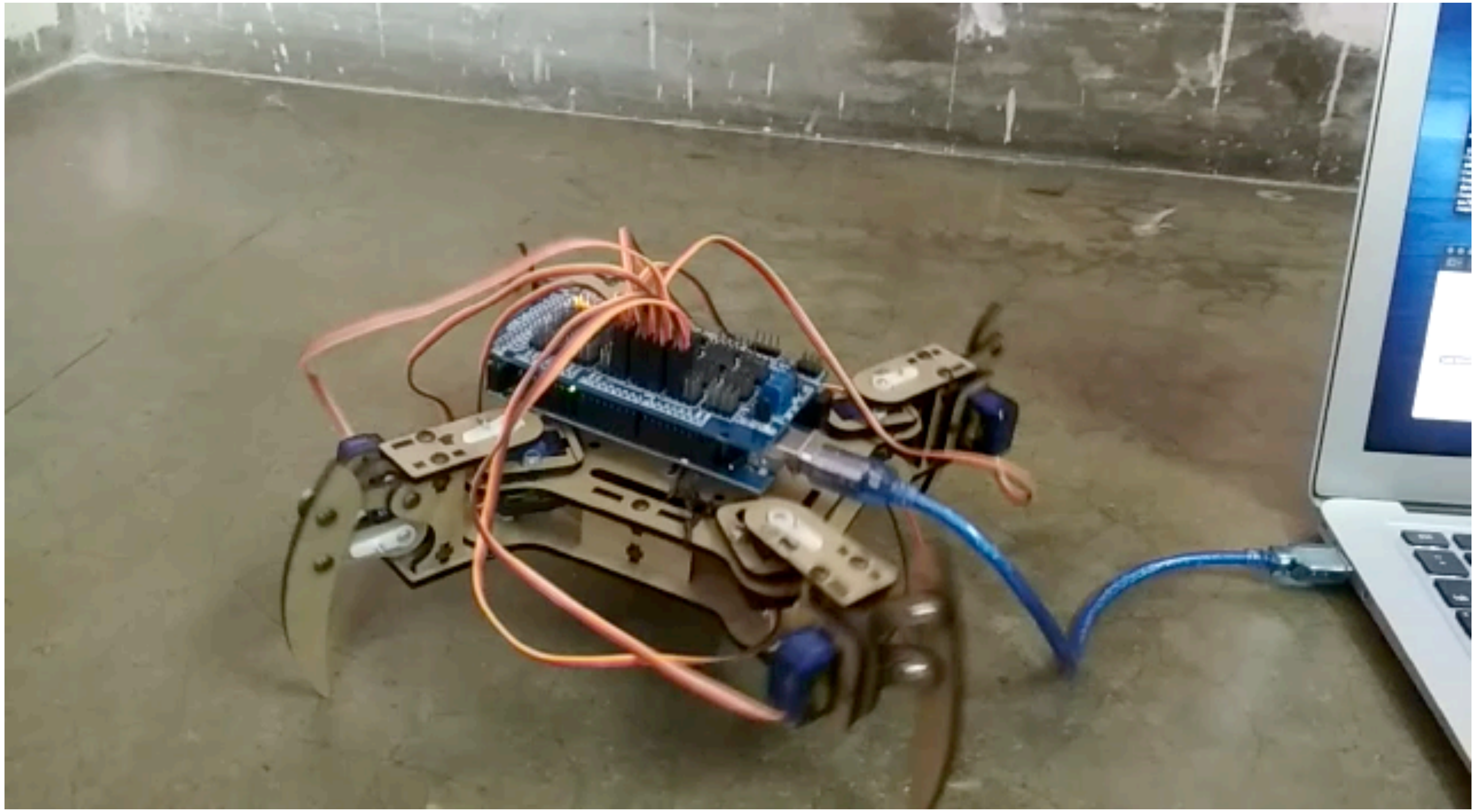
Quadruped Gait Learning

Design of Control System for Quadruped using Central Pattern Generators

Shreyas Shandilya

Robot walking Video

Observe the tendency of the robot to topple towards the swinging leg



Gait Pattern

Pattern in use and problems with it

- This Gait Pattern obtained from the formulation in the figure is for a creep walk of the quadruped
- The robot walks with this Gait Pattern
- But the stability of the quadruped is not maintained
- The quadruped tends to topple towards the leg in swing phase

Hip Activations:

$$\theta_h(t) = \begin{cases} \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \pi), & \text{if } 0 \le t \le \frac{\beta T}{2} \\ \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{(1-\beta)T} + \frac{(3-4\beta)\pi}{2(1-\beta)}), & \text{if } \frac{\beta T}{2} \le t \le \frac{T(2-\beta)}{2} \\ \theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \frac{(\beta-1)\pi}{\beta}), & \text{if } \frac{T(2-\beta)}{2} \le t \le T \end{cases}$$

For $i \in \{0,1\}$

And

$$\theta_h(t) = \begin{cases} -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \pi), & \text{if } 0 \le t \le \frac{\beta T}{2} \\ -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{(1-\beta)T} + \frac{(3-4\beta)\pi}{2(1-\beta)}), & \text{if } \frac{\beta T}{2} \le t \le \frac{T(2-\beta)}{2} \\ -\theta_h sin(\frac{(t-\frac{iT}{4})\pi}{\beta T} + \frac{(\beta-1)\pi}{\beta}), & \text{if } \frac{T(2-\beta)}{2} \le t \le T \end{cases}$$

For $i \in \{2,3\}$

Knee Activations:

$$\theta_k(t) = \begin{cases} \theta_k sin(\frac{t\pi}{T(1-\beta)} - \frac{\beta\pi}{2(1-\beta)}), & \text{if } \dot{\theta}_h(t) \ge 0 \\ 0, & \text{otherwise} \end{cases}$$

For $i \in \{0,1\}$

And

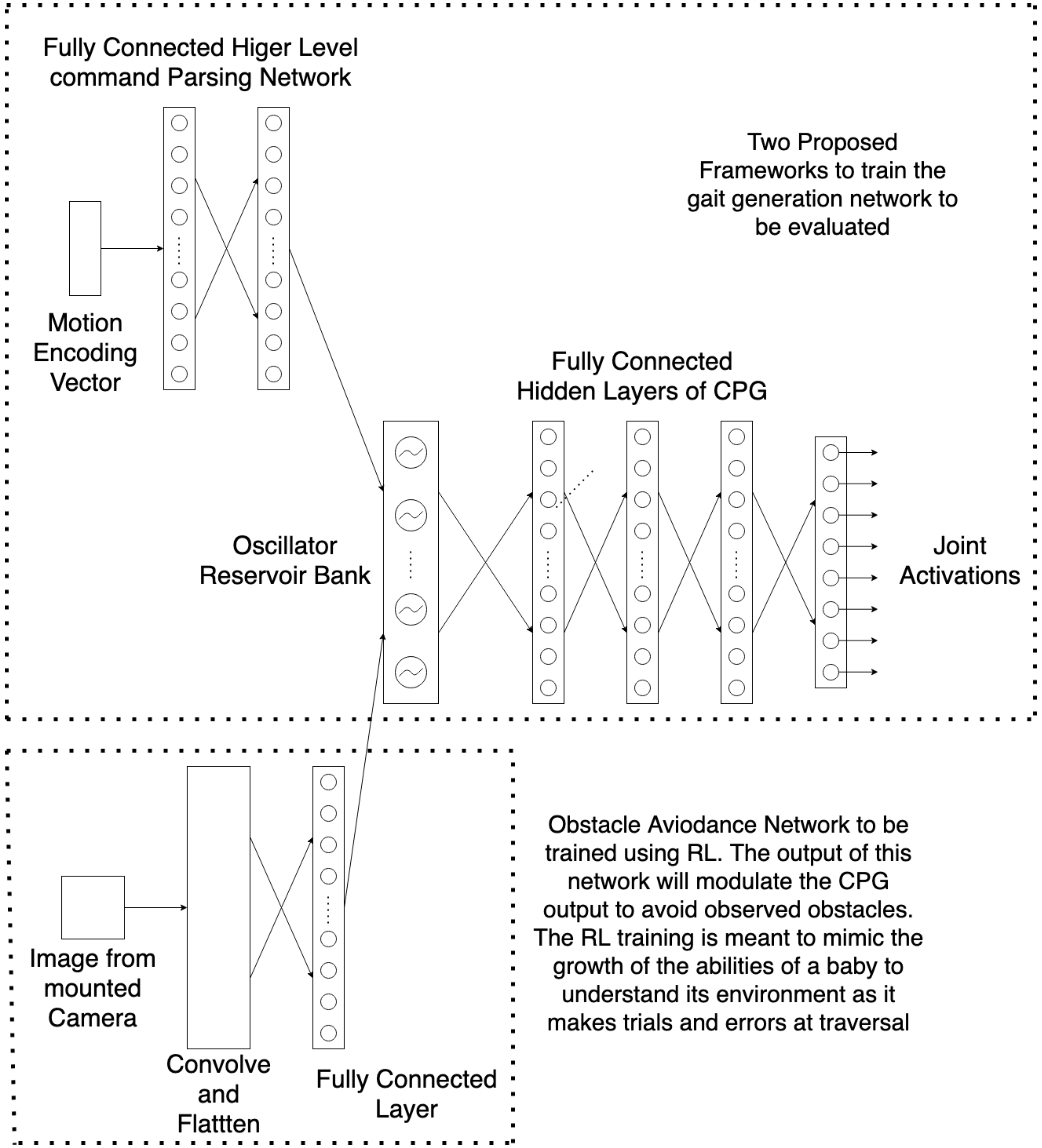
$$\theta_k(t) = \begin{cases} \theta_k sin(\frac{t\pi}{T(1-\beta)} - \frac{\beta\pi}{2(1-\beta)}), & \text{if } \dot{\theta}_h(t) \le 0 \\ 0, & \text{otherwise} \end{cases}$$

For $i \in \{2,3\}$

Architecture

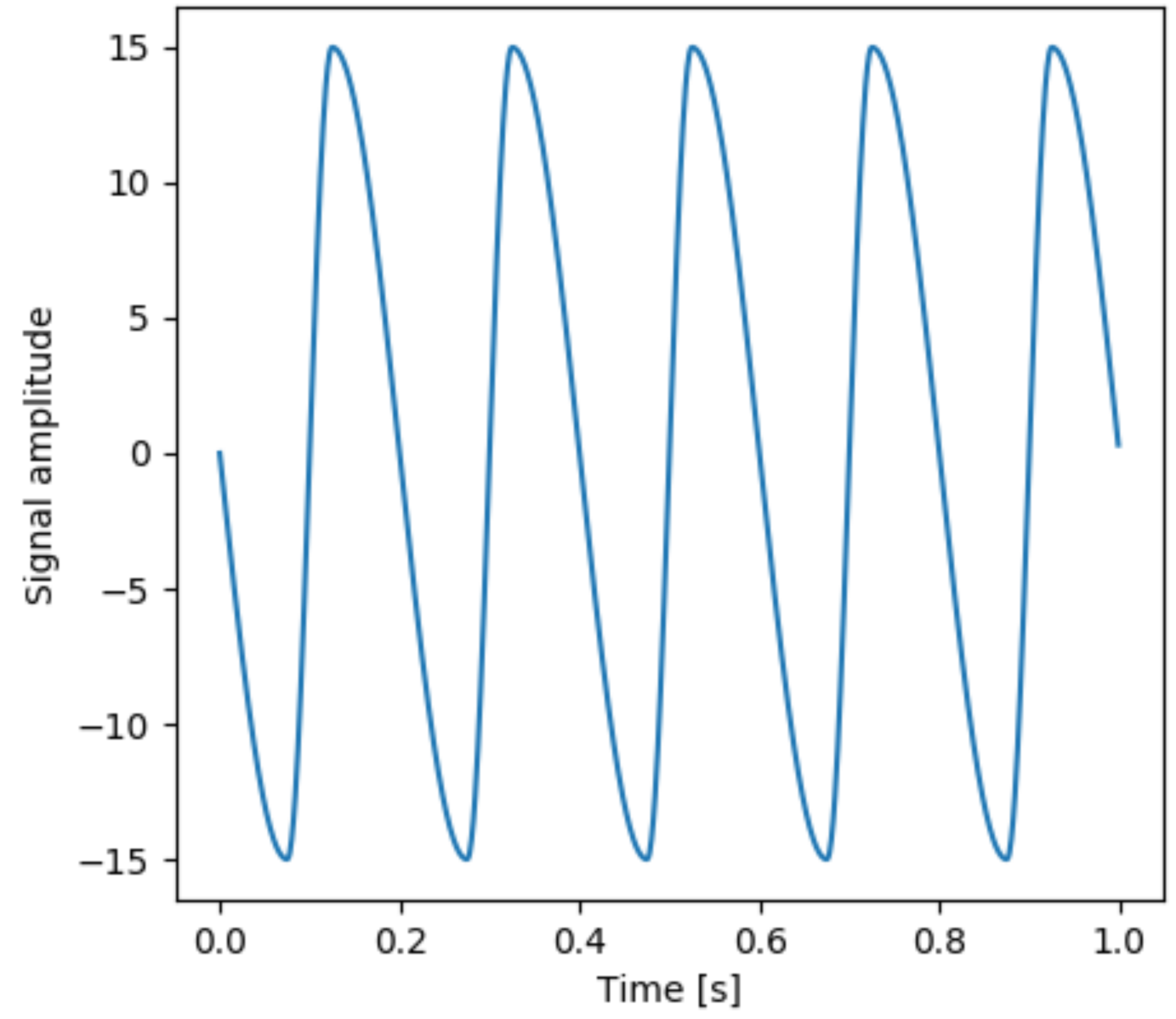
Proposed solution to ensure gait stability

- Two methods Proposed to train the gait generation network
  - Train the gait generation network end to end using back props and then use RL training for improvement of gait
  - Optimise Gait Patterns using fitness function, then train the network end to end using back prop

