Project Report : Robot Learning and Control
Investigation of a Non-Anthromorphic Bipedal Robot

Saurabh Belgaonkar, Sajid Ahamed M A, Eshwar S R

June 19, 2021

1 Introduction

Humanoid robots, designed to mimic both the form and function of human beings, have been on the leading edge of robotics research. The end goal of the humanoid robot is to have generalist machine that could be used to perform tasks in humans environments, when the tasks are too dangerous, dull, or dirty for biological workers. Significant work has been done in the field of full sized humanoids using expensive high degree-of-freedom (DoF) platforms with complex control and rigid linkages. These platforms have seen drastic advancements in recent years, but are still years away from being practical due to concerns over complexity, cost, reliability, and safety. In order to have a biped platform that could be applicable today, a shift in the humanoid paradigm is necessary. Research on full sized humanoid robots tends to focus on improving their ability to work in environments designed for humans. This is often accomplished using rigidly controlled robots with two arms and two legs attached to a torso, and with 6 or more degrees of freedom in each limb. These robots are quite versatile, but are generally too slow, unsafe, and expensive to be used practically.

So, authors of the paper, instead of trying to tackle the deficiencies of bipedal walking with a novel control algorithm on a high DoF humanoid, attempted to tackle the issues by rethinking the fundamental design of a bipedal robot and humanoid. A biped does not necessarily need to have the exact same morphology or features of a human to perform human-centric tasks. The aim of the paper, which I am trying to replicate, was to develop a bipedal robot with a completely new form factor; a robot whose design can apply the control approaches common amongst bipedal robots today but with significantly enhanced agility and reliability. To this effect, the Non-Anthropomorphic Bipedal Robotic System (NABiRoS)(Figure1) was developed by authors: a robot with a novel lower body conifguration that resembles a normal humanoid robot walking sideways.

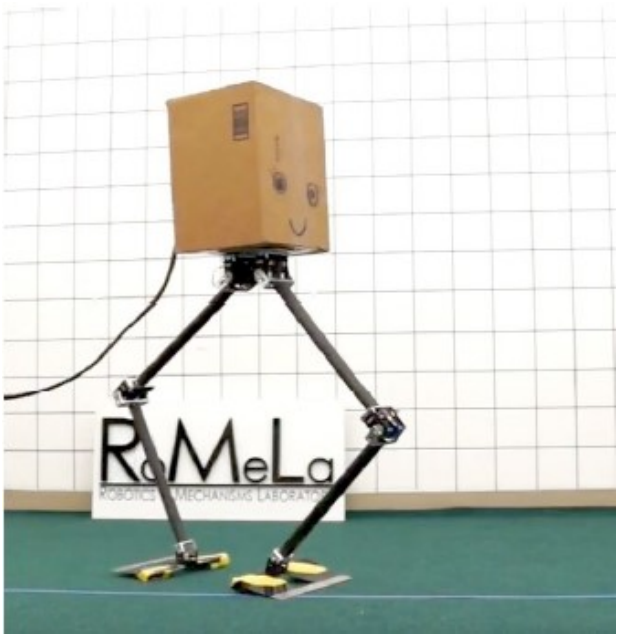


Figure 1: Non-Anthropomorphic Bipedal Robotic System (NABiRoS) prototype developed by RoMela lab

2 Concept and design

Bipedal walking is a complex, three-dimensional control and stability problem that humans often take for granted. One of the reasons why bipedal walking is difficult on classic humanoids is because of the offset in the hip joints in the frontal plane. In typical forward walking, this offset creates undesirable oscillatory moments that force the robot to lean in the direction orthogonal to the direction of motion. A small perturbation in this orthogonal direction can then easily destabilize the robot, and classical humanoid robots would struggle to recover from this sort of disturbance due to the restrictive workspace of the legs and torque-limited actuators. Therefore, these robots are forced to take small, calculated steps, as well as use a wide array of expensive force/torque and inertial sensors to perform simple walking or balancing tasks.

However, these moments dont appear when taking steps side-to-side (as the hip offset is in line with the direction of motion), and if the main mode of transport is sideways walking, the leg can be simplified significantly. The forward-facing, anthropomorphic knees are no longer necessary, and can be rotated such that the legs are aligned in a plane. By aligning the legs with the sagittal plane, stable forward walking can be achieved using the sideways walking motion (Figure 2).

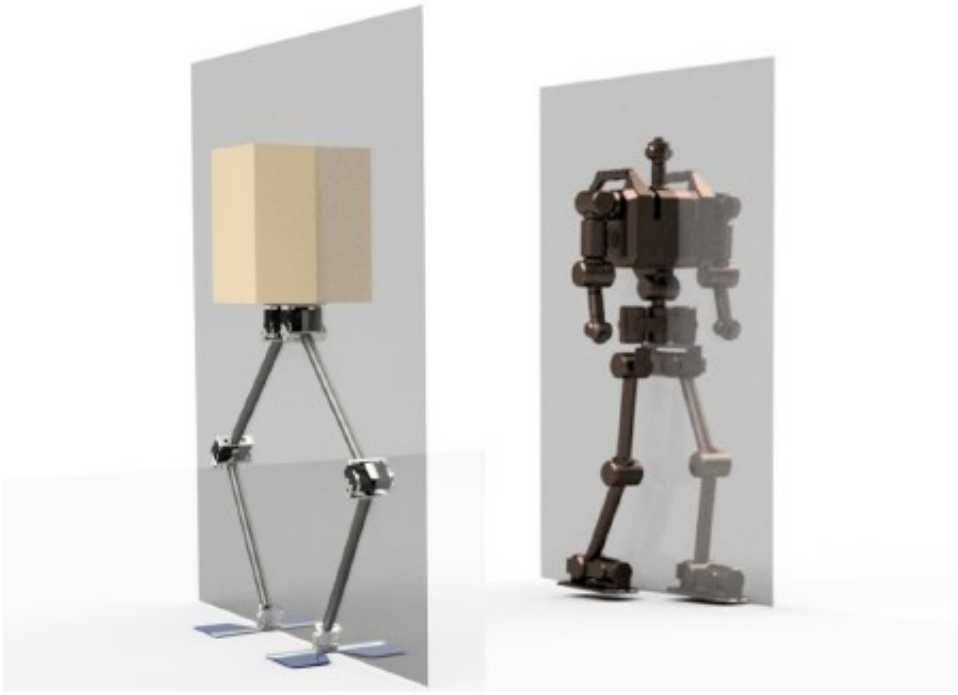


Figure 2: Comparison of the original NABiRoS (left) and a more traditional humanoid (right) that shows the sagittal plane of each.

This leg realignment also means the number of DoF in a leg can be significantly reduced, and the ankle can be removed and replaced a much simpler foot element. The novel configuration of NABiRoS simplifies the control and stability problem to a two dimensional planar one, and offers a different approach for a simple, lightweight, and practical bipedal robot.

NABiRoS achieves bipedal walking along the front-back axis with only two degrees of freedom on each leg; one at the hip, and one at the knee. There are no degrees of freedom at the ankles, significantly reducing the weight and moment of inertia of each leg. The robot developed by RoMeLa lab had leg links comprised of aluminum brackets and carbon fiber tubes. The ‘body’ of the robot rested above the hip joints and comprised of a box that covers the computing electronics. The materials used allowed robot to be extremely lightweight (3.88 kg, with external power).

3 Locomotion using Control

Traditional fully actuated bipeds have enough DoF in each leg to allow the foot to achieve an arbitrary spacial configuration within the leg's workspace. NABiRoS legs are restricted to a plane, meaning a minimum of three actuated DoF is necessary to be fully actuated. The replacement of the ankle joint for the compliant foot makes the system underactuated, in the sense that there are fewer actuators than available spacial degrees of freedom.Underactuation drastically complicates the control of robots in general, making the classical dynamics cancellation approaches used widely in manipulator arms challenging. Furthermore, the NABiRoS prototype is only equipped with position controlled actuators at each degree of freedom, and is not equipped with an inertial measurement unit (IMU) or force torque sensors.

Therefore, in order to command a walking motion with such limited sensory feedback and position controlled actuators, it is necessary to generate joint trajectories offine, and then 'play them back' on the robot. That is, generate a time-parameterized end-effector trajectory in Cartesian space, convert it to joint space using inverse kinematics, then execute it on the robot by looping at a constant rate and interpolating the joint positions based on execution time. This is one of the simplest approaches to control on a legged robot, yet can be still be effective if the robot has some sort of inherent stability. Also, RoMeLa lab has shown this approach to work on NABIRos in physical setting.

3.1 CAD model

For our work, we made a CAD model(Figure 3) in solidworks using the dimension and weight properties given in the paper. The CAD model was also converted to a URDF file, to input in the dynamics modelling software which don't except CAD models like Pybullet, Gazebo. The dimensions and the weight of the robots were taken as given in the paper and are as follows -

- **Base**(Hip) → **Weight-** 1.84kg and **Length between hip joints** - 0.1m
- **Femur**(Upper leg) → **Weight-** 0.48kg and **Length-** 0.3811m
- **Femur**(lower leg) → **Weight-** 0.44kg and **Length-** 0.3623m
- **Foot** → **Weight-** 0.1kg

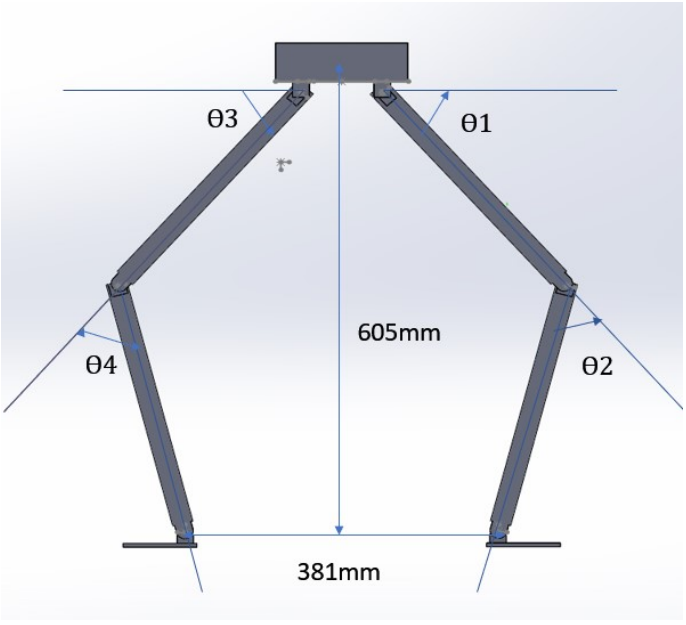


Figure 3: Front view of CAD model of NABI made in solidworks along with the distance between the feets and height of the center of mass of the base from the ground.

