underline{r}_i, \quad \underline{r}_{i,ei} = \underline{r}_{i+1} - \underline{r}_i\right)$