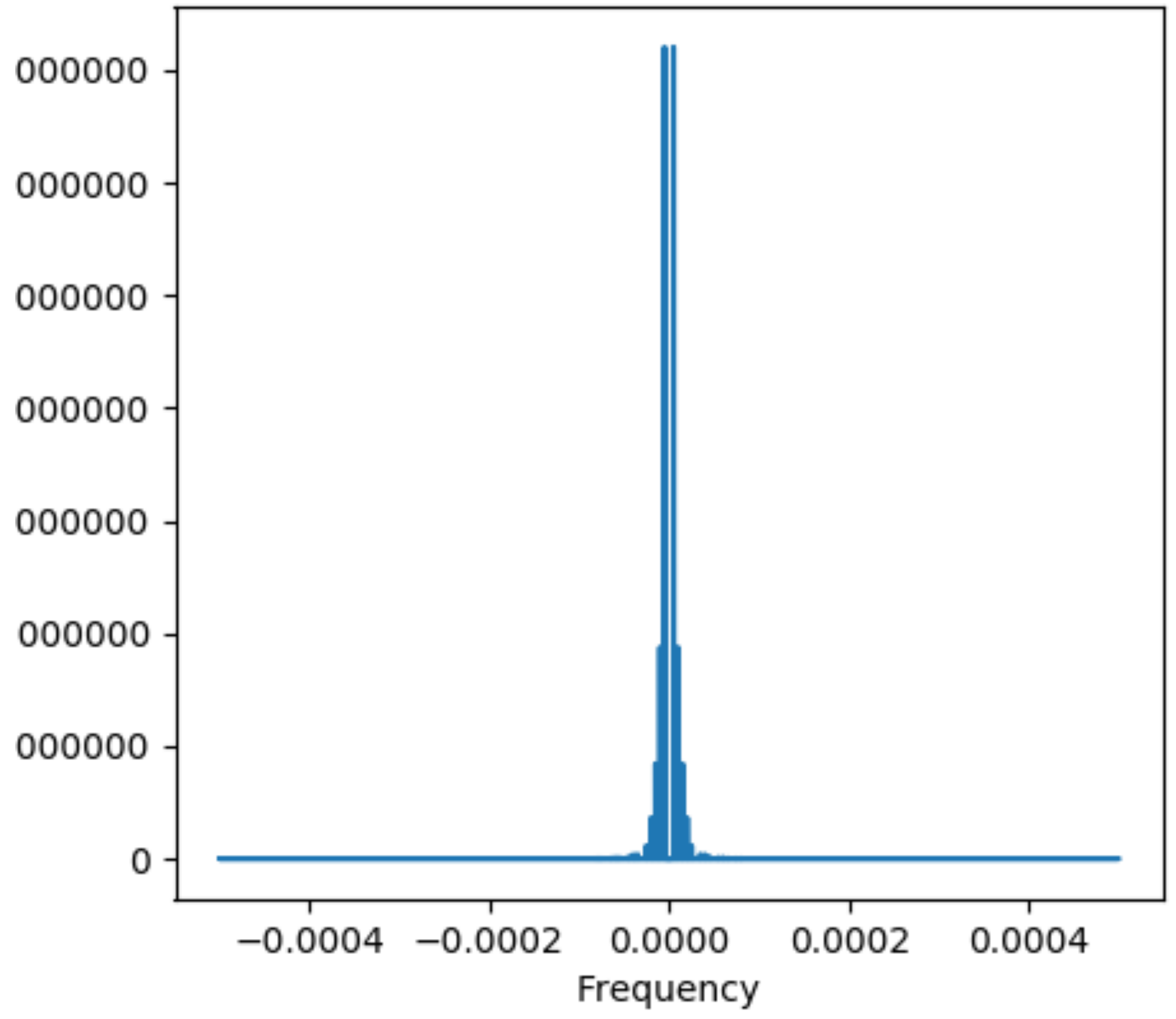


Supervised Learning of Gait Pattern

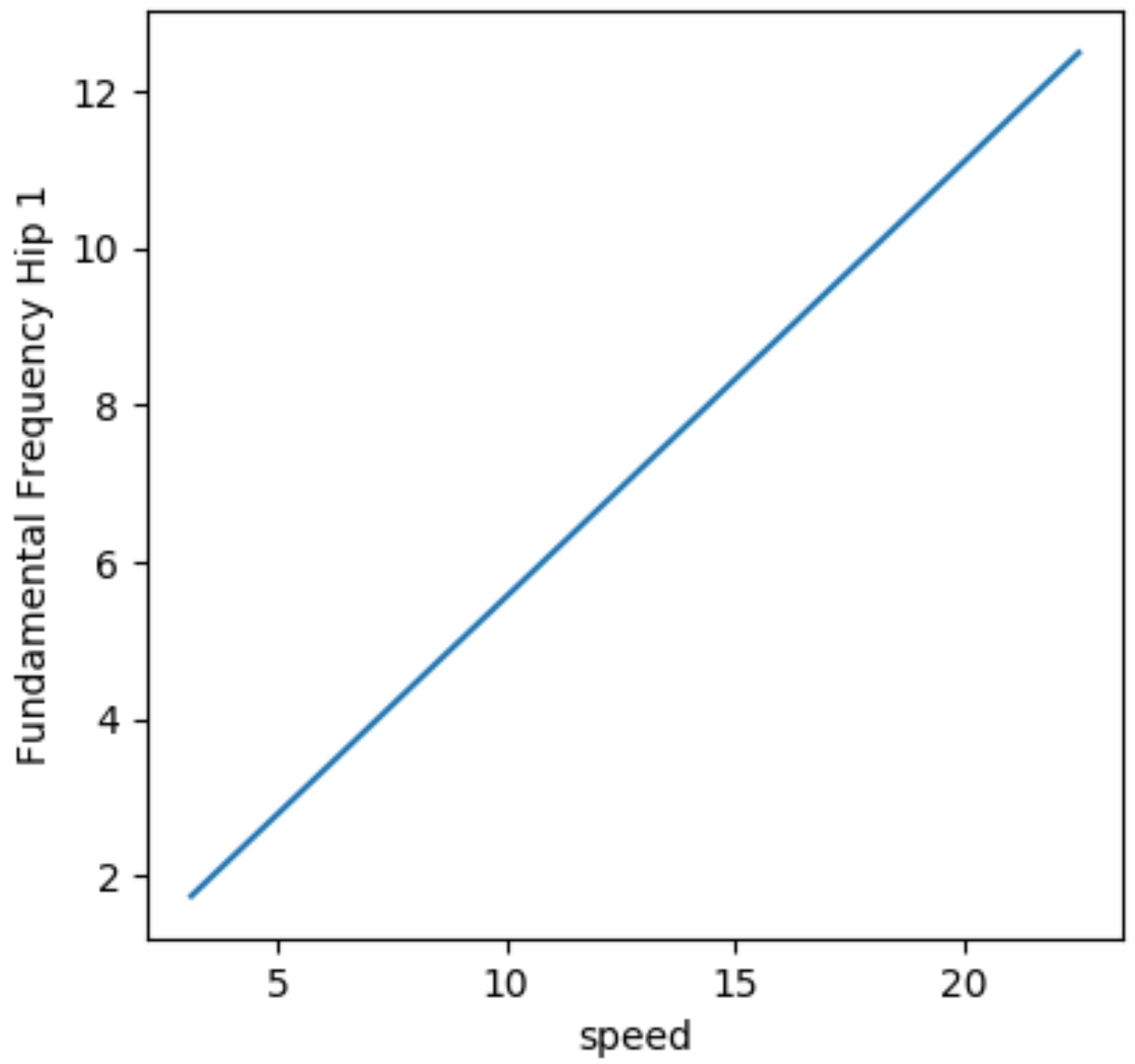
- A proportional relationship between fundamental frequency of gait signal and speed of quadruped was established
- Using the fundamental frequency as input, it was demonstrated that the proposed network is capable of generating gait patterns

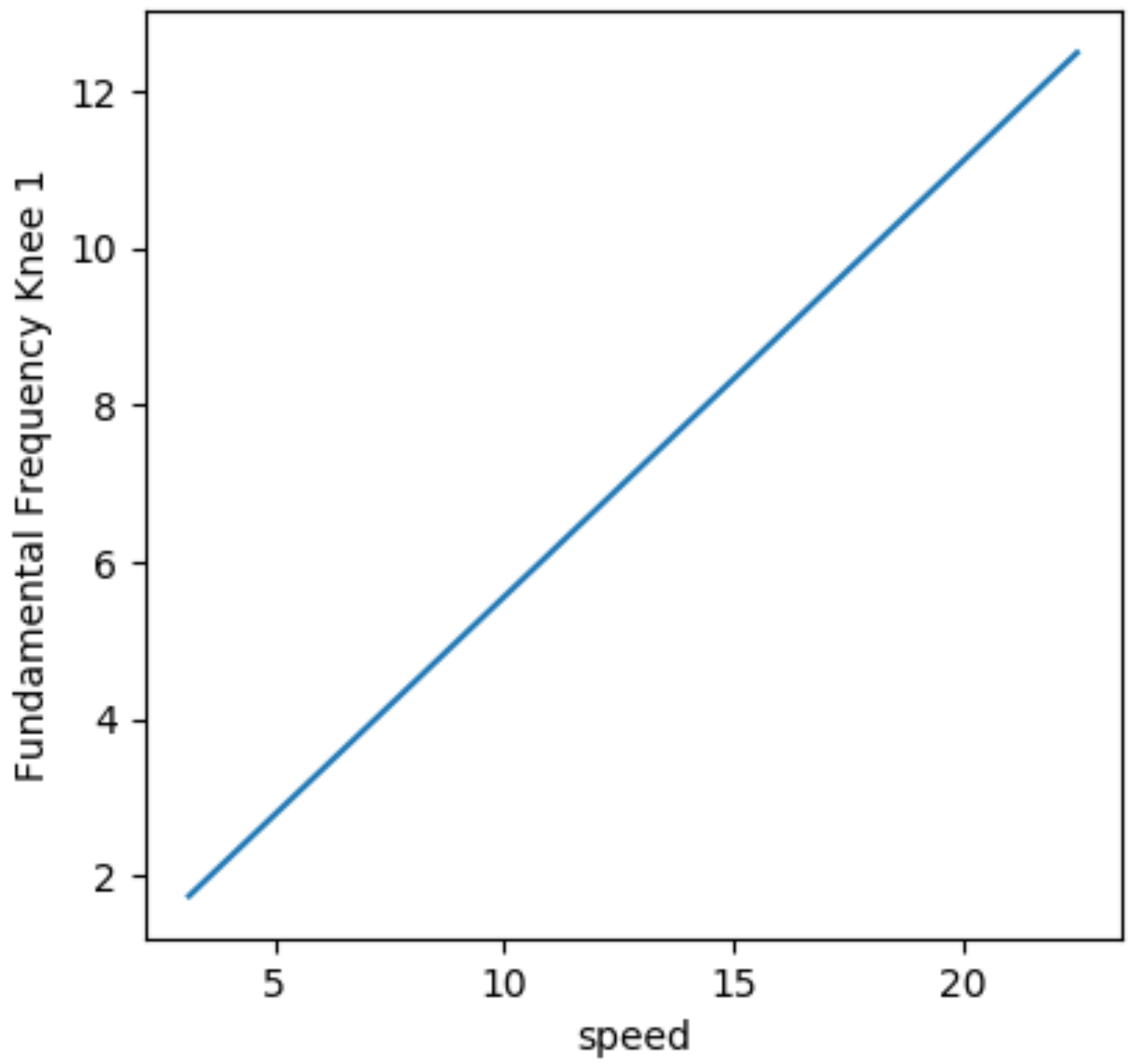


Hip Joint Signal

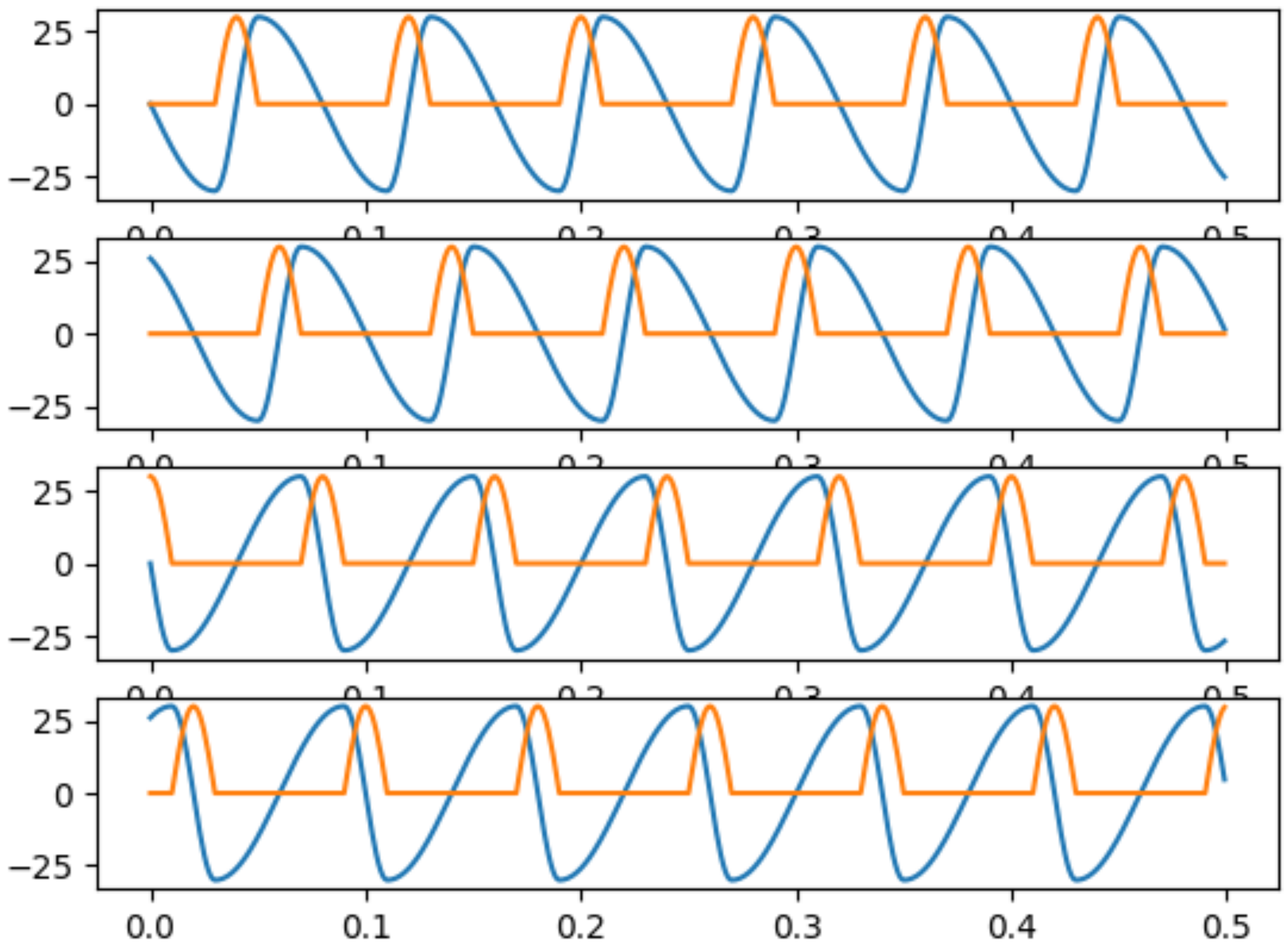


Frequency spectrum of hip joint signal

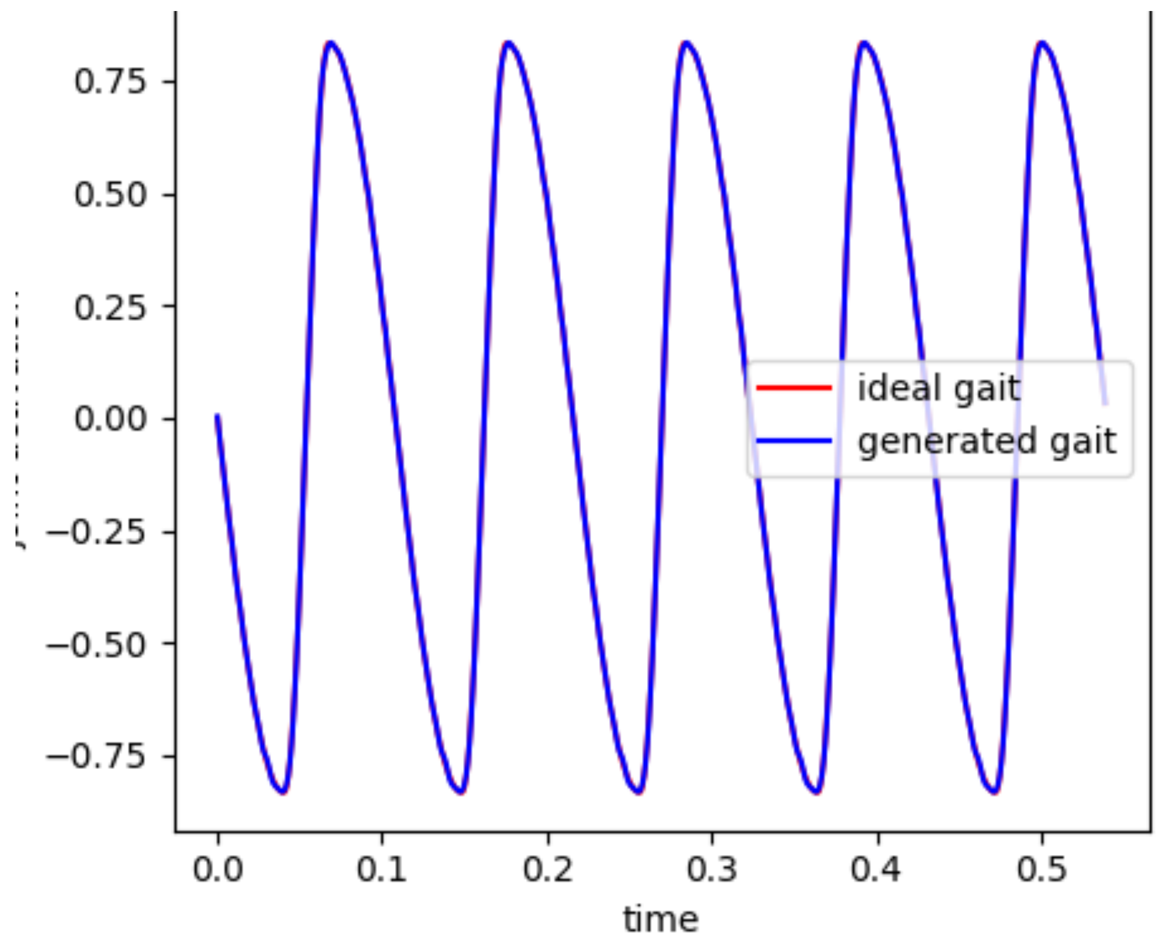


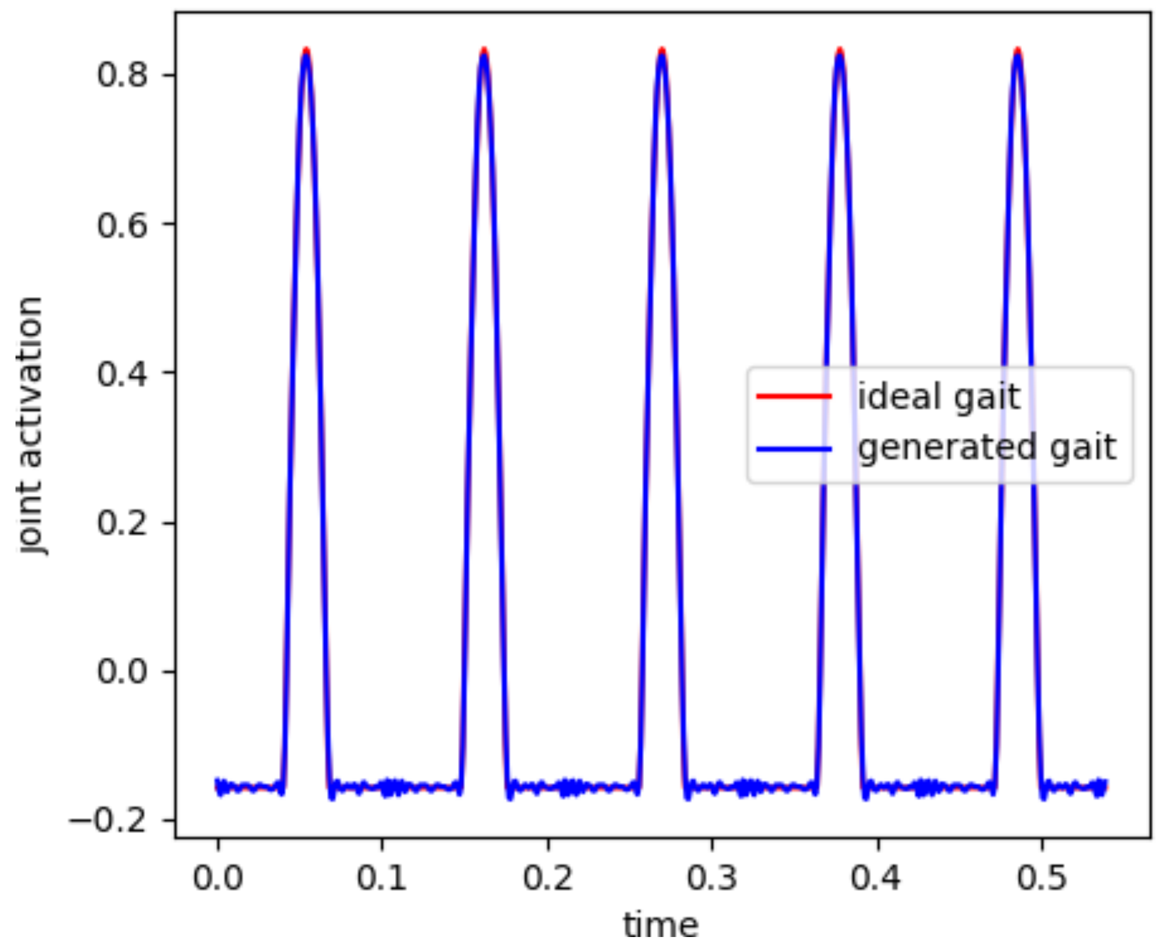


Supervised Learning of Gait Pattern



Gait Pattern for Creep Gait





Model was able to reconstruct gait signal with 100% accuracy for unto 10 different patterns at a time