3.2 ZMP based walking

There are many approaches that can be used to generate walking trajectories, with new approaches like CPG(Central Pattern Generator), Reinforcement learning and the traditional approach like ZMP based approach. Here, a ZMP based approach is used to generate the trajectory of the center of mass of the robot. The key insight of this approach is the use of the simplified cart-table model, and then applying a preview controller to track a prescribed ZMP trajectory.

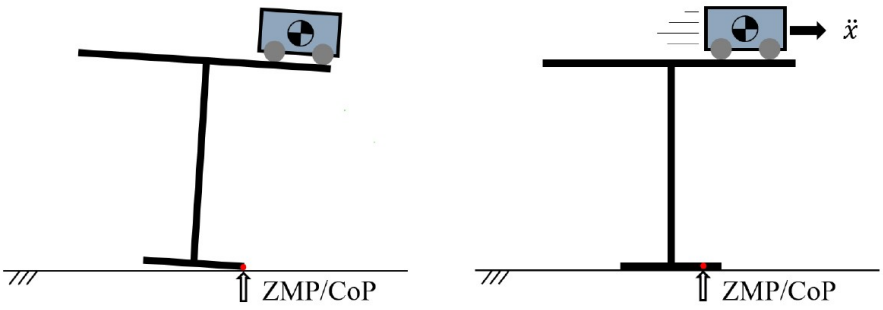


Figure 4: Intuition on the cart table model: if the cart is over the edge of the table base and not moving (left), the ZMP is at the edge of the table base and the table will start tipping over. If the cart is accelerating at the edge of the table (right), the reaction force on the table produces a moment that keeps the ZMP within the table base, preventing tipping.

The cart-table model can be described by a cart that is sliding on the surface of a massless table, with the at base of the table representing the support polygon generated by the stance foot (feet). The model describes how a cart moving on the surface of the table effects the Zero Moment Point (ZMP) at the table base, as shown in Figure4. This model is illustrated in more detail in Figure 5 which also show how the model is embedded into the NABiRoS system. The dynamics of this model can now be represented written out concisely as:

$$p = x - \frac{h}{g} * \ddot{x}$$

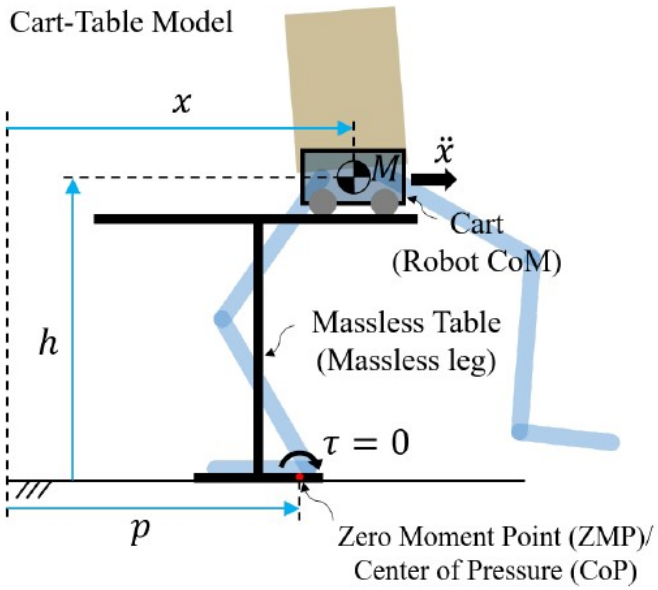


Figure 5: Simplified cart-table model used for ZMP based locomotion planning on NABiRoS. The cart represents the center of mass of the robot, and the table base is the stance foot of the robot.

With a corresponding representation in state space:

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \\ \dddot{x} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u$$

$$p = \begin{bmatrix} 1 & 0 & -\frac{h}{g} \end{bmatrix} \begin{bmatrix} x \\ \dot{x} \\ \ddot{x} \end{bmatrix}$$

Where p is the ZMP or Center of Pressure (CoP), x describes the horizontal component of the Center of Mass (CoM) of the robot (or the cart), h is the height of the CoM which remains constant, and g is the gravitational constant. Note that in the state space representation the system is augmented so that the system input u is the jerk of the CoM. Now, this model can be used with a preview controller that utilizes future information of the ZMP reference trajectory in order to yield a smooth CoM trajectory. The system in discrete form with sample time t can be written as:

$$x_{k+1} = x_k + 1$$

$$p_k = cx_k$$

Where

$$\boldsymbol{x}_k = \begin{bmatrix} x(k\Delta t) & \dot{x}(k\Delta t) & \ddot{x}(k\Delta t) \end{bmatrix}^T$$

$$u_k = u(k\Delta t)$$

$$p_k = p(k\Delta t)$$

$$\boldsymbol{A} = \begin{bmatrix} 1 & \Delta t & \Delta t^2 \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} \frac{\Delta t^3}{6} \\ \frac{\Delta t^2}{2} \\ \Delta t \end{bmatrix}$$

$$\boldsymbol{c} = \begin{bmatrix} 1 & 0 & -\frac{h}{g} \end{bmatrix}$$

Now, to track a reference trajectory, we minimize the following quadratic cost:

$$J = \sum_{j=1}^{\infty} \{Q(p_j^{ref} - p_j)^2 + Ru_j^2\}$$

Where Q and R are positive gains on the error in state and input, respectively. For a preview controller, cost J is minimized by the following input:

$$u_k = -\boldsymbol{K}\boldsymbol{x}_k + \begin{bmatrix} f_1 & f_2 & \dots & f_N \end{bmatrix} \begin{bmatrix} p^{ref}_{k+1} \\ p^{ref}_{k+2} \\ \vdots \\ p^{ref}_{k+N} \end{bmatrix}$$

Where

$$\boldsymbol{K} = (R + \boldsymbol{b}^T \boldsymbol{P} \boldsymbol{b})^{-1} \boldsymbol{b}^T \boldsymbol{P} \boldsymbol{A}$$

$$f_i = (R + \boldsymbol{b}^T \boldsymbol{P} \boldsymbol{b})^{-1} [(\boldsymbol{A} - \boldsymbol{b}\boldsymbol{K})^{(i-1)}]^T \boldsymbol{c}^T Q$$

And P is the solution to the discrete Riccati Equation:

$$P = A^T PA + c^T Qc - A^T Pb(R + b^T Pb)^{-1} 1 b^T PA$$

The control law described by comprises of the normal optimal state feedback law plus an additional feed-forward term which is the inner product of N future reference points gained by weights [f1, f2, ...fN]. Thus, it can be said that the preview controller can look into the future, and act accordingly.

To implement this result on NABi-1, the following steps are taken:

- Prescribe foot placements (P0; P1; :::; Pn) which the robot will take i.e steps size taken are defined.Theoretically, the steps can come from a high level planner and can be made as large as the leg kinematics would allow for while maintaining a constant hip height, but for NABiRoS they were manually assigned and made small in order to reduce the effects of the leg dynamics.
- Define a ZMP trajectory p(t) through the support polygon that is produced by the prescribed foot placements. For NABiRoS, the ZMP trajectory was generated by linearly interpolating between the centers of the support polygon at every step.
- Utilize the ZMP preview controller described above to track the desired ZMP trajectory, outputting a CoM trajectory in the process.
- With the location of the feet and CoM known for a specified number of steps (and time), the robot legs can be commanded to track the trajectory via inverse kinematics (for joint space).
- An additional motion is necessary to be implemented for the swing leg for taking steps. For NABiRoS, the swing foot is commanded to follow a cycloid function which can help reduce the impact when contacting the ground.

In an ideal situation, the trajectory that is executed on the robot should perform the walking motion perfectly, but in reality this is never the case. Typically, there is an additional stabilizing feedback controller that is used to ensure adequate tracking of the foot, ZMP, and CoM. These are enabled by the use of additional sensors such as an IMU that can output center of mass accelerations or F/T sensors at the ankles to estimate the actual robot ZMP. However, NABiRoS does not have any of these forms of feedback, so the trajectories produced from this approach are directly executed,and the ZMP trajectory, foot placements, and step timing were tuned to produce a motion that was stable over at (and even slightly uneven) ground.

The main tuning parameters for this locomotion method on NABiRoS are

- The height of the table which determines how much forward-back swing occurs, and this is set by deciding what should be the initial joint angles of the robot when it is at rest.
- The duration of the single support phase.
- The step size, if the step size is too large, then the robot will not be able to maintain constant height.

the duration of the single support and double support phases which are tuned to accommodate the natural oscillations of the spring feet.

3.3 Result

So, a CAD model was developed in solidworks, and was also exported as URDF to be used with other dynamic software. A preview controller was developed in MATLAB to generate a center of mass trajectory for 10 secs based on the reference trajectory that was given. The step size was taken as 0.2m and time for support phase was taken as 0.7 sec. The result of this are shown in figure6. The trajectory of the center of mass generated is smooth.

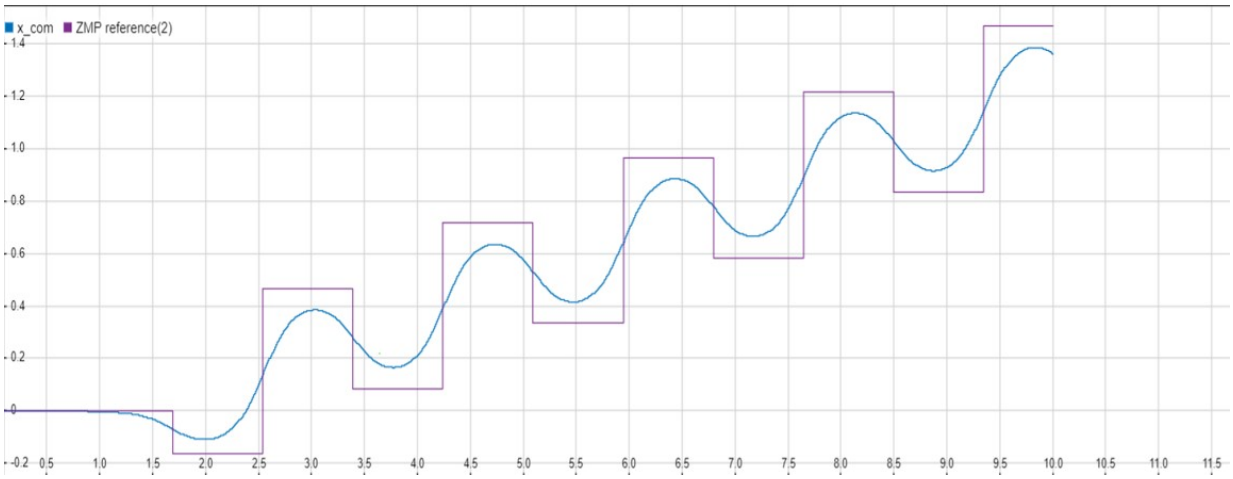


Figure 6: The purple color line denotes the ZMP reference trajectory, and the blue color line denotes the generated COM trajectory

Now, this center of mass trajectory was used to find the joints angles of support leg using inverse kinematics.Also, the joint angles of the front legs where found out using making the foot follow a cycloidal trajectory in ground reference frame. Also, there will be two phases. One, when the back leg is supporting, and the other when the front leg is supporting. This has been taken into account in the code, and a continuous time varying joint angles were generated as shown in figures7,8,9,10. From this figures it could be seen that variation of joint angles with respect to time is smooth.

Now, to check whether the joint angles variation is correct, and is working as intended. A kinematic simulation was done in MATLAB on the model that was made in MATLAB. A video was generated, and check to see whether it is working as intended. Everything was working as intended. The figure below are some of the snapshots from the video. However, we were not able to dynamically simulate the walking of robot in the simulator.

Click here for animated video

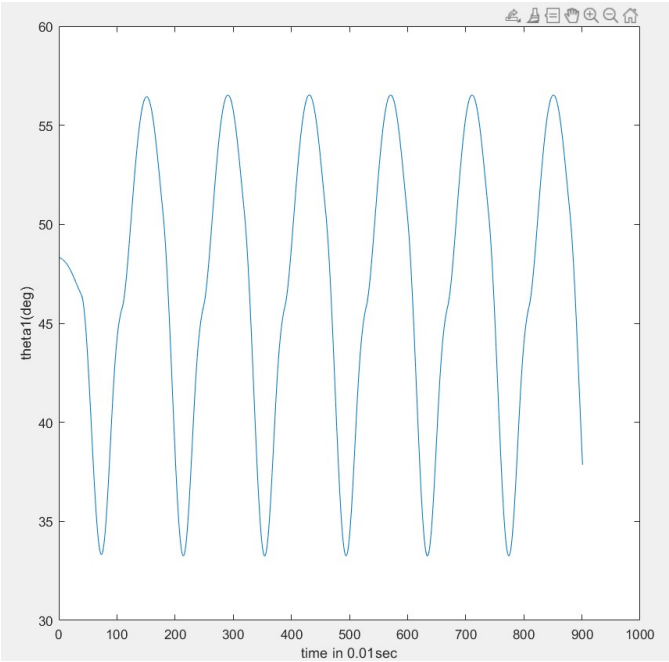


Figure 7: Variation of Theta1 in deg with time

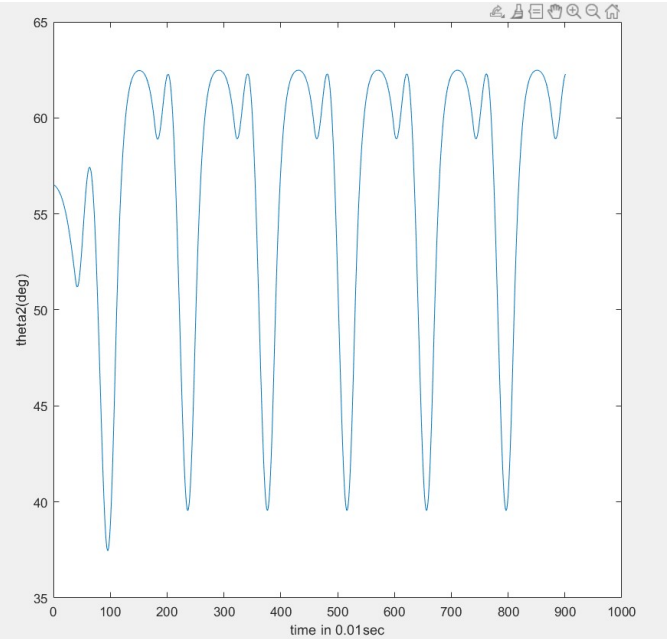


Figure 8: Variation of Theta2 in deg with time

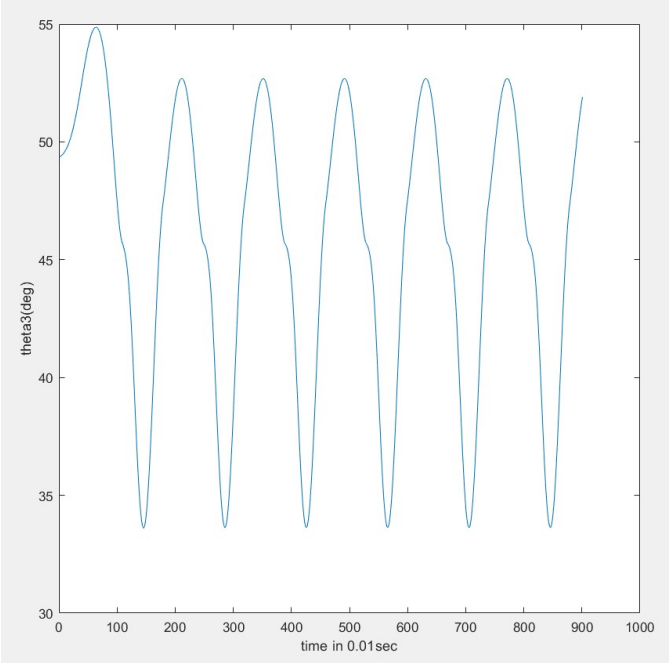


Figure 9: Variation of Theta3 in deg with time

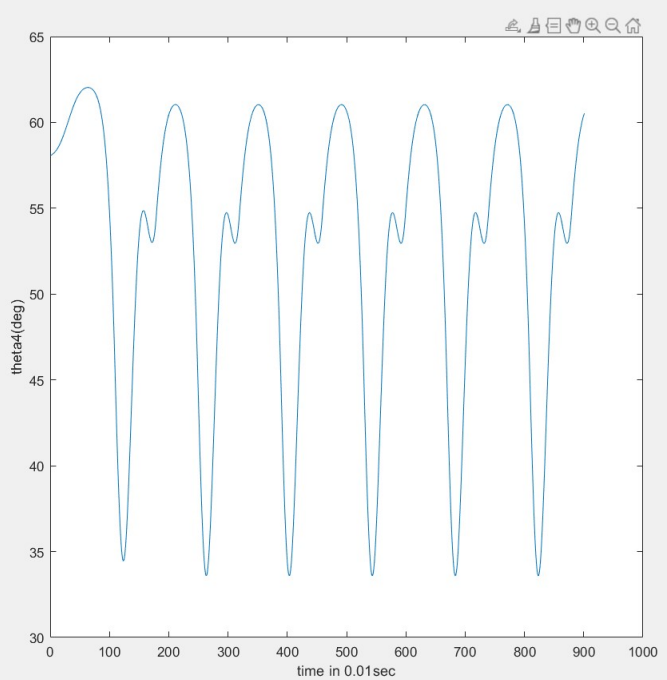


Figure 10: Variation of Theta4 in deg with time

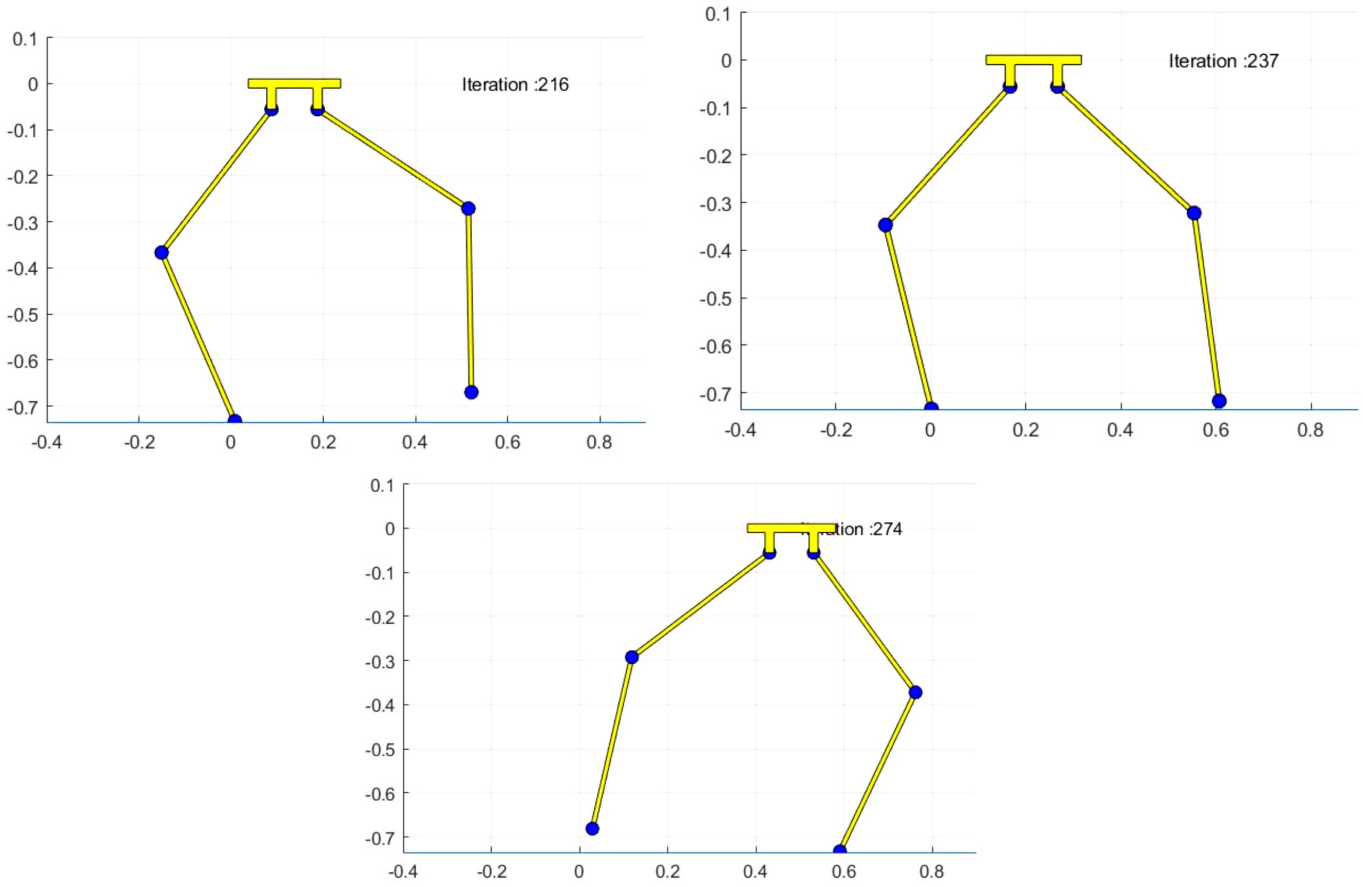


Figure 11: Snapshot of kinematic video

4 Locomotion via Reinforcement Learning

Reinforcement learning has been used extensively in real-world applications like robot locomotion, etc. Hence, the motivation use reinforcement learning to learn the locomotion of the Nabi robot. Rajeshwaran et al, (5) have shown the a linear policies are good enough to learn robot locomotion and are comparable to complex neural network based state-of-the-art techniques. Hence, we also chose to work with linear policy for our work. With this background we chose the ARS(4) algorithm.