Fitness Function

Evaluation Criteria to measure stability, speed and energy consumption

- Both the aforementioned training frameworks require a Fitness Function [1]
- Fitness Function must measure the following
  - Stability Criteria, S
  - Energy Efficiency Criteria, E
- $F = k_s S_i + k_e E_i$ where
  - F is the Fitness Value
  - k_s and k_e are parameters to account for relative importance of each criteria

Stability Criteria

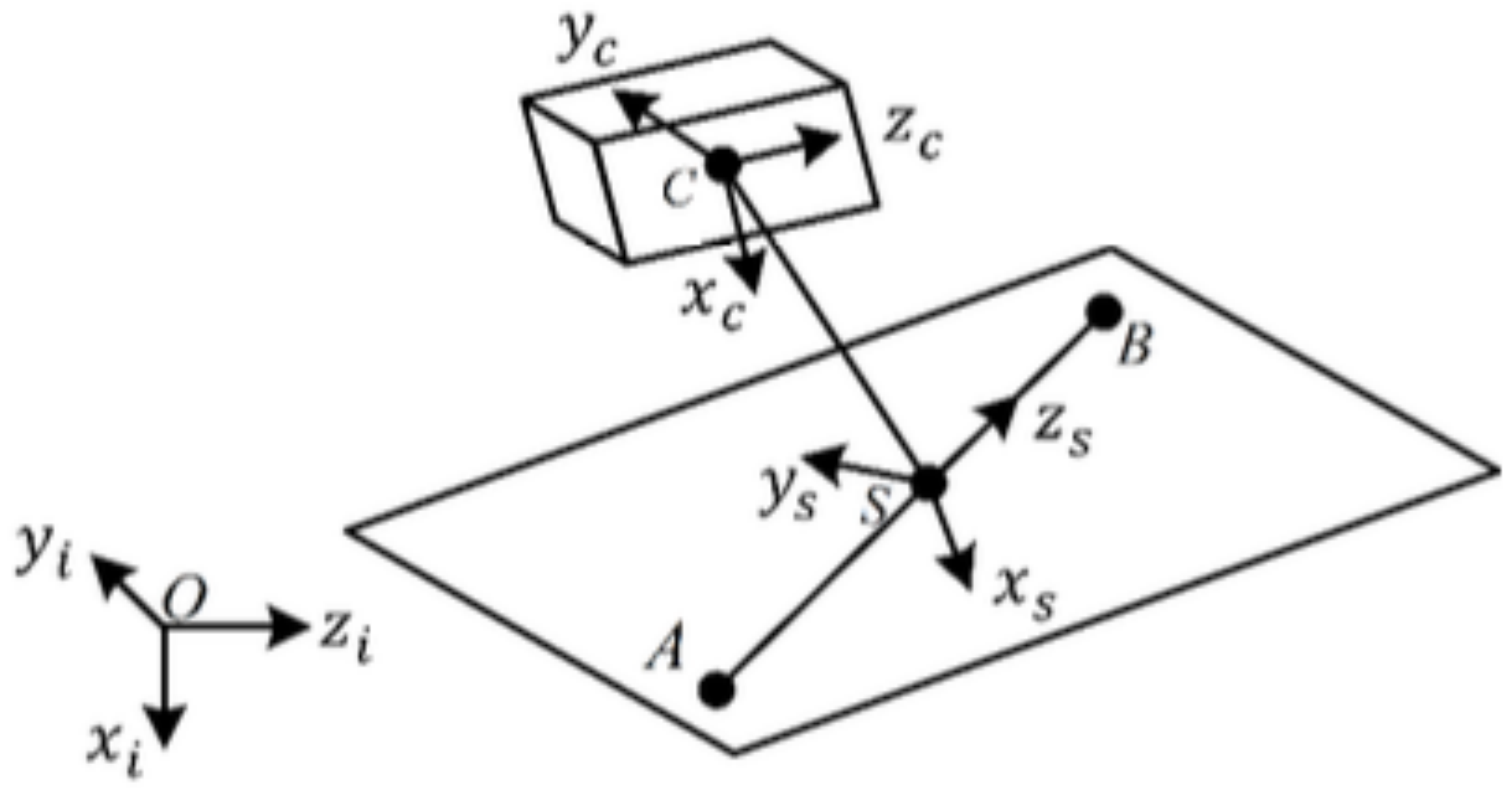
Evaluation of Dynamic and Static Stability of the Quadruped

- A Quantitative Measure of quadruped's dynamic stability
- The Criteria must not only take into account current stability but must also **anticipate the stability at the next time** instance, given the state of motion of the quadruped, like an animal does
- Yan Jia, Xiao Luo and others [2] in their paper propose a modified ZMP based stability evaluation criteria
- The main idea is that at a certain state, the motion of the robot is considered to be stable if the **torque caused by the ground-reaction force can prevent the robot from tumbling** around any support boundary

Stability Criteria

Coordinate Systems used

- There are two coordinate systems of interest-
  - The Inertial Coordinate system
  - The Support Coordinate system
- The Support Coordinate has z-axis along the support line AB and x axis perpendicular to the current support plane



Stability Criteria

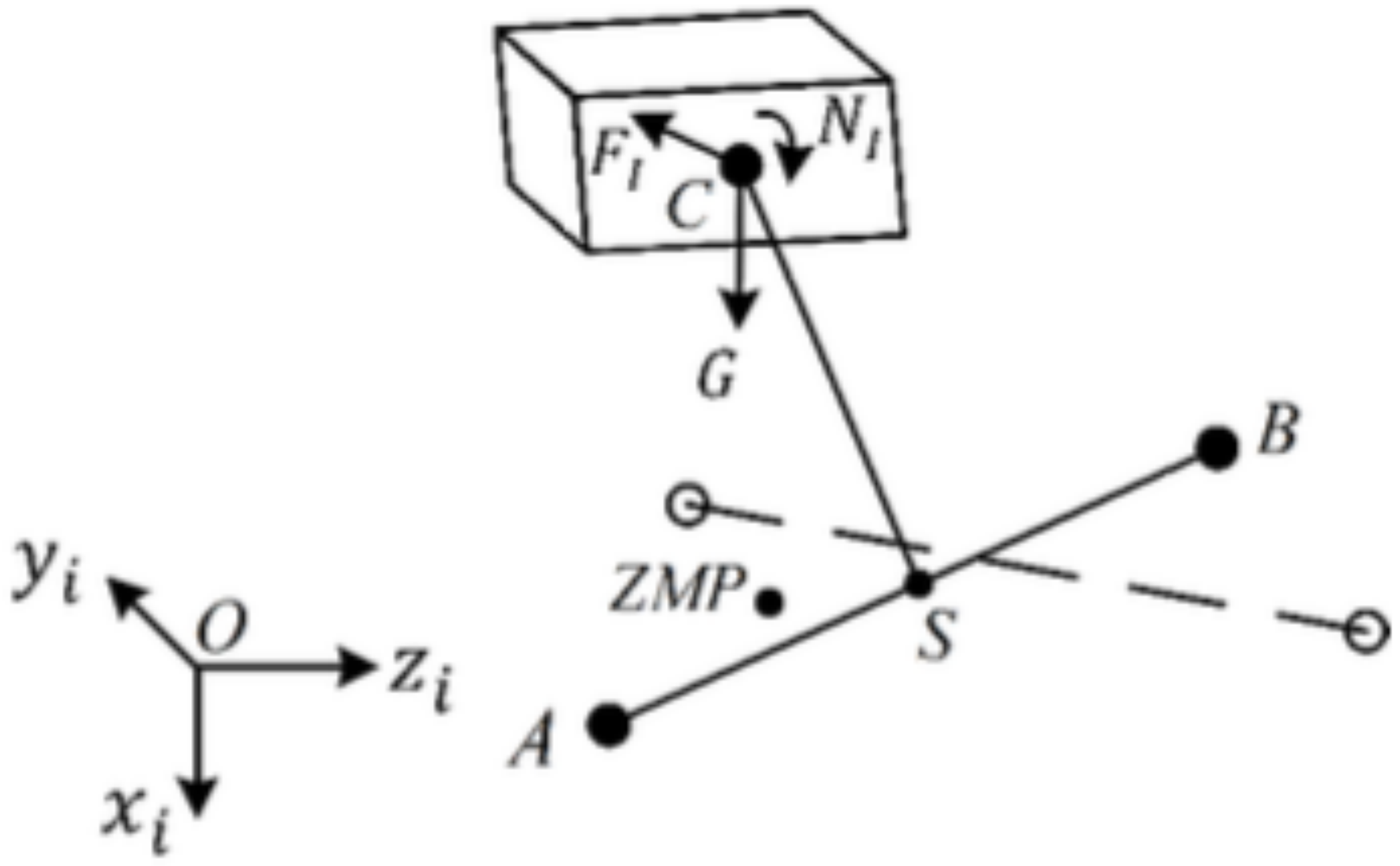
Equations for Zero Moment Point

$$-\left(z_C^i - z_{ZMP}^i\right)F_{Iy}^i + \left(y_C^i - y_{ZMP}^i\right)F_{Iz}^i + N_{Ix}^i = 0;$$

$$\left(z_C^i - z_{ZMP}^i\right)\left(F_{Ix}^i + G_x^i\right) - \left(x_C^i - x_{ZMP}^i\right)F_{Iz}^i + N_{Iy}^i = 0;$$

$$-\left(y_C^i - y_{ZMP}^i\right)\left(F_{Ix}^i + G_x^i\right) + \left(x_C^i - x_{ZMP}^i\right)F_{Iy}^i + N_{Iz}^i = 0.$$

The characteristic equations of ZMP. The inertial force and inertial moment here need to be computed by a dynamic analysis of the quadruped performed later in the report



The force schematic used to calculate the position of ZMP.

Here G is gravity, F is the inertial force on COM C, N is the inertial moment

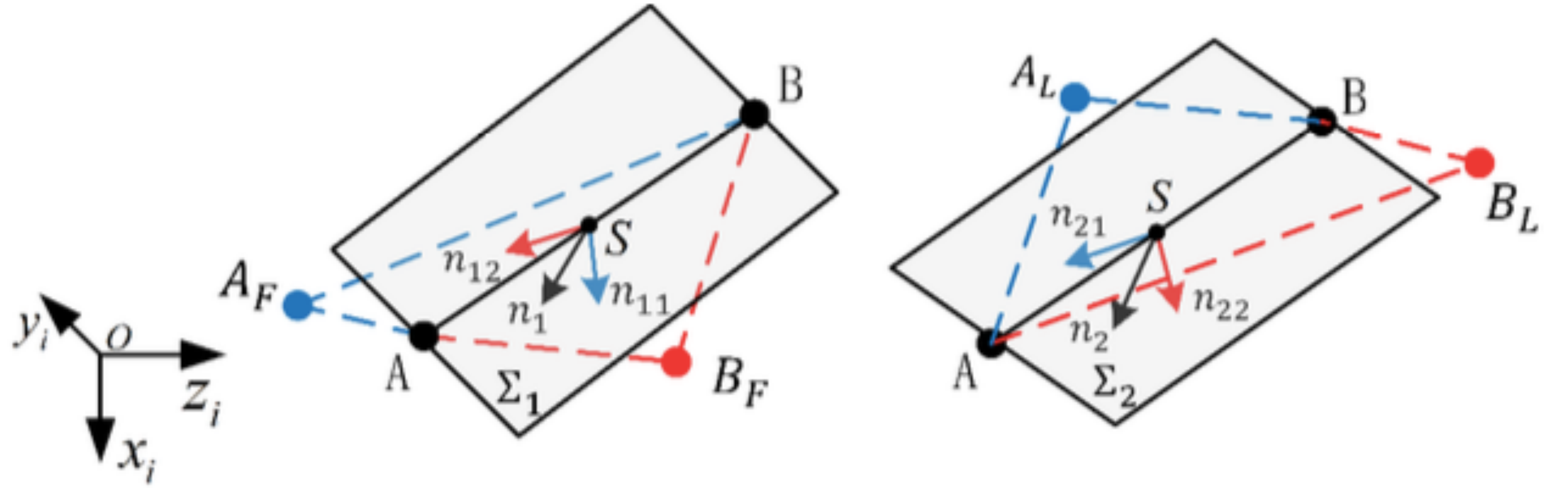
Stability Criteria

Virtual Support Plane and Modified Zero Moment Point

- The paper assumes that a robot is stable if the current or next set of support feet can provide the moment necessary to prevent the robot from tipping in any direction
- A Virtual Support Plane is proposed and it is proposed that if the modified ZMP proposed in the paper lies within this Virtual Support Plane, then the robot motion will be stable
- The Proposed Modified ZMP in the paper also includes a velocity term to account for the motion of the quadruped

Stability Criteria

Calculating the Virtual Support Plane



- S is the origin of the support coordinate system on which the ZMP will lie
- n_1 is the normal vector to the plane AA_FBB_F
- n_2 is the normal vector to the place AA_LBB_L
- x_s^i, y_s^i and z_s^i are the normal vectors to the virtual support plane in the inertial frame of reference
- T_b is the duration between two adjacent steps and *t* is the time gap between the current running time and the time point when the previous support line disappeared

$$x_s^i = \frac{\mu n_1 + (1-\mu) n_2}{\|\mu n_1 + (1-\mu) n_2\|}$$

$$\mu = -\frac{1}{T_b} t + 1 \qquad z_s^i = \frac{\vec{AB}}{\|\vec{AB}\|} \qquad y_s^i = z_s^i \times x_s^i$$

$$n_{11} = \frac{\vec{AB} \times \vec{AA_F}}{\|\vec{AB} \times \vec{AA_F}\|} \qquad n_{21} = \frac{\vec{AB} \times \vec{AA_L}}{\|\vec{AB} \times \vec{AA_L}\|}$$

$$n_{12} = \frac{\vec{AB} \times \vec{BB_F}}{\|\vec{AB} \times \vec{BB_F}\|} \qquad n_{22} = \frac{\vec{AB} \times \vec{BB_L}}{\|\vec{AB} \times \vec{BB_L}\|}$$