4.1 Goal

The goal of this part of the project was to make the Nabi Robot walk on a flat surface. And also extend it to make it walk on a wedge.

4.2 Model

We chose to run the simulations on MuJoCo simulator. Hence, we had to come up with the MJCF XML file. We found a reference file online, but it was faulty. We had to tweak it to make it work. Also, we added the wedge into the environment.

Next, the agent wanted a way to interact with the simulator, so we created an OpenAI gym environment interface in python for the agent to interact with the MuJoCo simulator. As we all know, well defined states, actions and reward function is necessary for the agent to learn via RL technique. All these state definitions, reward function definitions were incorporated in the Gym interface.

Finally, the agent was trained using the ARS algorithm written in python. And the learnt policy was used to generate a video of the robot walking on the wedge.

4.3 State, Action, Reward function and Policy

The state was represented using a 21-dimension vector. The first 3 components were the x,y,z coordinates of the base. Next 3 components were the euler angle representations of the base orientation. Next 4 components were the angles at the hips and knees. Followed by 6 components of transitional and angular velocoties of the base. Followed by 4 components of angular velocities of the hips and knees. The last component was a constant 1, ie the bias term.

The action space is a 4-dimensional vector. Which are the 4 torques to be applied at the hips and knees joints respectively.

Reward function consisted of 4 components. Firstly, it is directly proportional to the forward velocity. We also awarded a survival reward of 3 for every time step. The robot is supposed to walk in a straight line, hence any deviation attracted a penalty. And finally it also included the actuator cost.

We considered a linear deterministic policy. That is a matrix M is pre-multiplied to the state vector to obtain the action vector.

4.4 Training and Results

We use the MJCF model and the OpenAI Gym interface to learn an agent using the ARS V2-t algorithm. ARS stands for "Augmented Random Search". It is an extension to the existing famous Random Search. ARS uses a linear policy and the objective is to maximize the total rewards achieved in the episode. And the parameters are updated using the empirical derivative computed using the central difference method. i.e for a function $y = f(x)$, the central difference derivative can be computed numerically as $\nabla y \equiv \frac{f(x+\delta)-f(x-\delta)}{2\delta}$. In the Basic Random Search method, the parameter is perturbed with N small noises(directions) and perturbed policies are run once to get an estimate of the value of the policy. Then an average of all the derivatives using all the N directions is considered as an estimate of derivative. In the Augmented Random Search, the authors proposed to use only top "b" direction for computing the derivatives. Though the Evolutionary Strategies/Random Search methods were shown to work very well in practice, until recent times there was no mathematical analysis/guarantees. Then authors of (6) provided some bounds for the same.

Below are the hyperparameters used to learn the policy:

- N = 16
- b = 8
- lr = 0.02
- noise = 0.03
- episode length = 2000
- angle of wedge = 7 degrees

With the above configurations, the agent was able to learn to complete the walking on the wedge. It took order of 10^7 timesteps for the agent to learn to walk. This roughly took 20 - 30 mins. We trained it on a 12 core machine with i7 processors(3.5GHz). Click here for animated visualisation of learnt walking

Algorithm 2 Augmented Random Search (ARS): four versions **V1**, **V1-t**, **V2** and **V2-t**

1: **Hyperparameters:** step-size α, number of directions sampled per iteration N, standard deviation of the exploration noise ν, number of top-performing directions to use b ($b < N$ is allowed only for **V1-t** and **V2-t**)
2: **Initialize:** $M_0 = \mathbf{0} \in \mathbb{R}^{p \times n}$, $\mu_0 = \mathbf{0} \in \mathbb{R}^n$, and $\Sigma_0 = \mathbf{I}_n \in \mathbb{R}^{n \times n}$, $j = 0$.
3: **while** ending condition not satisfied **do**
4: Sample $\delta_1, \delta_2, \ldots, \delta_N$ in $\mathbb{R}^{p \times n}$ with i.i.d. standard normal entries.
5: Collect $2N$ rollouts of horizon H and their corresponding rewards using the $2N$ policies

$$\textbf{V1:} \begin{cases} \pi_{j,k,+}(x) = (M_j + \nu\delta_k)x \\ \pi_{j,k,-}(x) = (M_j - \nu\delta_k)x \end{cases}$$

$$\textbf{V2:} \begin{cases} \pi_{j,k,+}(x) = (M_j + \nu\delta_k)\,\mathrm{diag}\,(\Sigma_j)^{-1/2}\,(x - \mu_j) \\ \pi_{j,k,-}(x) = (M_j - \nu\delta_k)\,\mathrm{diag}(\Sigma_j)^{-1/2}(x - \mu_j) \end{cases}$$

for $k \in \{1, 2, \ldots, N\}$.
6: Sort the directions δ_k by $\max\{r(\pi_{j,k,+}), r(\pi_{j,k,-})\}$, denote by $\delta_{(k)}$ the k-th largest direction, and by $\pi_{j,(k),+}$ and $\pi_{j,(k),-}$ the corresponding policies.
7: Make the update step:

$$M_{j+1} = M_j + \tfrac{\alpha}{b\sigma_R} \sum_{k=1}^{b} \left[r(\pi_{j,(k),+}) - r(\pi_{j,(k),-})\right] \delta_{(k)},$$

where σ_R is the standard deviation of the $2b$ rewards used in the update step.
8: **V2** : Set μ_{j+1}, Σ_{j+1} to be the mean and covariance of the $2NH(j+1)$ states encountered from the start of training.[2]
9: $j \leftarrow j + 1$
10: **end while**

In the above section, two strategies for locomotion were introduced, one based on ZMP approach wherein kinematic walking was demonstrated and another based on Reinforcement learning wherein dynamic walking was demonstrated in Mujoco environment. Each strategy strategy has its own pros and cons. Though ZMP trajectory is simple to generate, it is an open loop strategy and doesn't accommodate the real-time effects such as terrain changes, impact disturbances, orthogonal plane gravity gradient and moreover may not always be feasible. On other hand Reinforcement learning strategy though has been learned taking into consideration the above said real-time effects, learning the strategy itself is a tedious process involving simulation over hundred thousand episodes. Now the question that arises is how good is the RL strategy learnt walking gait compared to its ZMP counterpart which is a standard desirable walking gait in the sagittal plane. Moreover can the ZMP walking model be some how incorporated so as to come up with a model based RL method to minimise the number of learning episodes. This required a common platform for testing both strategies in which ZMP could perform fairly well and this motivated us to develop our own dynamics simulator wherein we could limit the real-world disturbances and non-linear effects to sagittal plane. Also development of this simulator would helps us in an in-depth understanding of closed chain and parallel manipulator mechanics of NABIROS. This platform would also enable evaluation of the suitability of approximate models such as SLIP (Spring Load Inverted Pendulum) which could serve as reference models for model based Reinforcement learning. With this in mind we study the control and dynamics of NABIROS.

5 Control and Dynamics of NABIROS

It is emphasised here again that the walking gait based on ZMP approach is an open loop strategy and is referred to as static walking in literature. On contrary, the RL based walking gait is dynamic in nature, generated based on taking actions as dictated by a policy that maximizes a reward dependent on the desired and the actual state and hence referred to as dynamic walking. Irrespective of strategy being chosen, the output of these strategies are actions ($\tau_1,\ \tau_2,\ \tau_3,\ \tau_4$) so as to achieve the desired joint angles ($\theta_1,\ \theta_2,\ \theta_3,\ \theta_4$). These control actions when fed as input to the robot dynamics ($M(q)\,\ddot{q} + C(q,\dot{q}) + G(q) = \tau$, where q is

the generalized state vector of the robot and τ is the input vector to the system which is a vector of control inputs taking into consideration effect of other inputs such as frictional forces, normal reactions, moments acting on the robot), the robot exhibits the desired walking motion as output. In general the choice of generalized state vector is the position and orientation of COM, the joint angles and their derivatives. The control architecture for implementing the above in simulator is shown in the figure 12.

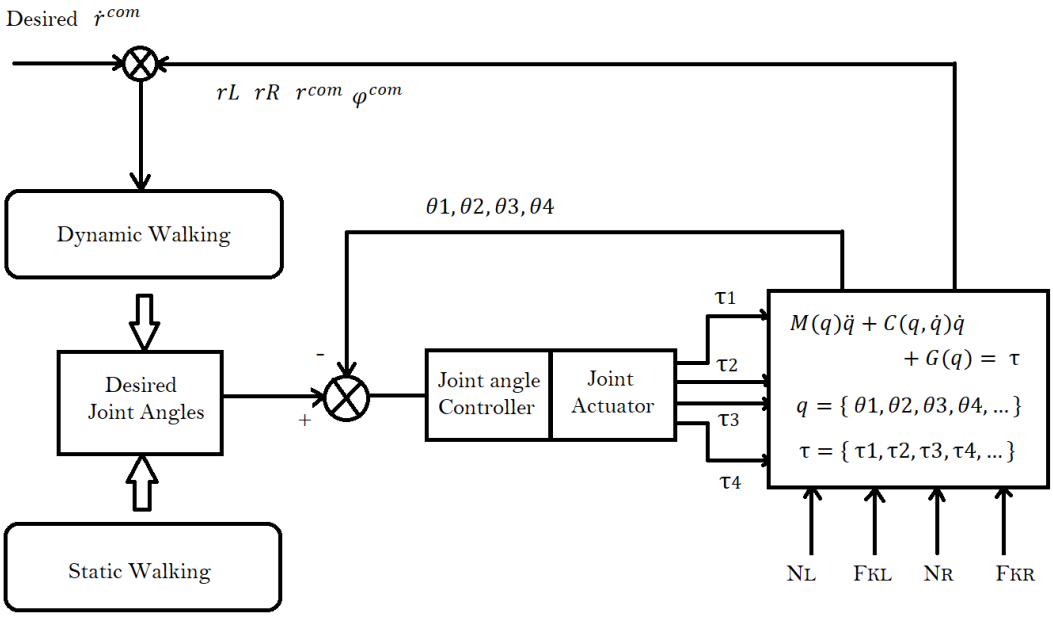


Figure 12: Control Architecture for NABIROS

Based on the above control architecture the action vector (τ_1, τ_2, τ_3, τ_4) is obtained by PD implementation based on error in the desired and actual joint angle trajectories. The next stage is feeding this control inputs to the dynamics of the robot, Mass matrix form $M(q)\ \ddot{q} + C(q, \dot{q}) + G(q) = \tau$ of which are derived following laws of physics based on Newtonian and Lagrangian mechanics. The free body diagram of the forces and moments acting on the robot are as depicted in the figure 13.

The mass - inertia parameters of the robot considered are as follows :

- M = 1.40983/2; Mass of base link
- mT = 0.23236; Mass of tibia
- mF = 0.49995; Mass of femur
- m = 0.10898; Mass of foot
- Ix = 0.00475204; Inertia of base link about x axis
- IxF = 0.0080336; Inertia of femur about x axis
- IxT = 0.0044393; Inertia of tibia about x axis
- lT = 0.39857; Length of tibia
- lF = 0.39152; Length of femur
- d = 0.1; Seperation at the hip joint
- h = 0.0558; height of base link COM from hip joint
- g = 9.84; acceleration due to gravity

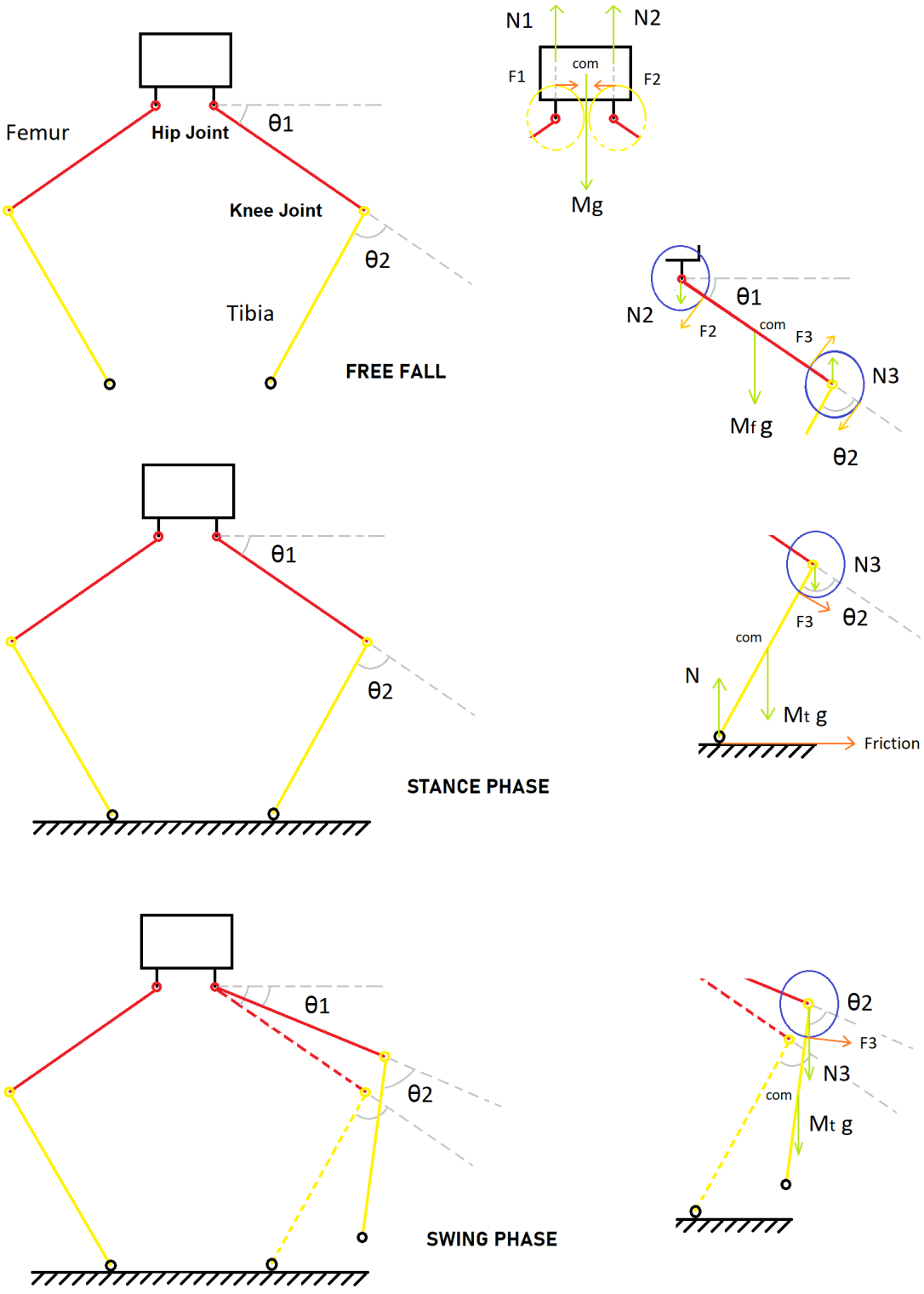
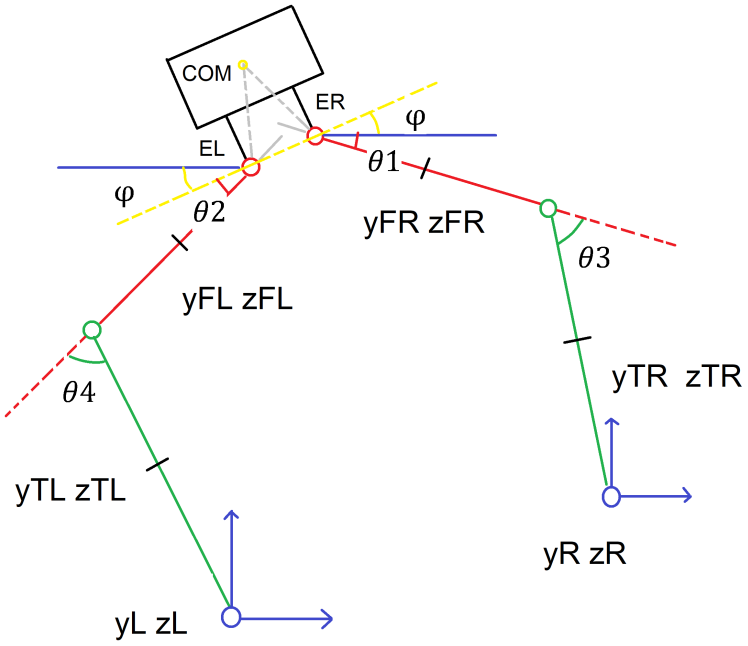


Figure 13: Free body diagram of NABIROS under free fall, stance and swing stage

The Y-Z plane is considered as the sagittal plane of the robot. Ideal approach for writing down the Lagrangian equations of motion (EOM) for the robot is by first identifying appropriate generalized coordinates.In this robot since the Normal reaction and frictional forces are acting when the feet touch the ground, position of both feet (yL, zL, yR, zR) along with the respective left leg and right leg joint angles ($\theta 2$, $\theta 4$, $\theta 1$, $\theta 3$) are chosen as generalised coordinates. If the ($yCOM$, $zCOM$) and ϕ represent the position and orientation of COM respectively then the forward kinematic equations are given as below :



$$
\begin{aligned}
\text{yTR} &= yR - 0.5\ lT\ cos(\phi+\theta_1+\theta_3)\\
\text{zTR} &= zR - 0.5\ lT\ sin(\phi+\theta_1+\theta_3)\\
\text{yFR} &= yR - lT\ cos(\phi+\theta_1+\theta_3) - 0.5\ lF\ cos(\phi+\theta_1)\\
\text{zFR} &= zR - lT\ sin(\phi+\theta_1+\theta_3) - 0.5\ lF\ sin(\phi+\theta_1)\\
\text{yER} &= yR - lT\ cos(\phi+\theta_1+\theta_3) - lF\ cos(\phi+\theta_1)\\
\text{zER} &= zR - lT\ sin(\phi+\theta_1+\theta_3) - lF\ sin(\phi+\theta_1)
\end{aligned}
$$

$$
\begin{aligned}
\text{yTL} &= yL + 0.5\ lT\ cos(\phi+\theta_2+\theta_4)\\
\text{zTL} &= zL + 0.5\ lT\ sin(\phi+\theta_2+\theta_4)\\
\text{yFL} &= yL + lT\ cos(\phi+\theta_2+\theta_4) + 0.5\ lF\ cos(\phi+\theta_1)\\
\text{zFL} &= zL + lT\ sin(\phi+\theta_2+\theta_4) + 0.5\ lF\ sin(\phi+\theta_1)\\
\text{yEL} &= yL + lT\ cos(\phi+\theta_2+\theta_4) + lF\ cos(\phi+\theta_1)\\
\text{zEL} &= zL + lT\ sin(\phi+\theta_2+\theta_4) + lF\ sin(\phi+\theta_1)
\end{aligned}
$$

where yTR, zTR, yFR, zFR, yTL, zTL, yFL, zFL are coordinates of COM's of tibia and femur link of respective legs and yER, zER, yEL, zEL are coordinates of end effector (hip joints) of respective legs. The rigid body constraint between coordinates of end effectors of respective legs in terms of coordinates of COM of base link are given as :

$$
\mathcal{F}\begin{cases}
\begin{pmatrix}\text{yEL}\\ \text{zEL}\end{pmatrix} = \begin{pmatrix}\text{yCOM}\\ \text{zCOM}\end{pmatrix} + \begin{pmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi\end{pmatrix}\begin{pmatrix}-d/2\\ -h\end{pmatrix}\\[2ex]
\begin{pmatrix}\text{yER}\\ \text{zER}\end{pmatrix} = \begin{pmatrix}\text{yCOM}\\ \text{zCOM}\end{pmatrix} + \begin{pmatrix} cos\phi & -sin\phi \\ sin\phi & cos\phi\end{pmatrix}\begin{pmatrix}d/2\\ -h\end{pmatrix}
\end{cases}
$$

With the above forward kinematic equations, taking their derivative and finding kinetic energy and potential energy of the system due to entire link masses considered at respective link COM's, we could compute the lagrangian as $\mathcal{L} = K.E - P.E$. The dynamic EOM of robot under the constraints of end effector motion $\mathcal{F}$ are then given by :

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) - \frac{\partial \mathcal{L}}{\partial q} + \sum_{i=1}^{C} \lambda_i \frac{\partial \mathcal{F}}{\partial q} = 0$$

$$\text{where , } \; q = (\;\; \theta 2,\; \theta 4,\; \theta 1,\; \theta 3,\; yL,\; zL,\; yR,\; zR\;)^{\mathrm{T}}$$

However solving the lagrangian and further simplification into Mass matrix form was found to be cumbersome, hence an alternate approach for deriving the EOM was adopted. By splitting mass at COM of base link into two halves, each half now assumed to concentrated at end effector of each leg and then considering each leg as a subsystem, we take advantage of the symmetry about orthogonal plane of the robot. Lagrangian EOM for each subsystem were for found and for introducing the constrained dynamics between both end effectors we make use of Newtonian mechanics which says every action has an equal and opposite reaction. Based on this assumption the EOM are derived as follows :