Stability Criteria

Calculation of Modified Zero Moment Point

- Zero Moment Point is the point at which the resultant moment on a body is zero
- Modified Zero Moment Point should-
  - Assess the current stability more efficiently and accurately
  - Provide a reference to eliminate undesired velocity during motion planning
- ZMP_o can be used to compute 3 measures of dynamic stability

$$x^s_{ZMP_0} = 0;$$

$$y^s_{ZMP_0} = y^s_{ZMP} + \eta\left(v^s_{yr} - v^s_{yd}\right);$$

$$z^s_{ZMP_0} = z^s_{ZMP} + \eta(v^s_{zr} - v^s_{zd}).$$

Equations to calculate modified ZMP in support coordinate system. v^s and v^d are the actual and expected velocities in support coordinate system

$$\eta = \frac{\frac{1}{2}(L+W)}{\|v_d\|}0.1 = 0.05\frac{(L+W)}{\|v_d\|}$$

L and W are the effective length and width respectively of the quadruped

$${}^i_sR = \left[\begin{array}{ccc} x^i_s & y^i_s & z^i_s \end{array}\right]$$

Rotation Matrix for transformation from support to inertial coordinate system

Stability Criteria

Calculation of the metric S

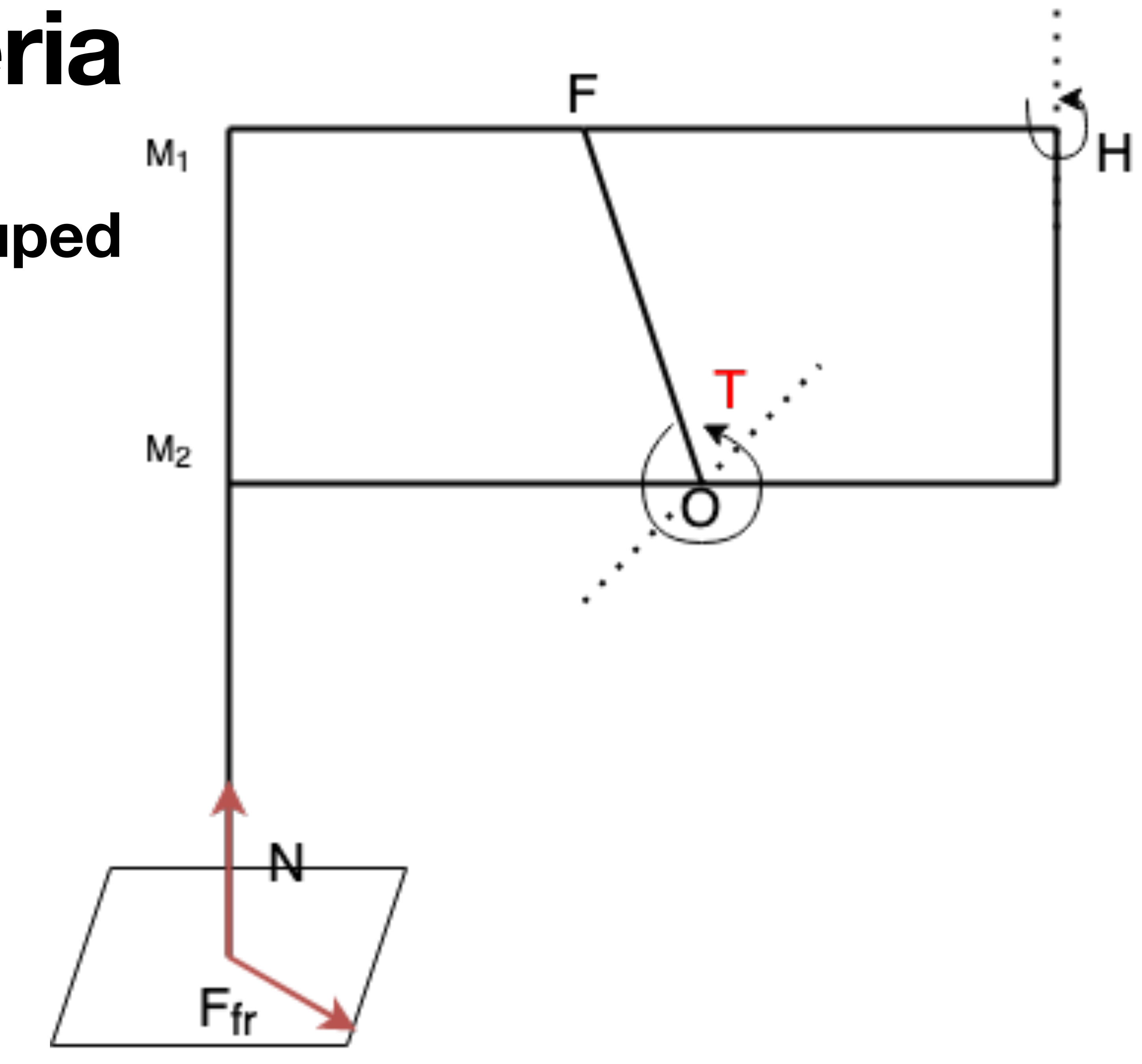
The following three choices for S can be calculated using ZMP_o-

- Distances between ZMP_o and the boundaries of the virtual-support quadrilateral in the support plane
- Angle between the vector pointing from CoM to ZMP_o and the normal vector of the virtual support place
- Distance between ZMP_o and the support line
- The assumption in the paper that the legs have negligible weight compared to the rest of the body does not hold for the quadruped used
- There is a need for evaluation of the repercussions on the concepts introduced

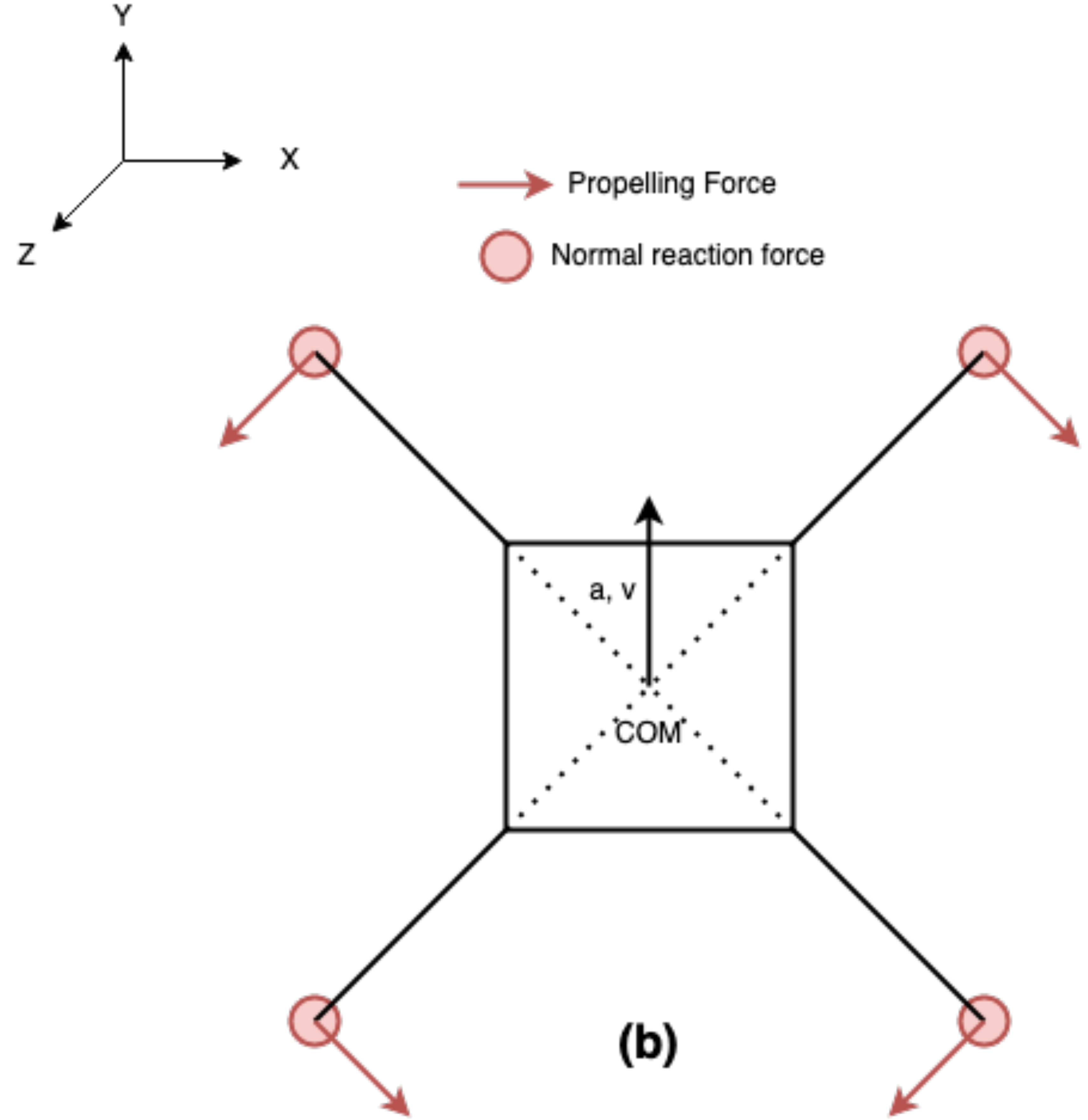
Calculating Stability Criteria

Computing ZMP for the available quadruped

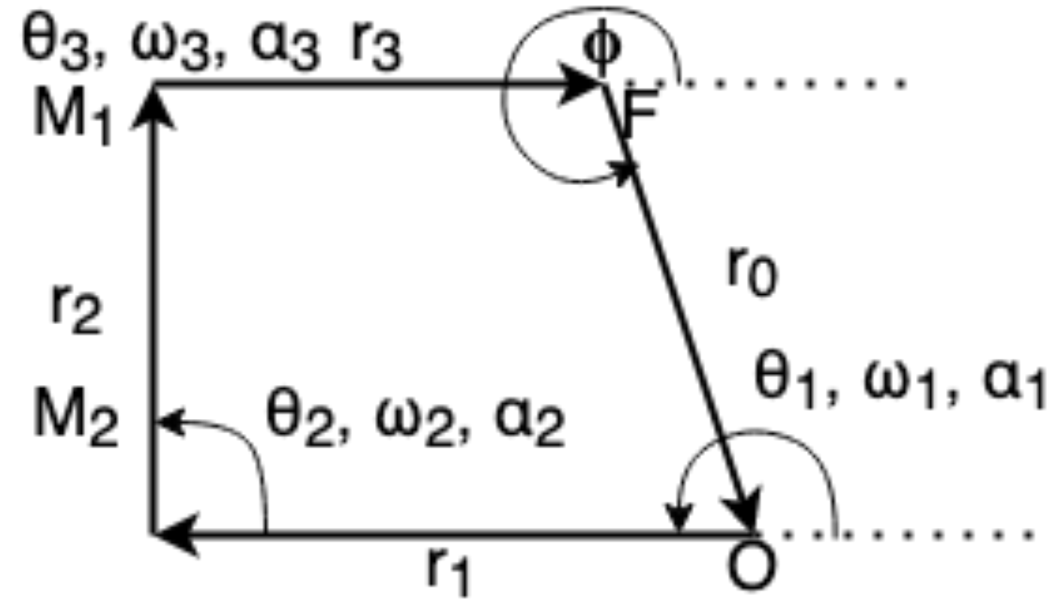
- The following figure depicts the simplified model of the quadruped.
- F_{fr} is the force due to friction acting at the point of contact. This is also the propelling force on the quadruped
- N is the normal reaction force from the ground
- θ, ω and α are the state of the joints in the four bar linkage in (c)



(a)



(b)



(c)

(a) Simplified model of the leg of the quadruped

(b) Simplified model of the quadruped as seen form the top

(c) Simplified four bar linkage to model the knee of the quadruped

Calculating Stability Criteria

Relationship between knee joint activation and four bar linkage model angular positions

Of the four angular positions that in the four bar linkage model of the knee joint, the angular position at O, θ_1 and F, ϕ are known (being the controlled angle). Thus, the other angular positions must also be formulated in terms of θ_1 and ϕ. l, β_1, β_2 and δ are intermediate variables.

$$l = \sqrt{\overrightarrow{|r_0|}^2 + \overrightarrow{|r_1|}^2 - 2|r_0||r_1|cos(\theta_1 - \phi + \pi)}$$

$$\beta_1 = \arcsin(\frac{|\overrightarrow{r_1}|}{l} sin(\theta_1 - \phi + \pi))$$

$$\beta_2 = \arccos \frac{|\overrightarrow{r_2}|^2 + l^2 - |\overrightarrow{r_3}|^2}{2|\overrightarrow{r_3}|l}$$

$$\delta = \arcsin(\frac{l}{|\overrightarrow{r_3}|} sin\beta_2)$$

$$\theta_2 = \beta_2 - \beta_1 + \phi - \pi$$

$$\theta_3 = -\beta_1 - \delta + \phi - \pi$$

Calculating Stability Criteria

Linkage vector in knee joint four bar linkage model

$$\overrightarrow{r_1} = \begin{bmatrix} L_1 cos\theta_1 \\ L_1 sin\theta_1 \\ 0 \end{bmatrix} \qquad \overrightarrow{r_2} = \begin{bmatrix} L_2 cos\theta_2 \\ L_2 sin\theta_2 \\ 0 \end{bmatrix} \qquad \overrightarrow{r_3} = \begin{bmatrix} L_3 cos\theta_3 \\ L_3 sin\theta_3 \\ 0 \end{bmatrix}$$

$$\overrightarrow{r_0} = \begin{bmatrix} L_0 cos\phi \\ L_0 sin\phi \\ 0 \end{bmatrix}$$

Here $\overrightarrow{r_0}$ is fixed and ϕ, the angle between $\overrightarrow{r_0}$ and x-axis is also fixed

L_0, L_1, L_2 and L_3 are the length of the three linkages given by $\overrightarrow{r_0}$, $\overrightarrow{r_1}$, $\overrightarrow{r_2}$ and $\overrightarrow{r_3}$ respectively.

L_0, L_1, L_2 and L_3 are the length of the three linkages given by $\overrightarrow{r_0}$, $\overrightarrow{r_1}$, $\overrightarrow{r_2}$ and $\overrightarrow{r_3}$ respectively.

Calculating Stability Criteria

Kinematics model of the Knee Joint

The following equations characterise the kinematic model of the knee joint.

$$\overrightarrow{r_1} + \overrightarrow{r_2} + \overrightarrow{r_3} + \overrightarrow{r_0} = \overrightarrow{0}$$

$$\overrightarrow{v_0} + \overrightarrow{v_1} + \overrightarrow{v_2} + \overrightarrow{v_3} = \overrightarrow{0}$$

$$\overrightarrow{a_0} + \overrightarrow{a_1} + \overrightarrow{a_2} + \overrightarrow{a_3} = \overrightarrow{0}$$

These aforementioned relationships arise as the four bar linkage is stationary with respect to F and O.

$$r_{1x} = L_1 cos\theta_1$$
$$r_{1y} = L_1 sin\theta_1$$
$$r_{1z} = 0$$

$$r_{0x} = L_0 cos\phi$$
$$r_{0y} = L_0 sin\phi$$
$$r_{0z} = 0$$

$$r_{1x} = L_1 cos\theta_1$$
$$r_{1y} = L_1 sin\theta_1$$
$$r_{1z} = 0$$

$$r_{1x} = L_1 cos\theta_1$$
$$r_{1y} = L_1 sin\theta_1$$
$$r_{1z} = 0$$

These relationships are shorthand for the components of the vectors $\overrightarrow{r_0}$, $\overrightarrow{r_1}$, $\overrightarrow{r_2}$ and $\overrightarrow{r_3}$

Calculating Stability Criteria

Calculating ω and α

$$\overrightarrow{\omega_0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad \overrightarrow{\omega_1} = \begin{bmatrix} 0 \\ 0 \\ \omega_{1z} \end{bmatrix} \qquad \overrightarrow{\omega_2} = \begin{bmatrix} 0 \\ 0 \\ \omega_{2z} \end{bmatrix} \qquad \overrightarrow{\omega_3} = \begin{bmatrix} 0 \\ 0 \\ \omega_{3z} \end{bmatrix}$$

$\overrightarrow{v_O} = 0$ The linkage $\overrightarrow{r_0}$ is fixed

$\overrightarrow{v_1} = \overrightarrow{\omega_1} \times \overrightarrow{r_1}$ Only rotation with ω_1 about origin of linkage

$\overrightarrow{v_2} = \overrightarrow{\omega_2} \times \overrightarrow{r_2}$ Only rotation with ω_2 about origin of linkage

$\overrightarrow{v_3} = \overrightarrow{\omega_3} \times \overrightarrow{r_3}$ Only rotation with ω_3 about origin of linkage

Calculating Stability Criteria

Calculating ω and α

$$\overrightarrow{v_0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad \overrightarrow{v_1} = \begin{bmatrix} -\omega_{1z} r_{1y} \\ \omega_{1z} r_{1x} \\ 0 \end{bmatrix} \qquad \overrightarrow{v_2} = \begin{bmatrix} -\omega_{1z} r_{1y} \\ \omega_{1z} r_{1x} \\ 0 \end{bmatrix} \qquad \overrightarrow{v_3} = \begin{bmatrix} -\omega_{1z} r_{1y} \\ \omega_{1z} r_{1x} \\ 0 \end{bmatrix}$$

Since $\overrightarrow{v_0} + \overrightarrow{v_1} + \overrightarrow{v_2} + \overrightarrow{v_3} = \overrightarrow{0}$, we get the following equations.

$$0 - \omega_{1z} r_{1y} - \omega_{2z} r_{2y} - \omega_{3z} r_{3y} = 0$$

$$0 + \omega_{1z} r_{1x} + \omega_{2z} r_{2x} + \omega_{3z} r_{3x} = 0$$

Solving the above two we get ω_{2z} and ω_{3z}

$$\omega_{2z} = -\omega_{1z} \frac{r_{1y} r_{3x} - r_{3y} r_{1x}}{r_{2y} r_{3x} - r_{3y} r_{2x}}$$

$$\omega_{3z} = -\omega_{1z} \frac{r_{2y} r_{1x} - r_{1y} r_{2x}}{r_{2y} r_{3x} - r_{3y} r_{2x}}$$

Calculating Stability Criteria

Calculating ω and α

$$\overrightarrow{\alpha_0} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad \overrightarrow{\alpha_1} = \begin{bmatrix} 0 \\ 0 \\ \alpha_{1z} \end{bmatrix} \qquad \overrightarrow{\alpha_2} = \begin{bmatrix} 0 \\ 0 \\ \alpha_{2z} \end{bmatrix} \qquad \overrightarrow{\alpha_3} = \begin{bmatrix} 0 \\ 0 \\ \alpha_{3z} \end{bmatrix}$$

$\overrightarrow{v_O} = 0$ — The linkage $\overrightarrow{r_0}$ is fixed

$\overrightarrow{a_1} = \overrightarrow{\alpha_1} \times \overrightarrow{r_1} + \overrightarrow{\omega_1} \times \overrightarrow{\omega_1} \times \overrightarrow{r_1}$ — Only rotation with α_1 about origin of linkage