RIGHT LEG LAGRANGIAN FORMULATION

$$p = (\; \theta_1,\; \theta_3,\; yR\;,\; zR\;)^{\mathrm{T}}$$

$$KE_R = 0.5\; m\; (\dot{y}R^2 + \dot{z}R^2) + 0.5\; mT\; (dyTR^2 + dzTR^2) + 0.5\; IxT\; (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3})^2$$
$$+ 0.5\; mF\; (dyFR^2 + dzFR^2) + 0.5\; IxF\; (\dot{\phi} + \dot{\theta_1})^2 + 0.5\; M\; (dyER^2 + dzER^2) + 0.5\; Ix\; \dot{\phi}^2$$
$$PE_R = (m\; zR + mT\; zTR + mF\; zFR + M\; zER)\; g$$

$$\mathcal{L}_R = KE_R - PE_R$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{\theta_1}}\right) - \frac{\partial \mathcal{L}_R}{\partial \theta_1} = 0 \qquad \frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{\theta_3}}\right) - \frac{\partial \mathcal{L}_R}{\partial \theta_3} = 0$$
$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{y}R}\right) - \frac{\partial \mathcal{L}_R}{\partial yR} = 0 \qquad \frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{z}R}\right) - \frac{\partial \mathcal{L}_R}{\partial zR} = 0$$

$$D11 = IxF + IxT + (M + 0.25\; mF)\; lF^2 + (M + mF + 0.25\; mT)\; lT^2 + 2\; (M + 0.5\; mF)\; lF\; lT\; cos(\theta_3)$$
$$D12 = IxT + (M + mF + 0.25\; mT)\; lT^2 + (M + 0.5\; mF)\; lF\; lT\; cos(\theta_3)$$
$$D13 = (M + 0.5\; mF)\; lF\; sin(\phi + \theta_1) + (M + mF + 0.5\; mT)\; lT\; sin(\phi + \theta_1 + \theta_3)$$
$$D14 = -(M + 0.5\; mF)\; lF\; cos(\phi + \theta_1) - (M + mF + 0.5\; mT)\; lT\; cos(\phi + \theta_1 + \theta_3)$$

$$C11 = 0$$
$$C12 = (-2\; M\; (\dot{\phi} + \dot{\theta_1}) - mF\; (\dot{\phi} + \dot{\theta_1} + 0.5\; \dot{\theta_3}))\; lT\; lF\; sin(\theta_3)$$
$$C13 = 0$$
$$C14 = 0$$

$$G1 = (-(M + 0.5\; mF)\; lF\; cos(\phi + \theta_1) - (M + mF + 0.5\; mT)\; lT\; cos(\phi + \theta_1 + \theta_3))\; g +$$
$$\ddot{\phi}\; (IxF + IxT + (M + 0.25\; mF)\; lF^2 + (M + mF + 0.25\; mT)\; lT^2 + 2\; (M + 0.5\; mF)\; lF\; lT\; cos(\theta_3))$$

$$D21 = IxT + (M + mF + 0.25\ mT)\ lT^2 + (M + 0.5\ mF)\ lF\ lT\ cos(\theta_3)$$
$$D22 = IxT + (M + mF + 0.25\ mT)\ lT^2$$
$$D23 = (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3)$$
$$D24 = -(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3)$$

$$C21 = (M + 0.5\ mF)\ lF\ lT\ sin(\theta_3)\ \dot{\theta_1}$$
$$C22 = 0$$
$$C23 = 0$$
$$C24 = 0$$

$$G2 = (-(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3))\ g + \ddot{\phi}\ (IxT + (M + mF + 0.25\ mT)\ lT^2 + (M + 0.5\ mF)\ lF\ lT\ cos(\theta_3)) + \dot{\phi}\ ((M + 0.5\ mF)\ lF\ lT\ sin(\theta_3)\ (\dot{\phi} + 2\ \dot{\theta_1}))$$

$$D31 = (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3) + (M + 0.5\ mF)\ lF\ sin(\phi + \theta_1)$$
$$D32 = (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3)$$
$$D33 = M + m + mT + mF$$
$$D34 = 0$$

$$C31 = (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3}) + (M + 0.5\ mF)\ lF\ cos(\phi + \theta_1)\ (\dot{\phi} + \dot{\theta_1})$$
$$C32 = (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3})$$
$$C33 = 0$$
$$C34 = 0$$

$$G3 = 0 + \ddot{\phi}\ ((M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3) + (M + 0.5\ mF)\ lF\ sin(\phi + \theta_1)) + \dot{\phi}\ ((M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3}) + (M + 0.5\ mF)\ lF\ cos(\phi + \theta_1)\ (\dot{\phi} + \dot{\theta_1}))$$

$$D41 = -(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3) - (M + 0.5\ mF)\ lF\ cos(\phi + \theta_1)$$
$$D42 = -(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3)$$
$$D43 = 0$$
$$D44 = M + m + mT + mF$$

$$C41 = (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3}) + (M + 0.5\ mF)\ lF\ sin(\phi + \theta_1)\ (\dot{\phi} + \dot{\theta_1})$$
$$C42 = (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3})$$
$$C43 = 0$$
$$C44 = 0$$

$$G4 = (M + m + mT + mF)\ g + \ddot{\phi}\ (-(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_1 + \theta_3) - (M + 0.5\ mF)\ lF\ cos(\phi + \theta_1)) + \dot{\phi}\ ((M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_1 + \theta_3)\ (\dot{\phi} + \dot{\theta_1} + \dot{\theta_3}) + (M + 0.5\ mF)\ lF\ sin(\phi + \theta_1)\ (\dot{\phi} + \dot{\theta_1}))$$

$$D_{right} = \begin{pmatrix} D11 & D12 & D13 & D14 \\ D21 & D22 & D23 & D24 \\ D31 & D32 & D33 & D34 \\ D41 & D42 & D43 & D44 \end{pmatrix} \quad C_{right} = \begin{pmatrix} C11 & C12 & C13 & C14 \\ C21 & C22 & C23 & C24 \\ C31 & C32 & C33 & C34 \\ C41 & C42 & C43 & C44 \end{pmatrix} \quad G_{right} = \begin{pmatrix} G1 & G2 & G3 & G4 \end{pmatrix}^{\mathrm{T}}$$

LEFT LEG LAGRANGIAN FORMULATION

$$q = (\ \theta_2,\ \theta_4,\ yL\ ,\ zL\ )^{\mathrm{T}}$$

$$KE_L = 0.5\ m\ (\dot{y}L^2 + \dot{z}L^2) + 0.5\ mT\ (dyTL^2 + dzTL^2) + 0.5\ IxT\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4})^2$$
$$+ 0.5\ mF\ (dyFL^2 + dzFL^2) + 0.5\ IxF\ (\dot{\phi} + \dot{\theta_2})^2 + 0.5\ M\ (dyEL^2 + dzEL^2) + 0.5\ Ix\ \phi^2$$
$$PE_L = (m\ zL + mT\ zTL + mF\ zFL + M\ zEL)\ g$$

$$T_L = KE_L - PE_L$$

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{\theta_2}}\right) - \frac{\partial \mathcal{L}_R}{\partial \theta_1} = 0 \quad \frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{\theta_4}}\right) - \frac{\partial \mathcal{L}_R}{\partial \theta_3} = 0$$
$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{y}R}\right) - \frac{\partial \mathcal{L}_R}{\partial yL} = 0 \quad \frac{d}{dt}\left(\frac{\partial \mathcal{L}_R}{\partial \dot{z}R}\right) - \frac{\partial \mathcal{L}_R}{\partial zL} = 0$$

$$D11 = IxF + IxT + (M + 0.25\ mF)\ lF^2 + (M + mF + 0.25\ mT)\ lT^2 + 2\ (M + 0.5\ mF)\ cos(\theta_4)$$
$$D12 = IxT + (M + mF + 0.25\ mT)\ lT^2 + (M + 0.5\ mF)\ lF\ lT\ cos(\theta_4)$$
$$D13 = -(M + 0.5\ mF)\ lF\ sin(\phi + \theta_2) - (M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)$$
$$D14 = (M + 0.5\ mF)\ lF\ cos(\phi + \theta_2) + (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)$$

$$C11 = 0$$
$$C12 = (-2\ M\ (\dot{\phi} + \dot{\theta_1}) - mF\ (\dot{\phi} + \dot{\theta_2} + 0.5\ \dot{\theta_4}))\ lT\ lF\ sin(\theta_4)$$
$$C13 = 0$$
$$C14 = 0$$

$$G1 = ((M + 0.5\ mF)\ lF\ cos(\phi + \theta_2) + (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4))\ g$$
$$+ \ddot{\phi}\ (IxF + IxT + (M + 0.25\ mF)\ lF^2 + (M + mF + 0.25\ mT)\ lT^2 + 2\ (M + 0.5\ mF)\ lF\ lT\ cos(\theta_4))$$

$$D21 = IxT + (M + mF + 0.25\ mT)\ lT^2 + (M + 0.5\ mF)\ lF\ lT\ cos(\theta_4)$$
$$D22 = IxT + (M + mF + 0.25\ mT)\ lT^2$$
$$D23 = -(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)$$
$$D24 = (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)$$

$$C21 = (M + 0.5\ mF)\ lF\ lT\ sin(\theta_4)\ \dot{\theta_2}$$
$$C22 = 0$$
$$C23 = 0$$
$$C24 = 0$$

$$G2 = ((M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4))\ g + \ddot{\phi}\ (IxT$$
$$+ (M + mF + 0.25\ mT)\ lT^2 + (M + 0.5\ mF)\ lF\ lT\ cos(\theta_4)) + \dot{\phi}\ ((M + 0.5\ mF)\ lF\ lT\ sin(\theta_4)\ (\dot{\phi} + \dot{\theta_2}))$$

$$D31 = -(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4) - (M + 0.5\ mF)\ lF\ sin(\phi + \theta_2)$$
$$D32 = -(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)$$
$$D33 = M + m + mT + mF$$
$$D34 = 0$$

$$C31 = -(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4}) - (M + 0.5\ mF)\ lF\ cos(\phi + \theta_2)\ (\dot{\phi} + \dot{\theta_2})$$
$$C32 = -(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4})$$
$$C33 = 0$$
$$C34 = 0$$

$$G3 = 0 + \ddot{\phi}\ (-(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4) - (M + 0.5\ mF)\ lF\ sin(\phi + \theta_2))$$
$$+ \dot{\phi}\ (-(M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4}) - (M + 0.5\ mF)\ lF\ cos(\phi + \theta_2)\ (\dot{\phi} + \dot{\theta_2}))$$

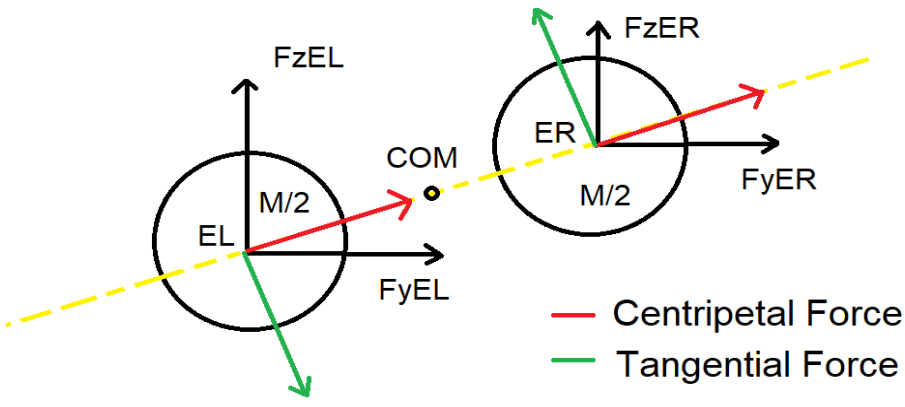
$$D41 = (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4) + (M + 0.5\ mF)\ lF\ cos(\phi + \theta_2)$$
$$D42 = (M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4)$$
$$D43 = 0$$
$$D44 = M + m + mT + mF$$

$$C41 = -(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4}) - (M + 0.5\ mF)\ lF\ sin(\phi + \theta_2)\ (\dot{\phi} + \dot{\theta_2})$$
$$C42 = -(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4})$$
$$C43 = 0$$
$$C44 = 0$$

$$G4 = (M + m + mT + mF)\ g + \ddot{\phi}\ ((M + mF + 0.5\ mT)\ lT\ cos(\phi + \theta_2 + \theta_4) + (M + 0.5\ mF)\ lF\ cos(\phi + \theta_2))$$
$$+ \dot{\phi}\ (-(M + mF + 0.5\ mT)\ lT\ sin(\phi + \theta_2 + \theta_4)\ (\dot{\phi} + \dot{\theta_2} + \dot{\theta_4}) - (M + 0.5\ mF)\ lF\ sin(\phi + \theta_2)\ (\dot{\phi} + \dot{\theta_2}))$$

$$D_{left} = \begin{pmatrix} D11 & D12 & D13 & D14 \\ D21 & D22 & D23 & D24 \\ D31 & D32 & D33 & D34 \\ D41 & D42 & D43 & D44 \end{pmatrix} \quad C_{left} = \begin{pmatrix} C11 & C12 & C13 & C14 \\ C21 & C22 & C23 & C24 \\ C31 & C32 & C33 & C34 \\ C41 & C42 & C43 & C44 \end{pmatrix} \quad G_{left} = \begin{pmatrix} G1 & G2 & G3 & G4 \end{pmatrix}^{\mathrm{T}}$$

NABIROS EQUATIONS OF MOTION :



$$\ddot{p} = D_{right}^{-1}\ (\ \tau_{right} - C_{right}\ p - G_{right}\ ) \qquad \ddot{q} = D_{left}^{-1}\ (\ \tau_{left} - C_{left}\ p - G_{left}\ )$$

$$
\begin{aligned}
\ddot{y}EL = \ddot{y}L &- (lT\ sin(\phi+\theta_2+\theta_4) + lF\ sin(\phi+\theta_2))\ (\ddot{\phi} + d\dot{\theta_2}) - lT\ sin(\phi+\theta_2+\theta_4)\ d\dot{\theta_4} \\
&- lT\ cos(\phi+\theta_2+\theta_4)\ (\dot{\phi}+\dot{\theta}_2+\dot{\theta}_4)^2 - lT\ cos(\phi+\theta_2)\ (\dot{\phi}+\dot{\theta}_2)^2 \\
\ddot{z}EL = \ddot{z}L &+ (lT\ cos(\phi+\theta_2+\theta_4) + lF\ cos(\phi+\theta_2))\ (\ddot{\phi} + d\dot{\theta_2}) + lT\ cos(\phi+\theta_2+\theta_4)\ d\dot{\theta_4} \\
&- lT\ sin(\phi+\theta_2+\theta_4)\ (\dot{\phi}+\dot{\theta}_2+\dot{\theta}_4)^2 - lT\ sin(\phi+\theta_2)\ (\dot{\phi}+\dot{\theta}_2)^2 \\
\ddot{y}ER = \ddot{y}R &+ (lT\ sin(\phi+\theta_1+\theta_3) + lF\ sin(\phi+\theta_1))\ (\ddot{\phi} + d\dot{\theta_1}) + lT\ sin(\phi+\theta_1+\theta_3)\ d\dot{\theta_3} \\
&+ lT\ cos(\phi+\theta_1+\theta_3)\ (\dot{\phi}+\dot{\theta}_1+\dot{\theta}_4)^2 + lT\ cos(\phi+\theta_1)\ (\dot{\phi}+\dot{\theta}_1)^2 \\
\ddot{z}ER = \ddot{z}R &- (lT\ cos(\phi+\theta_1+\theta_3) + lF\ cos(\phi+\theta_1))\ (\ddot{\phi} + d\dot{\theta_3}) - lT\ cos(\phi+\theta_1+\theta_3)\ d\dot{\theta_3} \\
&+ lT\ sin(\phi+\theta_1+\theta_3)\ (\dot{\phi}+\dot{\theta}_1+\dot{\theta}_3)^2 + lT\ sin(\phi+\theta_1)\ (\dot{\phi}+\dot{\theta}_1)^2
\end{aligned}
$$

The end effectors exert forces on the COM such that when resolved into centripetal forces and tangential forces, their net effect would result in net linear and angular accelerations of COM's.

$$
M\ \ddot{r}_{COM} = 0.5\ M\ (\ \ddot{y}EL\ cos\ \phi + \ddot{z}EL\ sin\ \phi\ ) + 0.5\ M\ (\ \ddot{y}ERcos\phi + \ddot{z}ER\ sin\phi\ ) \quad \text{along centripetal axis}
$$

$$
Ix\ \ddot{\phi} = M\ d\ (\ \ddot{y}EL\ sin\phi - \ddot{z}EL\ cos\phi\ ) + M\ d\ (\ \ddot{y}ERsin\ \phi - \ddot{z}ER\ cos\ \phi\ ) \quad \text{along tangential axis}
$$

Now since both half masses are actually halves of a rigid body their respective accelerations are dictated by these linear and angular acceleration of COM. Thus the actual accelerations at which the end effectors move are obtained by double differentiating the constraint equation $\mathcal{F}$ discussed earlier. The difference in the linear accelerations of end effectors that they tried to exert on COM and the actual accelerations that they have due to COM is proportional to the reaction forces influenced by the respective end effectors. The effect of the reaction forces experienced at the end effector can be exerted at the joints as reaction torque with the help of jacobian computed at that instant as follows.

$$
\mathcal{J}_L = \begin{pmatrix} -(lT\ sin(\phi+\theta_2+\theta_4) + lF\ sin(\phi+\theta_2)) & -lT\ sin(\phi+\theta_2+\theta_4) \\ (lT\ cos(\phi+\theta_2+\theta_4) + lF\ cos(\phi+\theta_2)) & lT\ cos(\phi+\theta_2+\theta_4) \end{pmatrix}
$$

$$
\begin{aligned}
\ddot{\theta}_2 s = \ddot{y}EL - \ddot{y}L &+ (lT\ sin(\phi+\theta_2+\theta_4) + lF\ sin(\phi+\theta_2))\ \ddot{\phi} + lT\ cos(\phi+\theta_2+\theta_4)\ (\dot{\phi}+\dot{\theta}_2+\dot{\theta}_4)^2 \\
&+ lT\ cos(\phi+\theta_2)\ (\dot{\phi}+\dot{\theta}_2)^2 \\
\ddot{\theta}_4 s = \ddot{z}EL - \ddot{z}L &- (lT\ cos(\phi+\theta_2+\theta_4) + lF\ cos(\phi+\theta_2))\ \ddot{\phi} \\
&+ lT\ sin(\phi+\theta_2+\theta_4)\ (\dot{\phi}+\dot{\theta}_2+\dot{\theta}_4)^2 + lT\ sin(\phi+\theta_2)\ (\dot{\phi}+\dot{\theta}_2)^2
\end{aligned}
$$

$$
{\tau_L}^{reac} = \mathcal{J}_L^{-1}\ (\ \ddot{\theta}_2 s \quad \ddot{\theta}_4 s\ )^{\mathrm{T}}
$$

$$
\mathcal{J}_R = \begin{pmatrix} (lT\ sin(\phi+\theta_1+\theta_3) + lF\ sin(\phi+\theta_1)) & lT\ sin(\phi+\theta_1+\theta_3) \\ -(lT\ cos(\phi+\theta_1+\theta_3) + lF\ cos(\phi+\theta_1)) & -lT\ cos(\phi+\theta_1+\theta_3) \end{pmatrix}
$$

$$
\begin{aligned}
\ddot{\theta}_1 s = \ddot{y}ER - \ddot{y}R &- (lT\ sin(\phi+\theta_1+\theta_3) + lF\ sin(\phi+\theta_1))\ \ddot{\phi} - lT\ cos(\phi+\theta_1+\theta_3)\ (\dot{\phi}+\dot{\theta}_1+\dot{\theta}_3)^2 \\
&- lT\ cos(\phi+\theta_1)\ (\dot{\phi}+\dot{\theta}_1)^2 \\
\ddot{\theta}_3 s = \ddot{z}ER - \ddot{z}R &+ (lT\ cos(\phi+\theta_1+\theta_3) + lF\ cos(\phi+\theta_1))\ \ddot{\phi} \\
&- lT\ sin(\phi+\theta_1+\theta_3)\ (\dot{\phi}+\dot{\theta}_1+\dot{\theta}_3)^2 - lT\ sin(\phi+\theta_1)\ (\dot{\phi}+\dot{\theta}_1)^2
\end{aligned}
$$

$$
{\tau_R}^{reac} = \mathcal{J}_R^{-1}\ (\ \ \ddot{\theta}_1 s \quad \ddot{\theta}_3 s\ \ )^{\mathrm{T}}
$$

The control architecture under this approach is as given in figure **??**.