$\overrightarrow{a_2} = \overrightarrow{\alpha_2} \times \overrightarrow{r_2} + \overrightarrow{\omega_2} \times \overrightarrow{\omega_2} \times \overrightarrow{r_2}$ — Only rotation with α_2 about origin of linkage

$\overrightarrow{v_3} = \overrightarrow{\omega_3} \times \overrightarrow{r_3}$ — Only rotation with α_3 about origin of linkage

Calculating Stability Criteria

Calculating ω and α

Since $\overrightarrow{a_0} + \overrightarrow{a_1} + \overrightarrow{a_2} + \overrightarrow{a_3} = \overrightarrow{0}$, we get the following equations.

$$0 - \alpha_{1z} r_{1y} - \omega_{1z}^2 r_{1x} - \alpha_{2z} r_{2y} - \omega_{2z}^2 r_{2x} - \alpha_{3z} r_{3y} - \omega_{3z}^2 r_{3x} = 0$$

$$0 + \alpha_{1z} r_{1x} - \omega_{1z}^2 r_{1y} + \alpha_{2z} r_{2x} - \omega_{2z}^2 r_{2y} + \alpha_{3z} r_{3x} - \omega_{3z}^2 r_{3y} = 0$$

Solving the above two we get α_{2z} and α_{3z}

$$\alpha_{2z} = \frac{r_{3x}(-\alpha_{1z} r_{1y} - \omega_{1z}^2 r_{1x} - \omega_{2z}^2 r_{2x} - \omega_{3z}^2 r_{3x}) - r_{3y}(-\alpha_{1z} r_{1x} - \omega_{1z}^2 r_{1y} - \omega_{2z}^2 r_{2y} - \omega_{3z}^2 r_{3y})}{r_{2y} r_{3x} - r_{3y} r_{2x}}$$

$$\alpha_{3z} = \frac{-r_{2x}(-\alpha_{1z} r_{1y} - \omega_{1z}^2 r_{1x} - \omega_{2z}^2 r_{2x} - \omega_{3z}^2 r_{3x}) + r_{2y}(-\alpha_{1z} r_{1x} - \omega_{1z}^2 r_{1y} - \omega_{2z}^2 r_{2y} - \omega_{3z}^2 r_{3y})}{r_{2y} r_{3x} - r_{3y} r_{2x}}$$

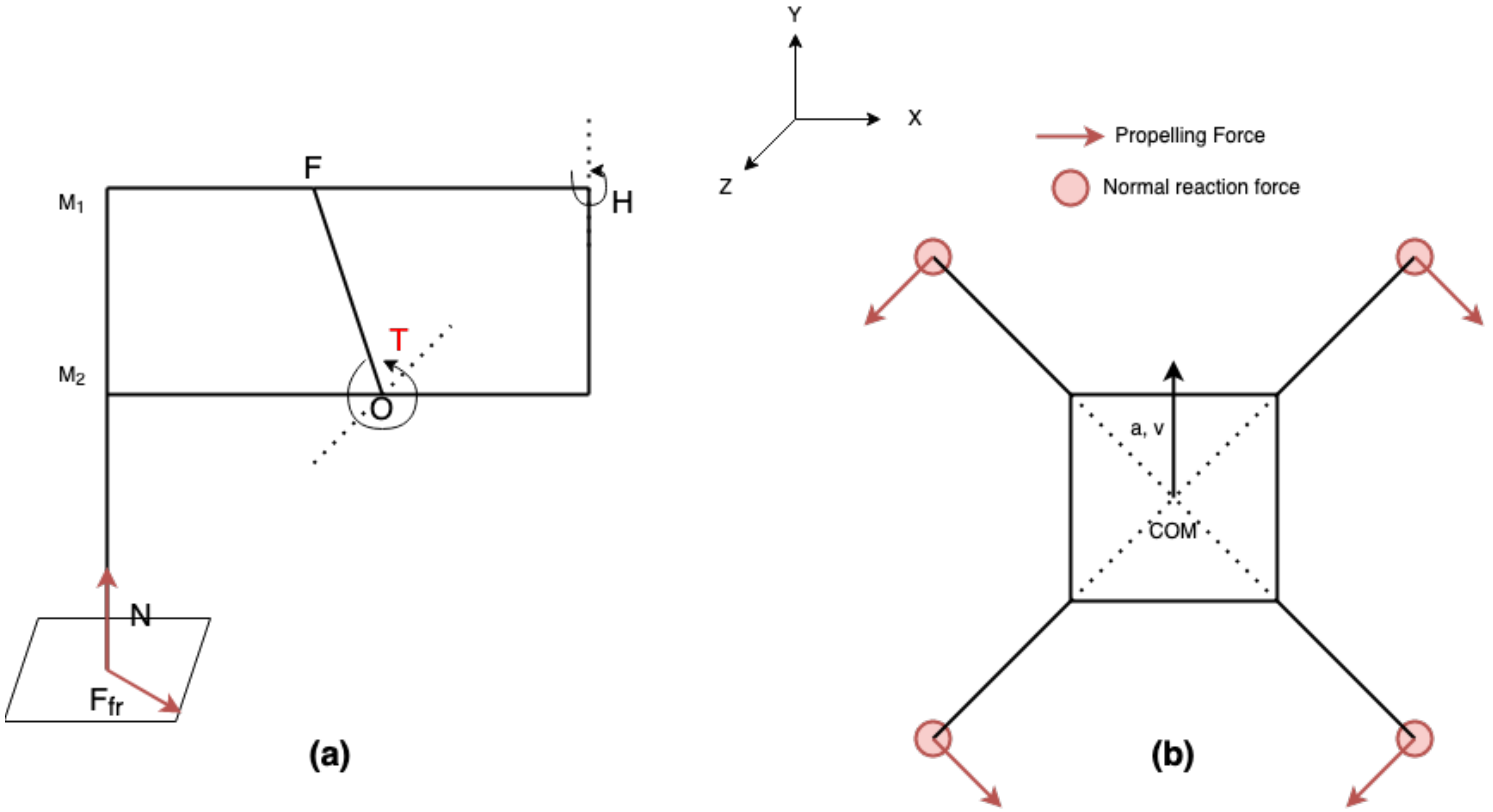
Calculating Stability Criteria

Dynamic Model of the Available Quadruped

$$\sum \overrightarrow{F_{fr}} + \sum \overrightarrow{N} - m\overrightarrow{g} = m\overrightarrow{a}$$

$$\sum_i \overrightarrow{F_{fr}} \times R_i + \sum_i \overrightarrow{N} \times R_i = I_R \overrightarrow{\alpha_R}$$

The above two equations depict the dynamics of the quadruped in vector form. Rotation is possible about the y and z axes, while translation is only possible along the y and z axes.

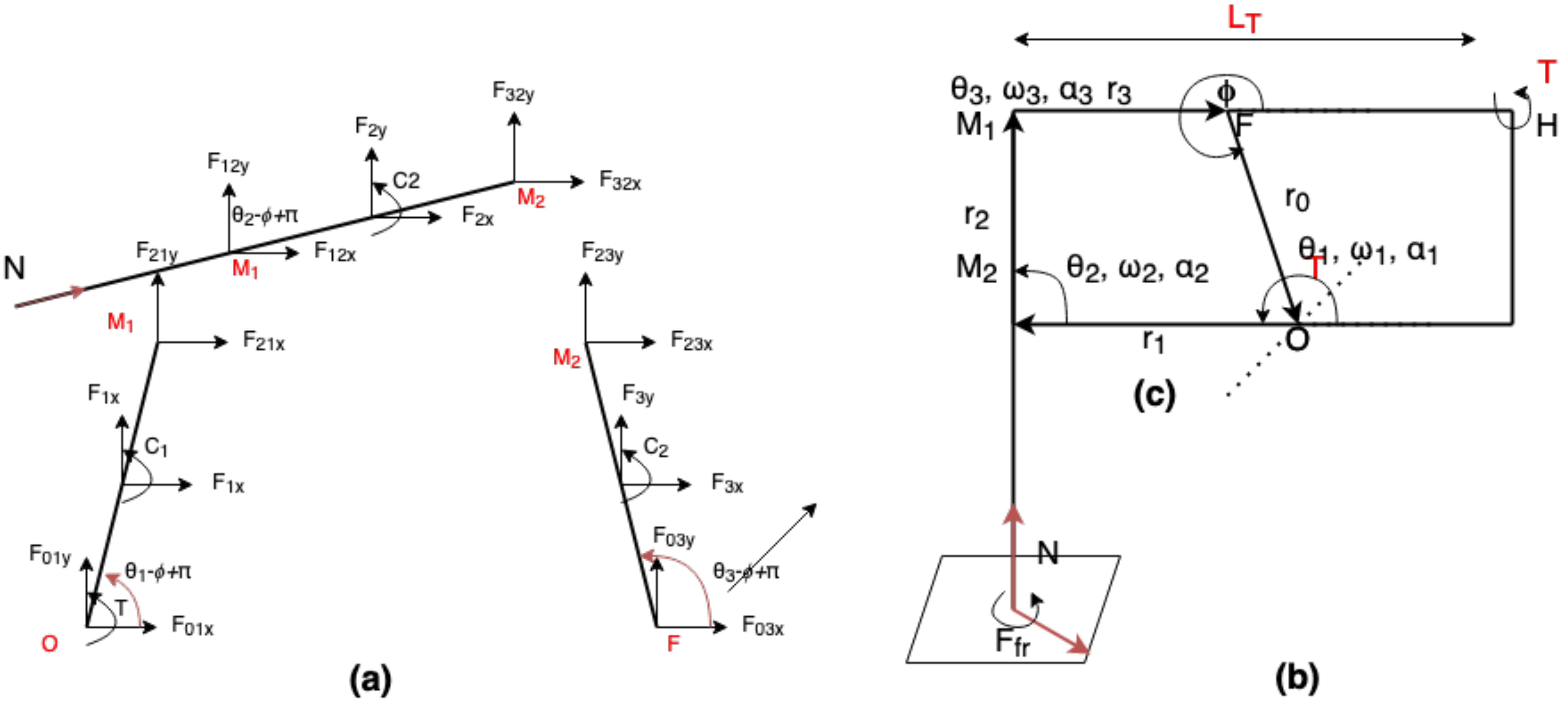


Calculating Stability Criteria

Stance Dynamics

a. Illustration of all forces acting on the four bar linkage in the knee during stance phase. The segment M1H will be rigid during stance

b. A simplified model of a leg. N and F_{fr} are normal reaction and friction forces respectively



Calculating Stability Criteria

Stance Dynamics

$$F_{01x} + F_{1x} + F_{21x} = 0$$

$$F_{01y} + F_{1y} + F_{21y} = 0$$

$$F_{12x} + F_{2x} + F_{32x} + N\cos\theta_2 - \phi + \pi = 0$$

$$F_{12y} + F_{2y} + F_{32y} + N\sin\theta_2 - \phi + \pi = 0$$

$$F_{01x} + F_{1x} + F_{21x} = 0$$

$$F_{01y} + F_{1y} + F_{21y} = 0$$

$$C_1 = 0$$

$$C_2 = 0$$

$$C_3 = 0$$

$$\overrightarrow{F_{23}} = -\overrightarrow{F_{32}}$$

$$\overrightarrow{F_{21}} = -\overrightarrow{F_{12}}$$

$$T + C_1 - F_{1x}\frac{|\overrightarrow{r_1}|}{2}\cos\theta_1 - \phi + \pi + F_{1y}\frac{|\overrightarrow{r_1}|}{2}\cos\theta_1 - \phi + \pi + F_{21y}|\overrightarrow{r_1}|\cos\theta_1 - \phi + \pi - F_{21x}|\overrightarrow{r_1}|\sin\theta_1 - \phi + \pi = 0$$

$$C_2 - F_{2x}\frac{|\overrightarrow{r_2}|}{2}\cos\theta_2 - \phi + \pi + F_{2y}\frac{|\overrightarrow{r_2}|}{2}\cos\theta_2 - \phi + \pi + F_{32y}|\overrightarrow{r_2}|\cos\theta_2 - \phi + \pi - F_{32x}|\overrightarrow{r_2}|\sin\theta_2 - \phi + \pi = 0$$

$$C_3 - F_{3x}\frac{|\overrightarrow{r_3}|}{2}\cos\theta_3 - \phi + \pi + F_{3y}\frac{|\overrightarrow{r_3}|}{2}\cos\theta_3 - \phi + \pi - F_{23y}|\overrightarrow{r_1}|\cos\theta_1 - \phi + \pi + F_{23x}|\overrightarrow{r_1}|\sin\theta_1 - \phi + \pi = 0$$

Calculating Stability Criteria

Calculating $\overrightarrow{F_{fr}}$ and $\overrightarrow{N}$

- $\overrightarrow{F_{fr}}$ and $\overrightarrow{N}$ are the friction and normal reaction acting at the points of contact with the ground
- To compute these forces the dynamics at the point of contact need to be considered
- For a successful walking motion, there must be no slipping at the point of contact
- Thus, we need to solve for equilibrium conditions there

$$\sum_{i}^{k} N_i - mg = 0$$

where k is the number of legs in contact with the group

$$T - F_{fr} L_T = 0$$

This is the condition of no slipping at the point of contact with the ground

$$\sum \overrightarrow{F_{fr}} = m\overrightarrow{a}$$

The sum of all frictional forces are responsible for the horizontal acceleration a of the quadruped

Calculating Stability Criteria

Calculating Inertial Force and Moment at COM

- The Inertial Force at the COM is given by $m\overrightarrow{a}$, where $\overrightarrow{a}$ is the acceleration of the COM
- The Moment at COM is given by $I_R\overrightarrow{\alpha_R}$, which is calculated by solving the dynamics of the quadruped
- Once the Inertial Force and Moment at the COM are known, the modified ZMP can computed using the method proposed in [2], which can then be used to compute the stability criteria

Solving Quadruped Dynamics

Solving a system of linear equations

- The dynamics equations of the quadruped together form a system of 69 equations and 69 variables
- $AX = B$ is a system of linear equations with X being the vector of the unknowns, A the matrix of coefficients and B the constants
- $X = A^{-1}B$ gives the solution to the aforementioned system of linear equations

Solving Quadruped Dynamics

Solving a system of linear equations

- Some simplifying assumptions were made to formulate the system of linear equations
  - A binary factor γ_i was introduced for each leg. If $\gamma_i = 0$ there is no contact with the ground at the leg, else there is contact
  - If $\gamma_i = \gamma_j = 0$, then $N_i = N_j = 0$.
  - If $\gamma_i = \gamma_j = 1$, then $N_i = N_j \neq 0$
  - The resistive force at the hip joint is given by ϵ_i when leg i is not in contact with the ground
  - Movement is on a flat surface with coefficient of friction μ
  - At least two legs are in contact with the ground at all points of time

Energy Efficiency Criteria

Calculation of metric E

- Energy Criteria E measures the energy efficiency of the quadruped
- The Criteria is inspired by the objective function used in [9]
- There are three aspects of the Energy Efficiency Criteria
    - Mean Power P_{avg}
    - Mean Power Derivation D_{avg}
    - Mean Torque Consumption P_L
- $E = P_{avg} + D_{avg} + P_L$

$$P_{avg} = \frac{1}{T} \sum_{i=1}^{4} \sum_{j=1}^{2} \int_0^T |\tau_{i,j} \dot{\theta}_{i,j}(t)| dt$$

$$D_{avg} = \sqrt{\frac{1}{T} \int_0^T (\sum_{i=1}^{4} \sum_{1}^{2} \tau_{i,j} \dot{\theta}_{i,j}(t) - P_{avg})^2 dt}$$

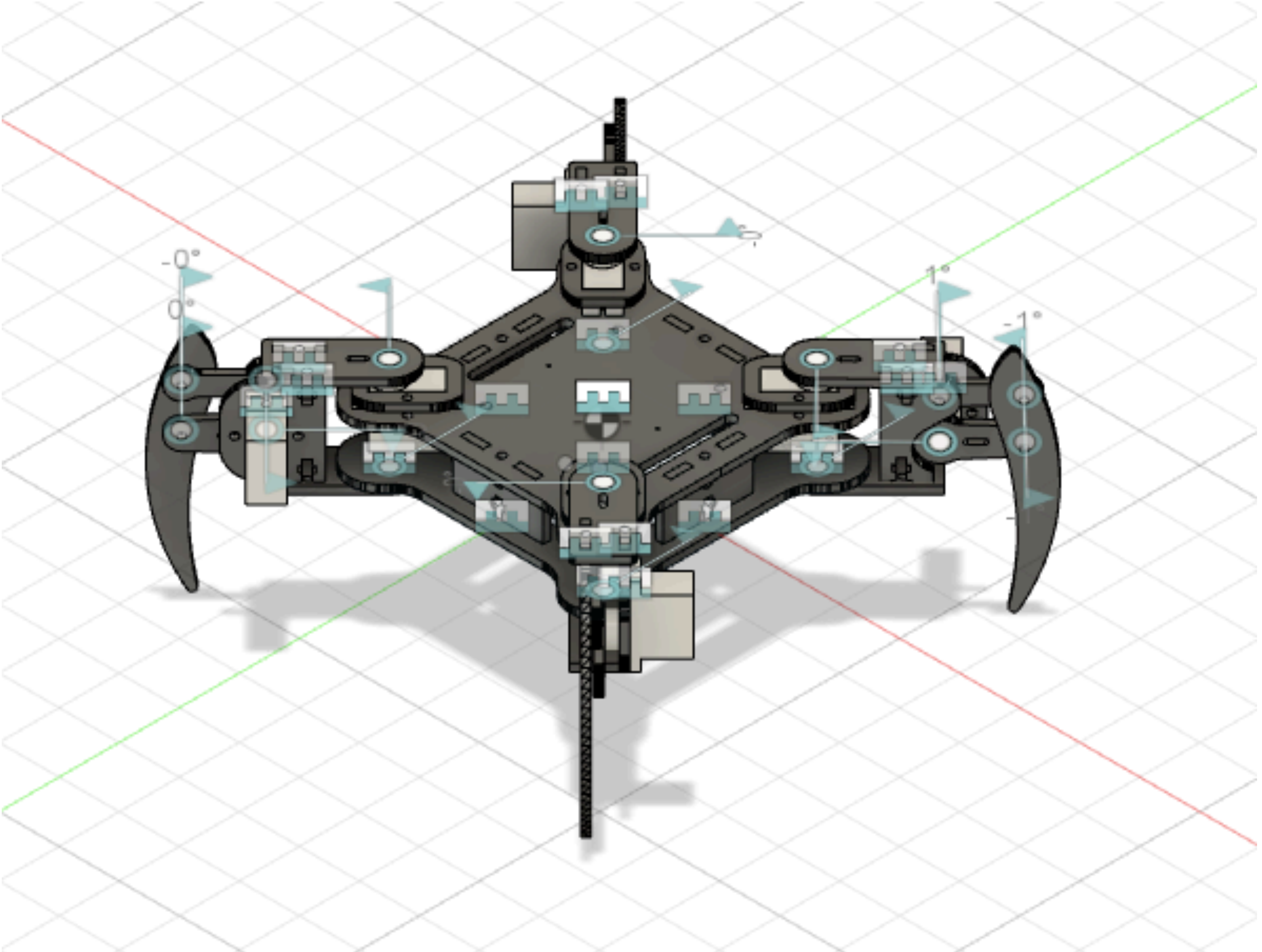
$$P_L = \frac{1}{T} \sum_{i=1}^{4} \sum_{j=1}^{2} \int_0^T (\tau_{i,j}(t))^2 dt$$

3D model

Quadruped Assembly on Fusion 360

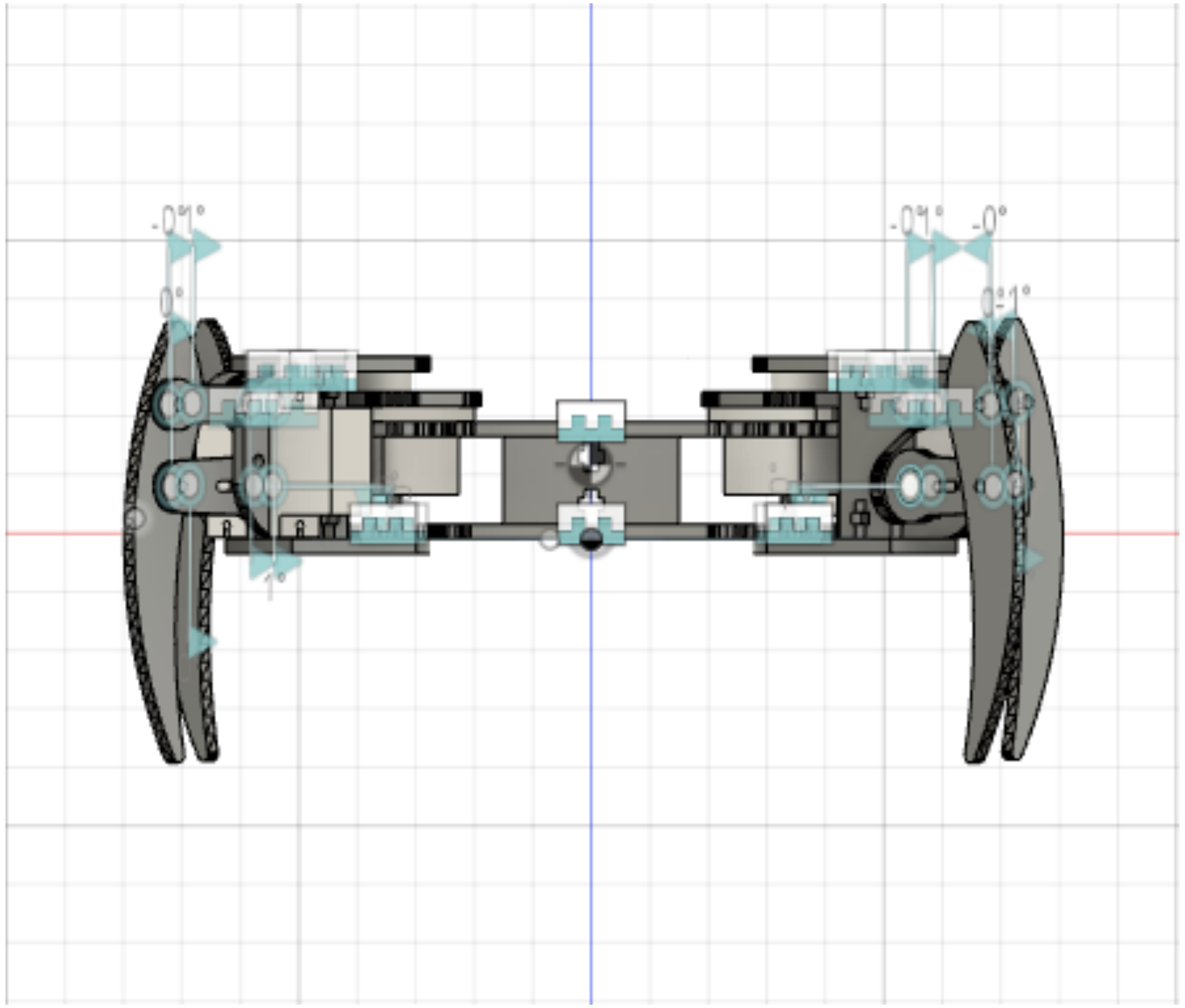
Need for 3D model-

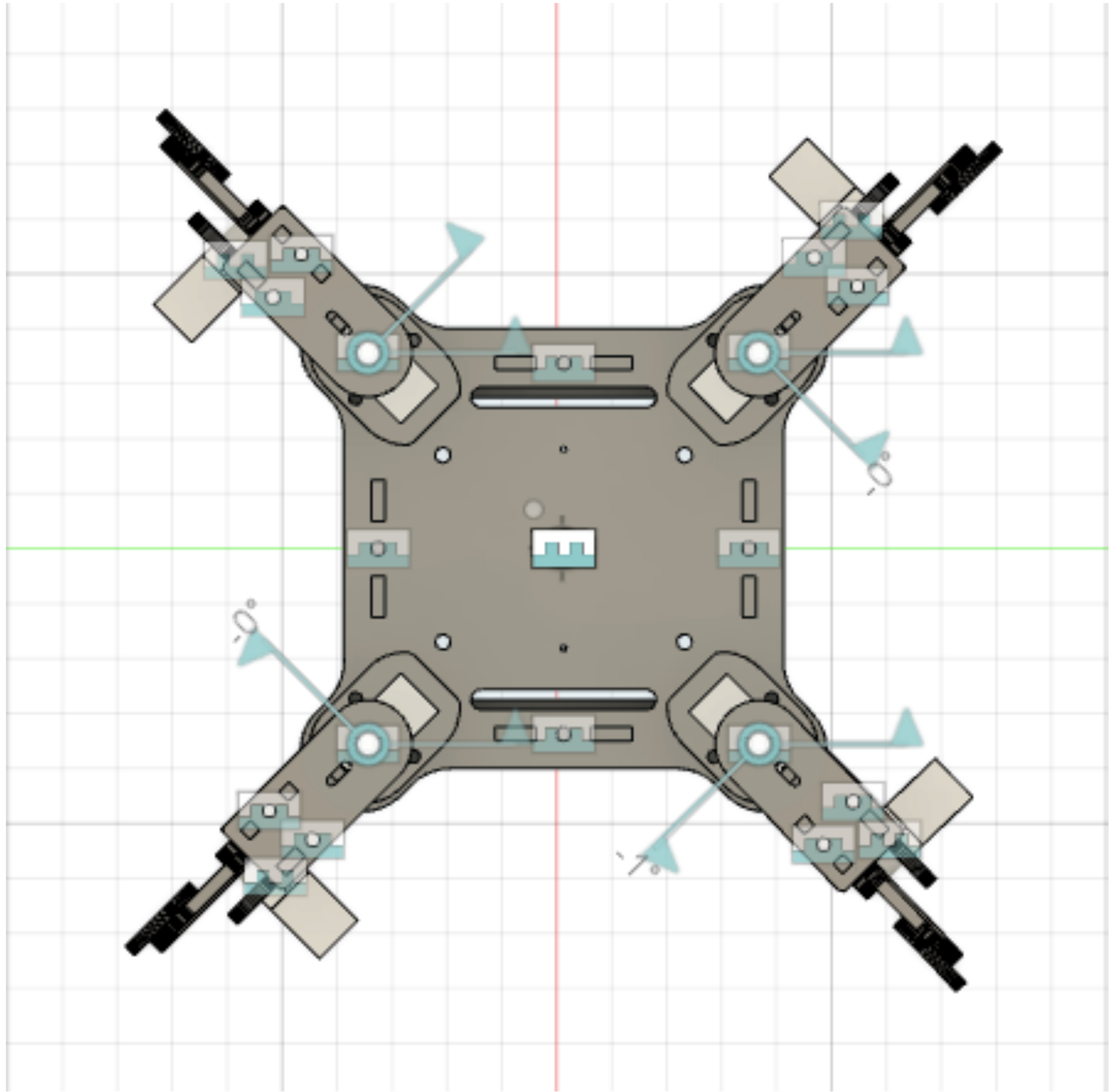
- Precise mass of quadruped components
- Precise Position of leg COM
- Moment of Inertia in Knee four bar linkage
- Easier method for URDF model generation
- Calculating Joint motion ranges
- Fusion 360 can also be used for ML



3D model

Quadruped Assembly on Fusion 360



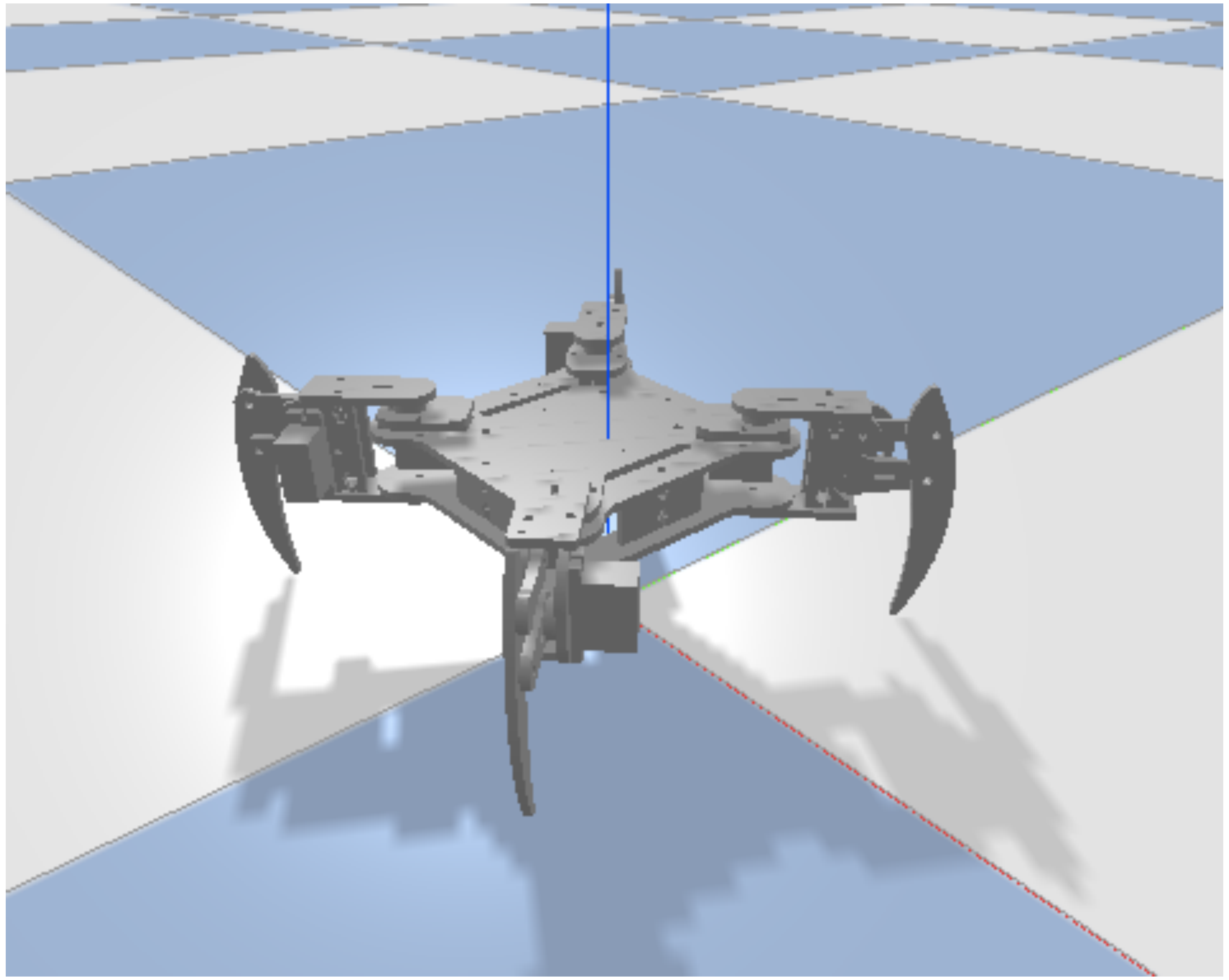


PyBullet

Collision and Inertia Model of the quadruped

- Collisions in the quadruped to be modelled using simple shapes like cylinder, boxed and spheres

- There is a need for a simpler model as the collision model using meshes requires compute power not currently available