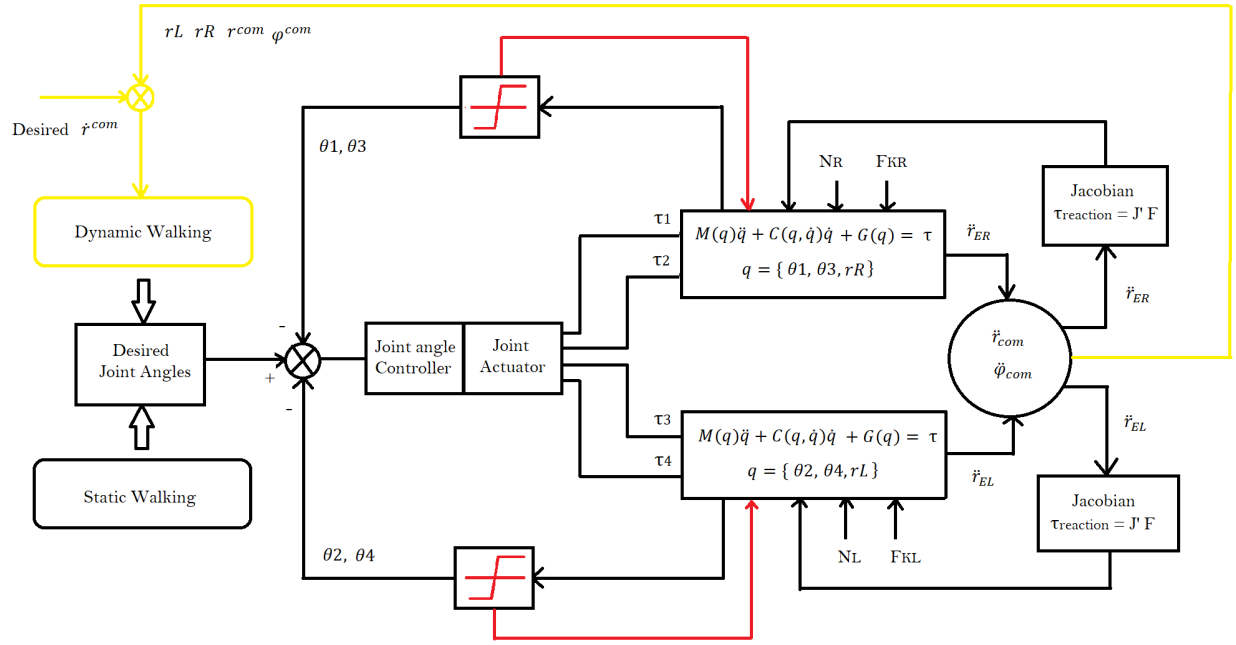
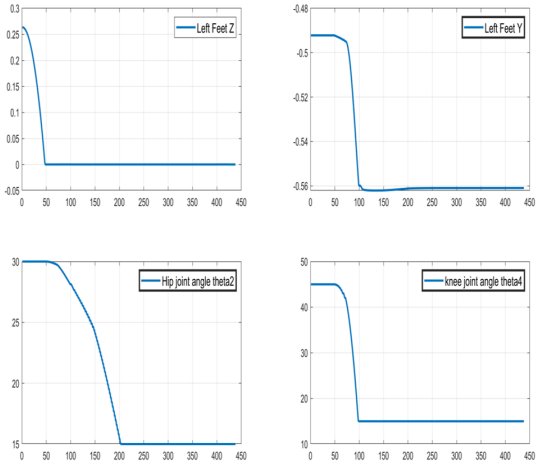


Figure 14: Control archicture for branched langrangian mechanics of robot

Based on the above architecture a dynamics simulator was setup in MATLAB environment that can exerted gravity, normal reaction, frictional force and reaction torque due to joint angle limits all restricted to sagittal plane of robot. The experimental results are as follows :

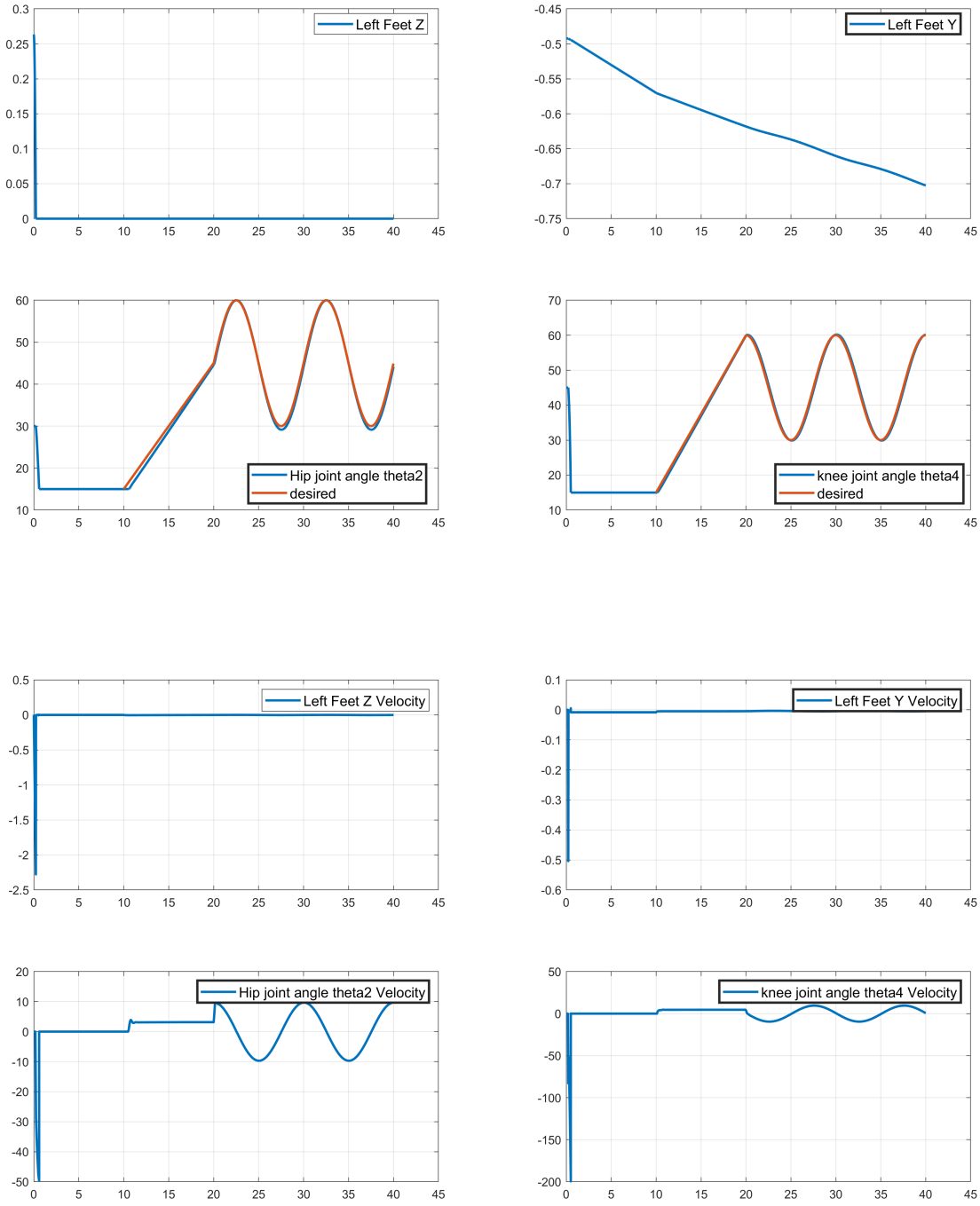
Experiment 1 : To verify the effect of joint angle limits on the robot.
The joint angles were limited in the range of [$\pi/12$, $5\pi/12$]. The study was carried out by actuating left leg knee and hip joints.
Click here for animated visualisation of knee actuation under zero gravity
Click here for animated visualisation of hip actuation under zero gravity

Experiment 2 : To verify the effect of gravity, normal reaction and friction forces on robot.
The robot is made to free fall. To study the effect of nrmal reaction and friction, these forces were only applied on the left leg on impact with the zero line. However to incorporate the reaction of the right leg if there forces were also acting on them, the orientation of the base link was locked by setting $\ddot{\phi} = 0$ and $\dot{\phi} = 0$.
Click here for animated visualisation of effect of normal reaction and friction on robot dynamics under gravity



Experiment 3 :To verify desired joint angle tracking by left leg. To incorporate the reaction of the right leg if there forces were also acting on them, the orientation of the base link was locked by setting $\ddot{\phi} = 0$ and

$\dot{\phi} = 0.$



Click here for animated joint angle tracking of left leg under gravity

6 Conclusion

So, a unique bipedal robot design by RoMeLa was created. And the control startegy for it was developed based on the ZMP approach. The preview controller was designed in MATLAB,and center of mass trajectories were generated. This center of mass trajectory was used to find the joint angles using inverse kinematics code that was written in MATLAB. Then to check whether it is working as intended, a kinematic simulation was done in MATLAB, and a video was generated. Later a model-free technique was used to make the robot learn to walk on a wedge. And a video of the same was generated using OpenAI interface and MuJoCo.

Though the idea was to run both the ZMP based control and RL on the simulator built by us, we had to work parallelly to meet the project deadlines. Hence, we chose different simulators for the experiments. Once the simulator is ready we would like to apply the same techniques on it and see the results.

Click here for Github Repository containing MATLAB and python codes, URDF and Mujoco files

7 Future scope

Other new methods such as CPG(central pattern generator) can be used to develop the control. Also, the limitation of the current design is that it can move only in a straight line. So, strategies to move NABIRos in a plane can be worked upon. RoMeLa Lab which had developed this preliminary design have actually developed NABIRos which can move in a plane by having two degree of freedom at hip joint. They were also, able to make it move in uneven terrain. So, further work can be done on NABIRos to make it more practical i.e walk in plane and uneven terrain without compromising the benefits inherent to the design.

Another work can be to make the Reinforcement Learning more sample efficient. As, discussed in the previous sections, the agent took quite high number of simulation timesteps to finally learn to walk. A good idea would be to incorporate the learning from control technique i.e, ZMP to initialize the RL agent. We hope this kind of an interaction between the control and learning can be used to increase the sample efficiency.

Another possible future work would be to use Meta RL to learn the agent to walk on varying slopes. Starting to learn to walk on wedges/slopes of varied angles from scratch would be tedious and not practical. Hence, learning a generic model to walk on slopes could be learnt using Meta RL and quickly adapt it to the new slopes as and when encountered.

8 References

1) S. Ghassemi, J. Yu, J. Hooks and D. Hong, "Feasibility study of a novel biped NABiRoS: Non anthropomorphic bipedal robotic system," 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), 2016, pp. 145-145, doi: 10.1109/HUMANOIDS.2016.7803269.

2) S. Ghassemi, J. Yu, J. Hooks and D. Hong, "Feasibility study of a novel biped NABiRoS: Non anthropomorphic bipedal robotic system," 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), 2016, pp. 145-145, doi: 10.1109/HUMANOIDS.2016.7803269.

3) Kajita, Shuuji, et al. "Biped walking pattern generation by using preview control of zero-moment point." 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422). Vol. 2. IEEE, 2003.

4) Mania, Horia, Aurelia Guy, and Benjamin Recht. "Simple random search provides a competitive approach to reinforcement learning." arXiv preprint arXiv:1803.07055 (2018).

5) Rajeswaran, Aravind, Kendall Lowrey, Emanuel Todorov, and Sham Kakade. "Towards generalization and simplicity in continuous control." arXiv preprint arXiv:1703.02660 (2017).

6) Nesterov, Yurii, and Vladimir Spokoiny. "Random gradient-free minimization of convex functions." Foundations of Computational Mathematics 17, no. 2 (2017): 527-566.