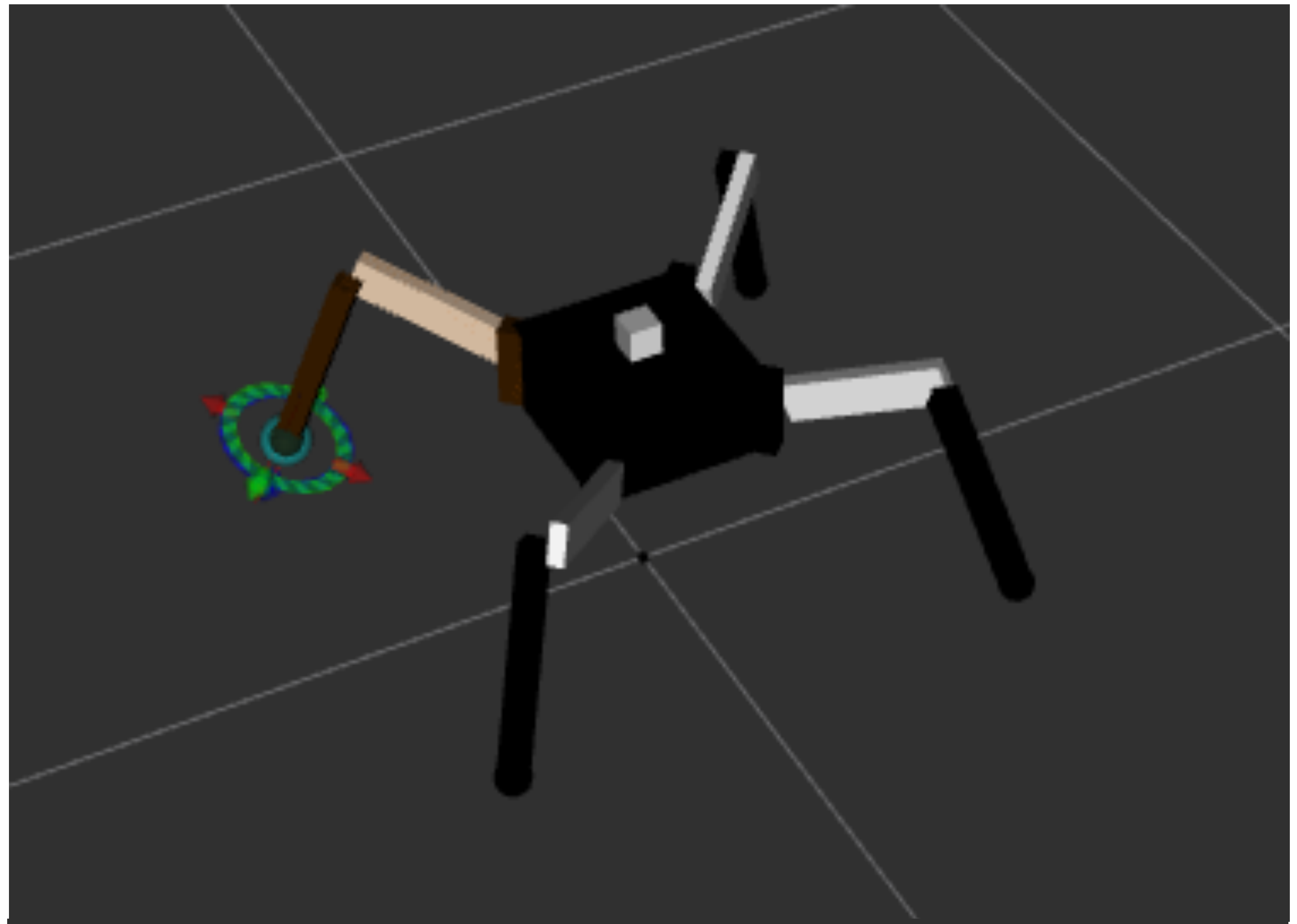
ROS and Gazebo

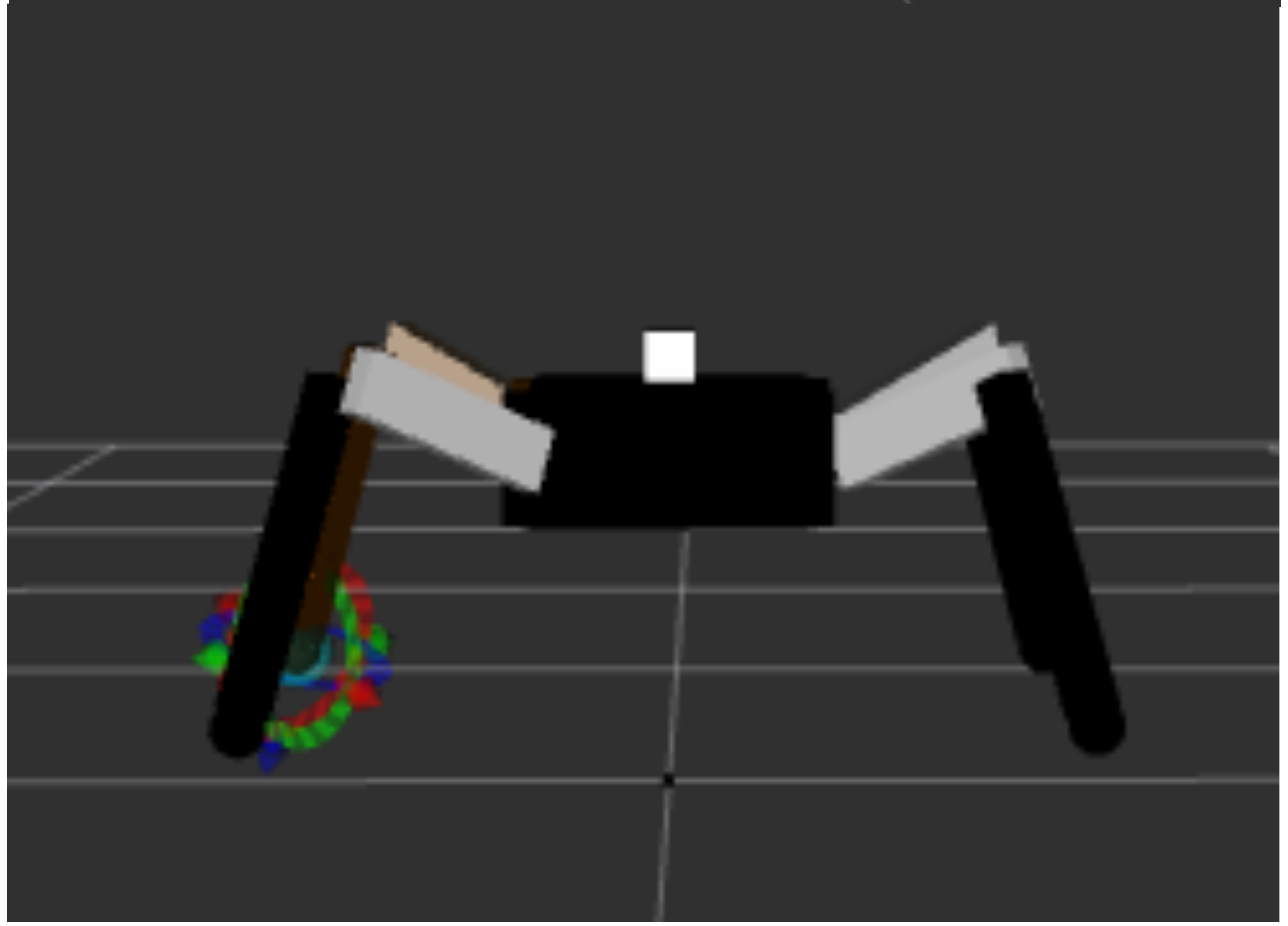
- Due to the limitations of the PyBullet API, the environment for RL had to be changed from PyBullet to ROS and Gazebo
- ROS implementation has a much lower simulation to hardware transfer time and Gazebo provides the flexibility of adding sensors into the simulations as needed
- Pybullet lacks a forward kinematics module that is needed for computing the future contact positions for the Stability Criteria
- MoveIt! Plugin with Gazebo can be very easily be used to get forward as well as inverse kinematics of the quadruped
- Gazebo also simplifies Stability Criteria Calculation as it directly provides callbacks for dynamics and kinematics from the physics engine used

ROS and Gazebo

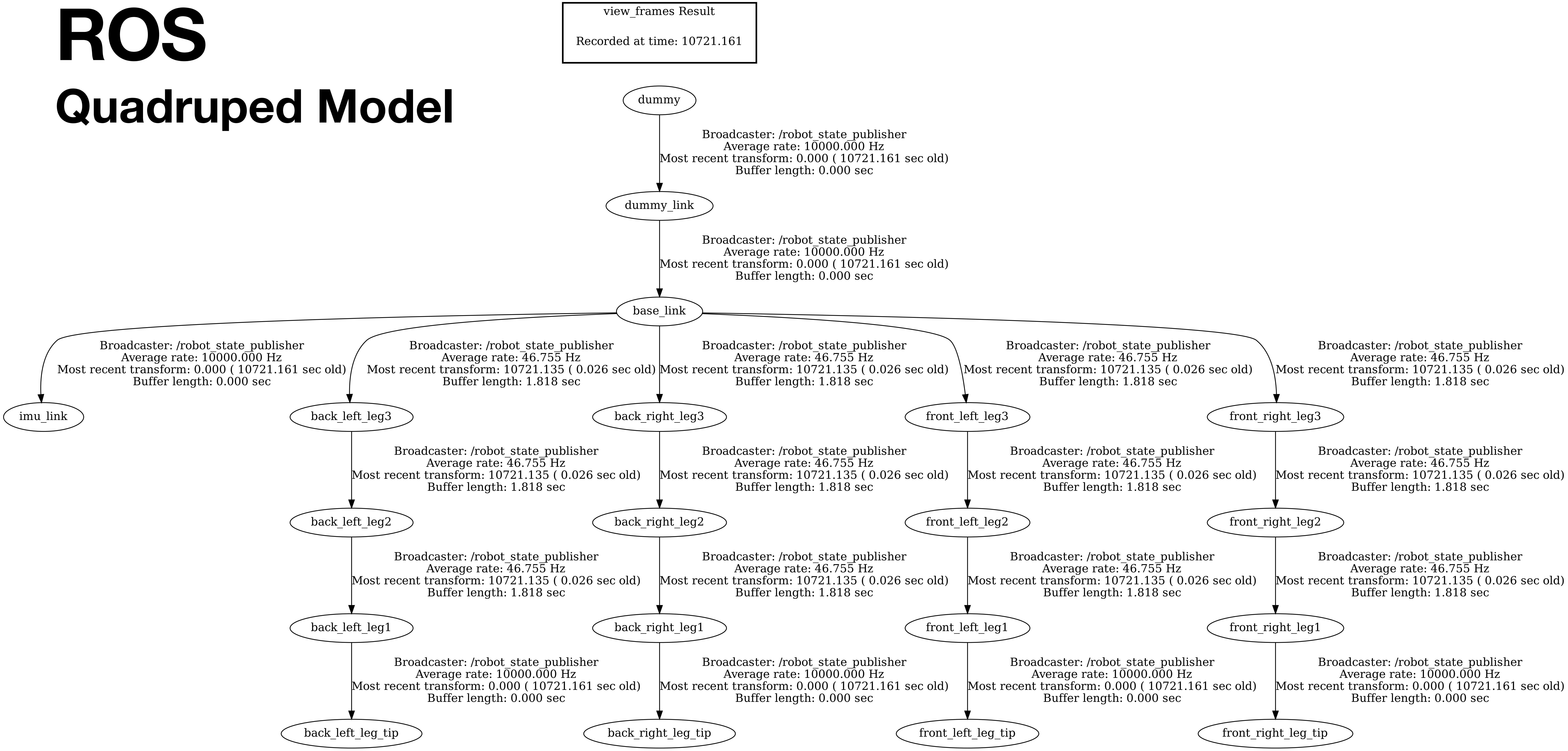
Simplified Quadruped Model

- The initially developed quadruped URDF model required very high computational power for proper simulation, which is not available
- A simplified URDF model was developed to simulate the kinematics and dynamics of the quadruped





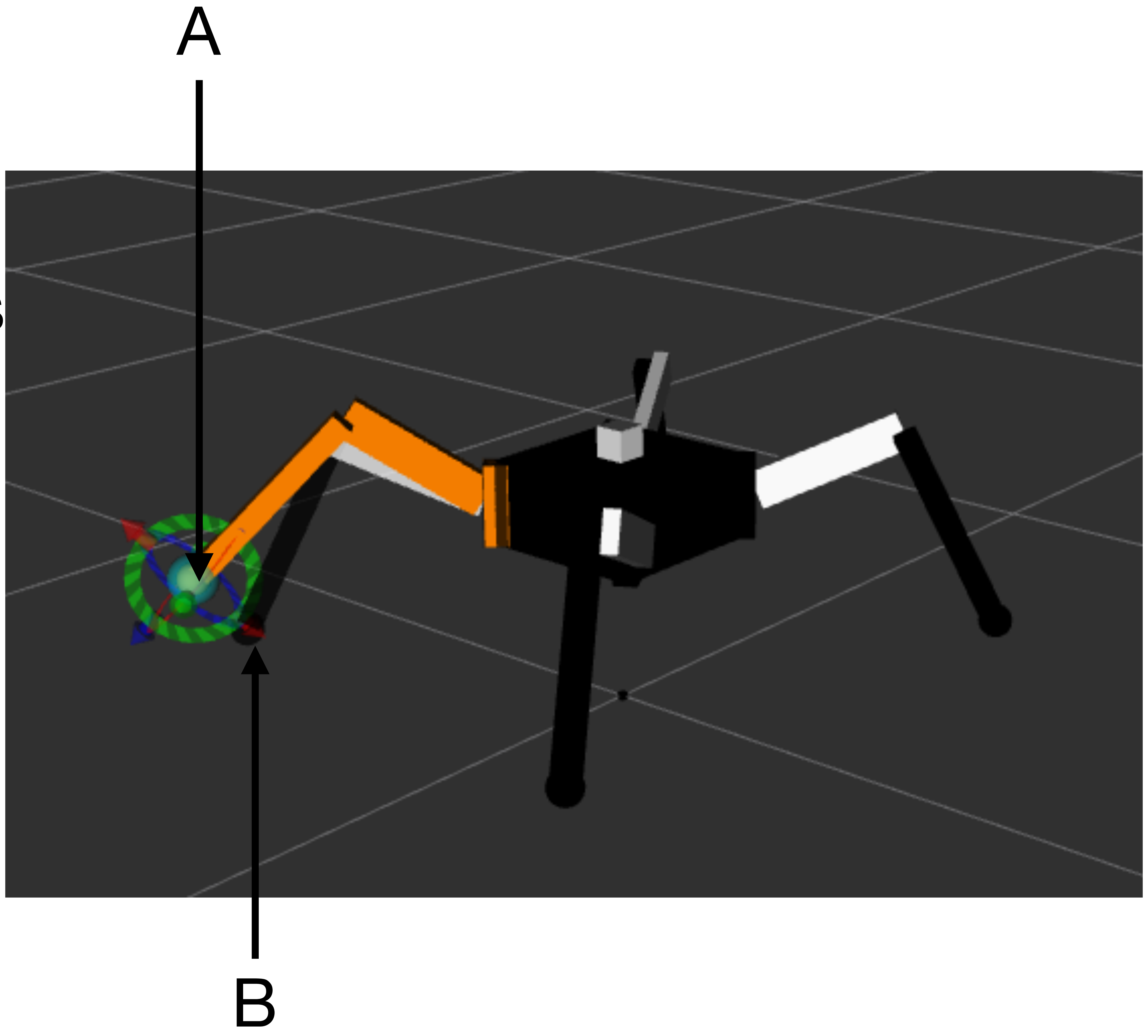
ROS
Quadruped Model



ROS

Quadruped Model - MoveIt!

- “MoveIt!” is a ROS path planning package used to plan manipulator paths
- “MoveIt!” was used to track the position of as end-effector, that is the leg tip as the leg moved from point A to point B
- This package was used in the computation of the stability reward and replaced the previous derived kinematics model of the quadruped



ROS

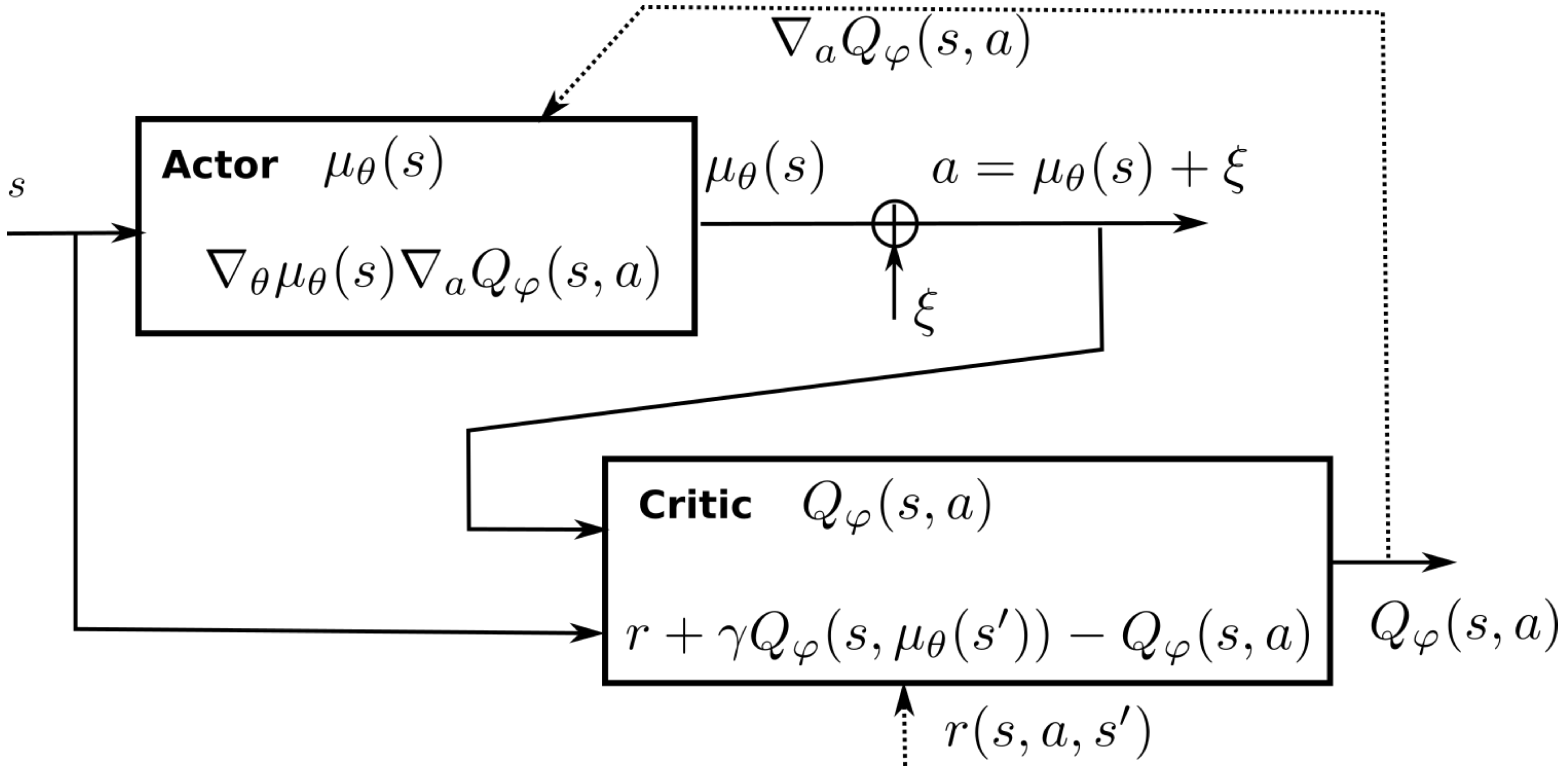
Sensors

The following sensors are mounted on the quadruped simulation model:

- IMU sensor:
  - To obtain orientation, linear acceleration and angular velocity
  - Sensed information used to construct robot state
  - Mounted at the top of the base link
- Contact Sensors
  - To obtain Point of contact, normal forces and other contact related information
  - Used to compute stability criteria of the reward

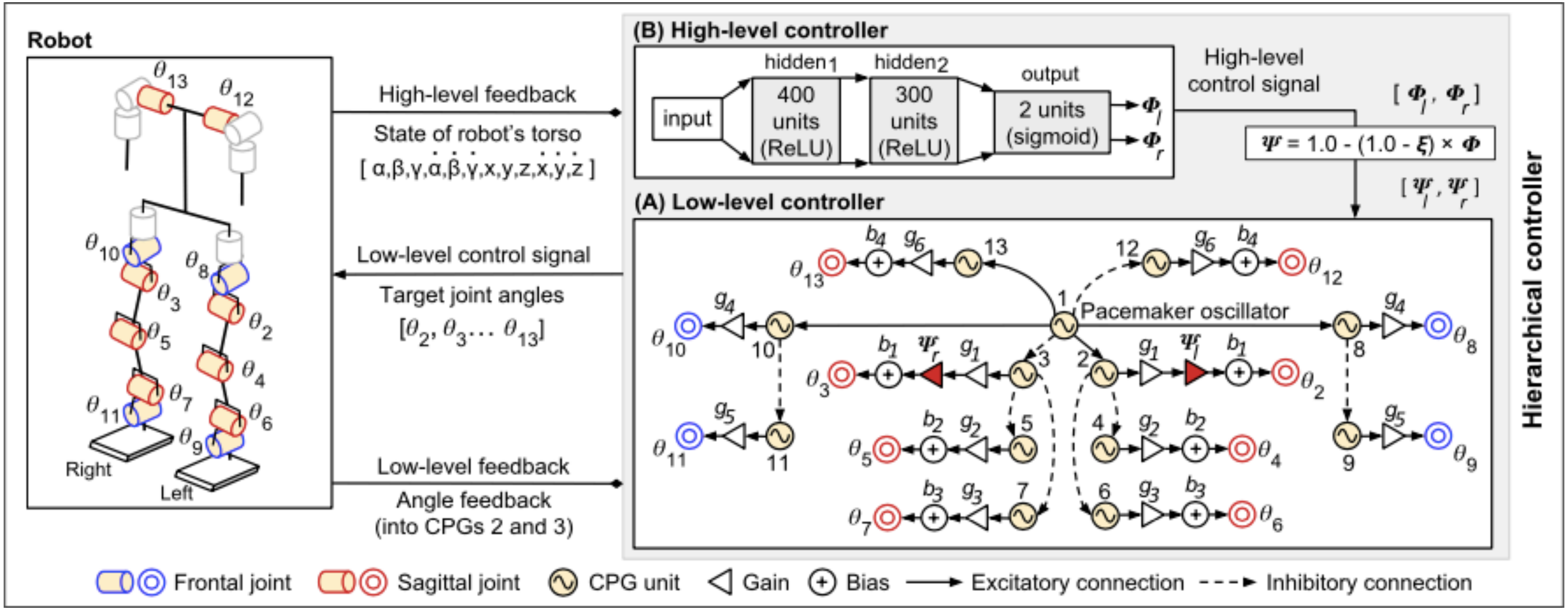
Reinforcement Learning

Deep Deterministic Policy Gradient



Reinforcement Learning

DDPG : Inspiration



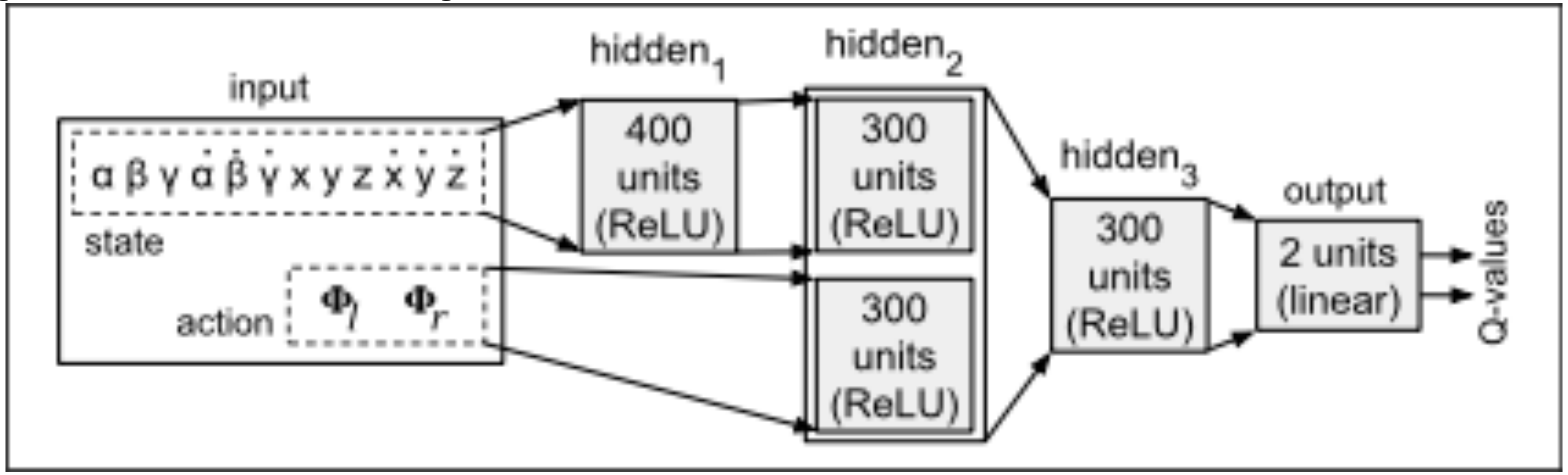
Reinforcement Learning

DDPG : Inspiration

- The proposed DDPG model is inspired by the DDPG for the higher level controller for the biped, proposed in [6]
- The CPG model in [6] is tuned using GA instead of inclusion in the DDPG
- The Actor Network in [6] produces the ψ_l and ψ_r which are used to steer the biped by modifying the gait signal according to the following equations:

$$\theta_2 = o_2 \Psi_l g_1 + b_1$$
$$\theta_3 = o_3 \Psi_r g_1 + b_1$$

- The proposed DDPG model combines CPG and MLP training to tune gait generation in a single step

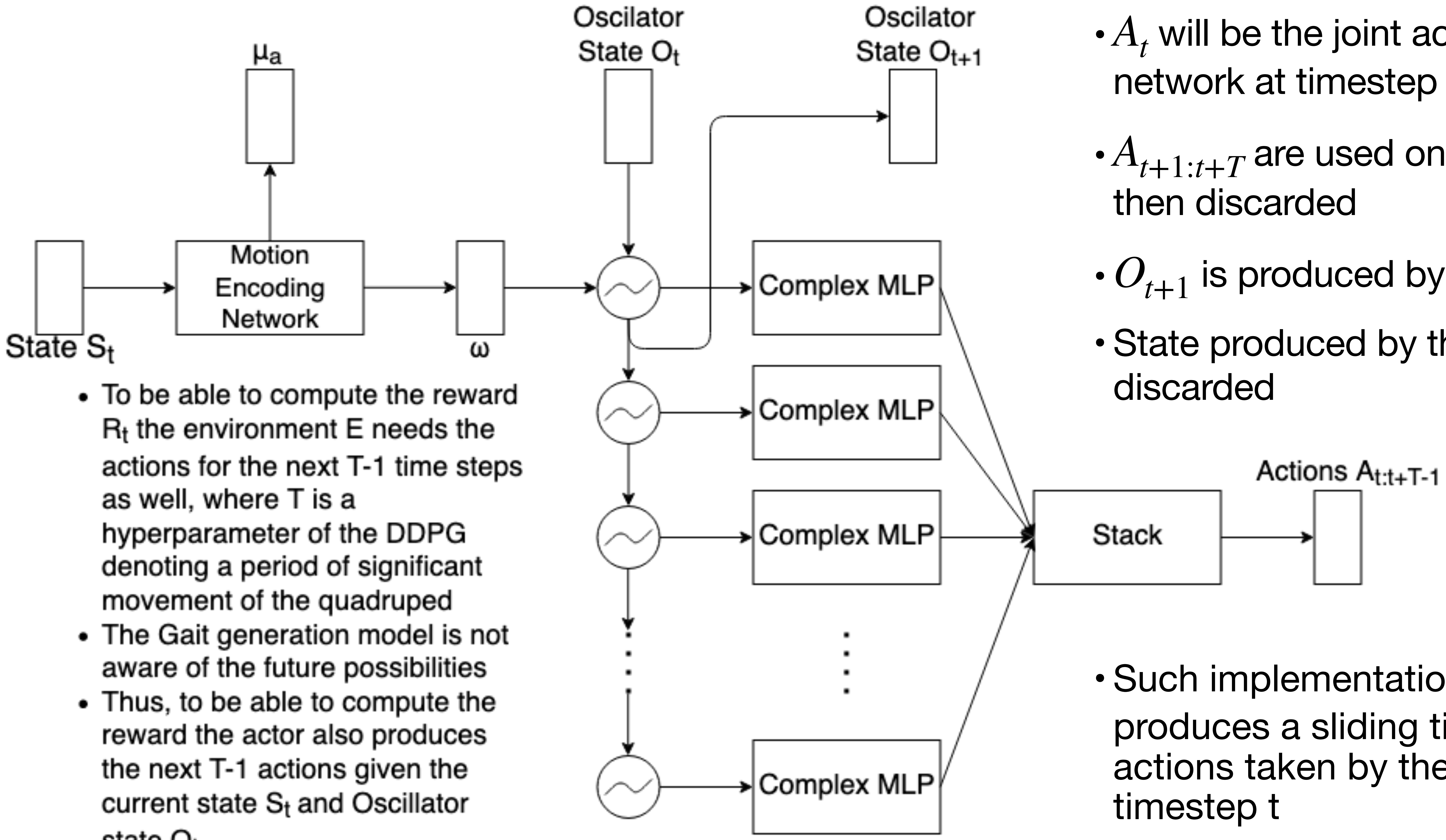


Critic Network in [6]

Reinforcement Learning

Actor Network

- A_t will be the joint activations produced by the network at timestep t
- $A_{t+1:t+T}$ are used only for reward calculation and then discarded
- O_{t+1} is produced by the CPG at timestep t
- State produced by the CPG after timestep t are discarded



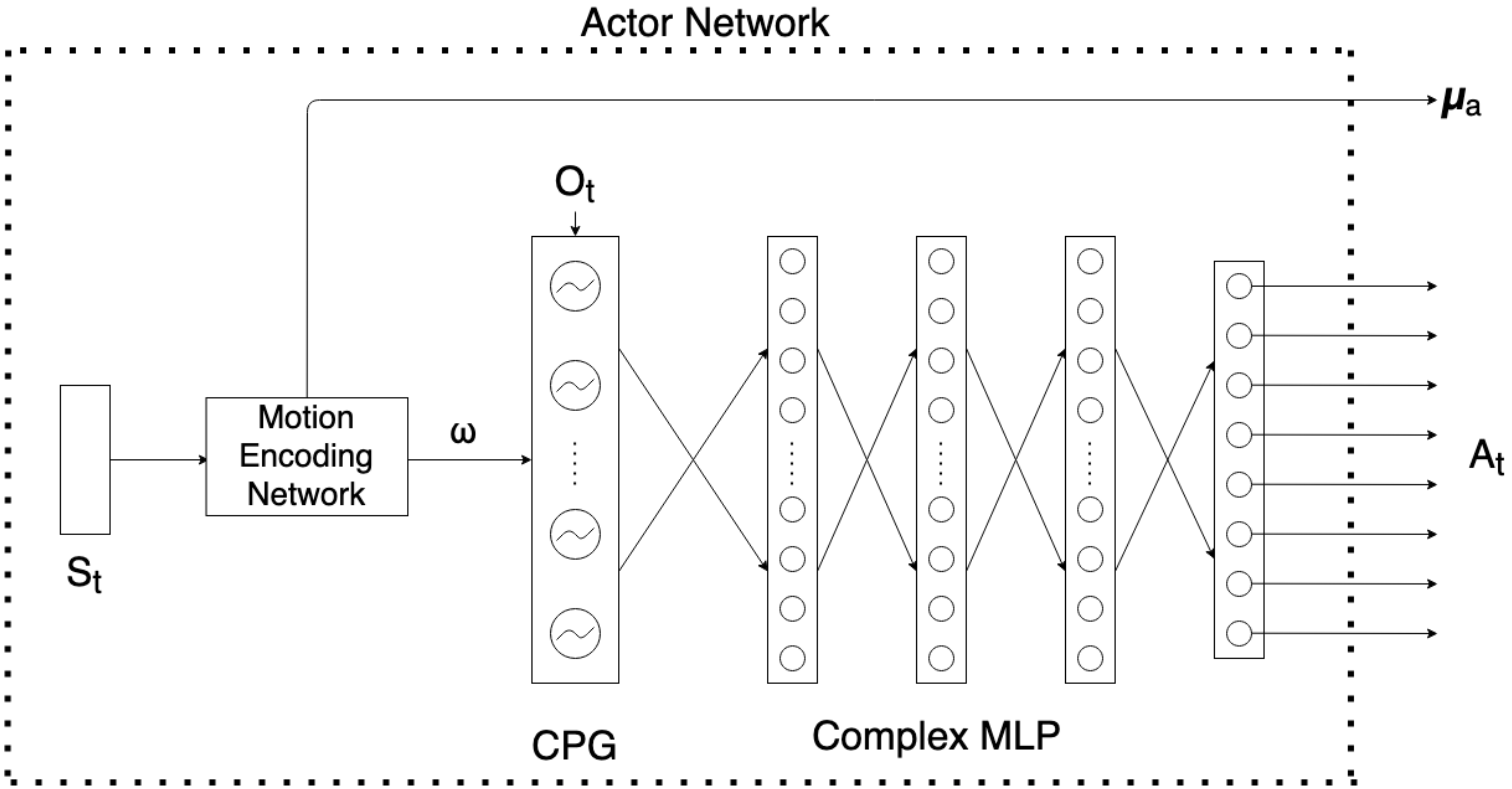
- To be able to compute the reward R_t the environment E needs the actions for the next T-1 time steps as well, where T is a hyperparameter of the DDPG denoting a period of significant movement of the quadruped
- The Gait generation model is not aware of the future possibilities
- Thus, to be able to compute the reward the actor also produces the next T-1 actions given the current state S_t and Oscillator state O_t

Unrolled Actor Network

- Such implementation of the Actor Network produces a sliding time window over $2T + 1$ actions taken by the quadruped around any timestep t

Reinforcement Learning

Snapshot of Actor Network during inference, at timestep t



Reinforcement Learning

Snapshot of Actor Network during inference, at timestep t

- The Actor gives two outputs μ_a and A_t
- μ_a is the normalised gait amplitude. The gait amplitude is limited within a range of $[0, \frac{\pi}{3}]$. $\mu_a = 1$ implies amplitude of $\frac{\pi}{3}$
- A_t is the normalised gait signal
- The vector $\mu_a \circ A_t$ is fed to the quadruped for gait generation
- The Normalisation is performed to improve convergence of the Actor model during pre-training

Reinforcement Learning

Actor Network: Oscillator Layer

- Hopf Oscillator was used within the Oscillator layer
- The oscillator was chosen because of its limit cycle behaviour at all parameter values
- A Oscillator Layer serves as the recurrent layer within the actor and generates the basis signals which are then combined by the complex MLP to produce the gait signal

$$\dot{z} = (\mu - | z |^2)z - z\omega$$

Complex Equation of Hopf Oscillator

$$x_{t+1} = x_t + (-y\omega + x(\mu - x^2 - y^2))dt$$

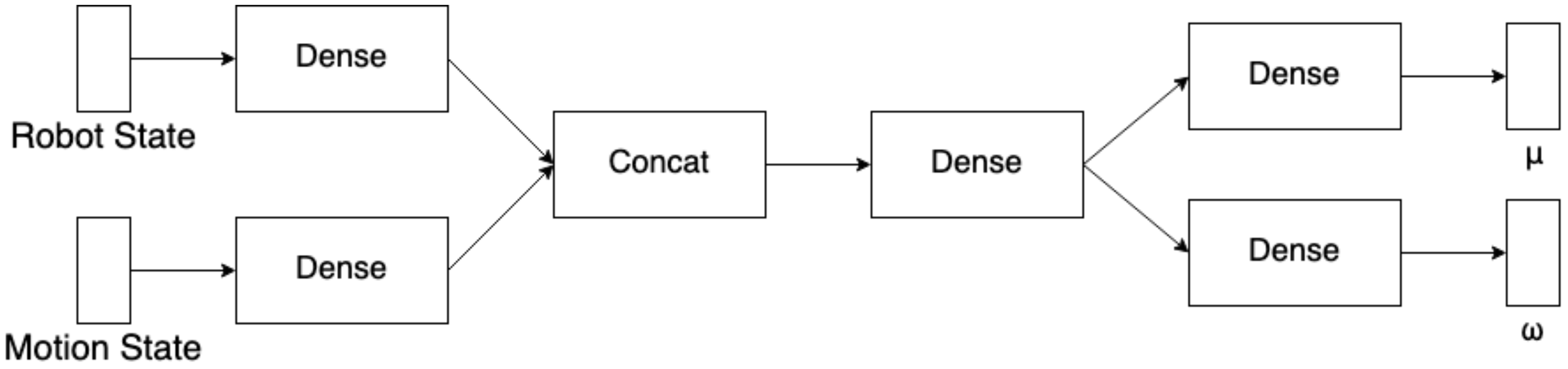
$$y_{t+1} = y_t + (x\omega + y(\mu - x^2 - y^2))dt$$

Cartesian Equations of Hopf Oscillator

Reinforcement Learning

Actor Network : Motion Encoding Network

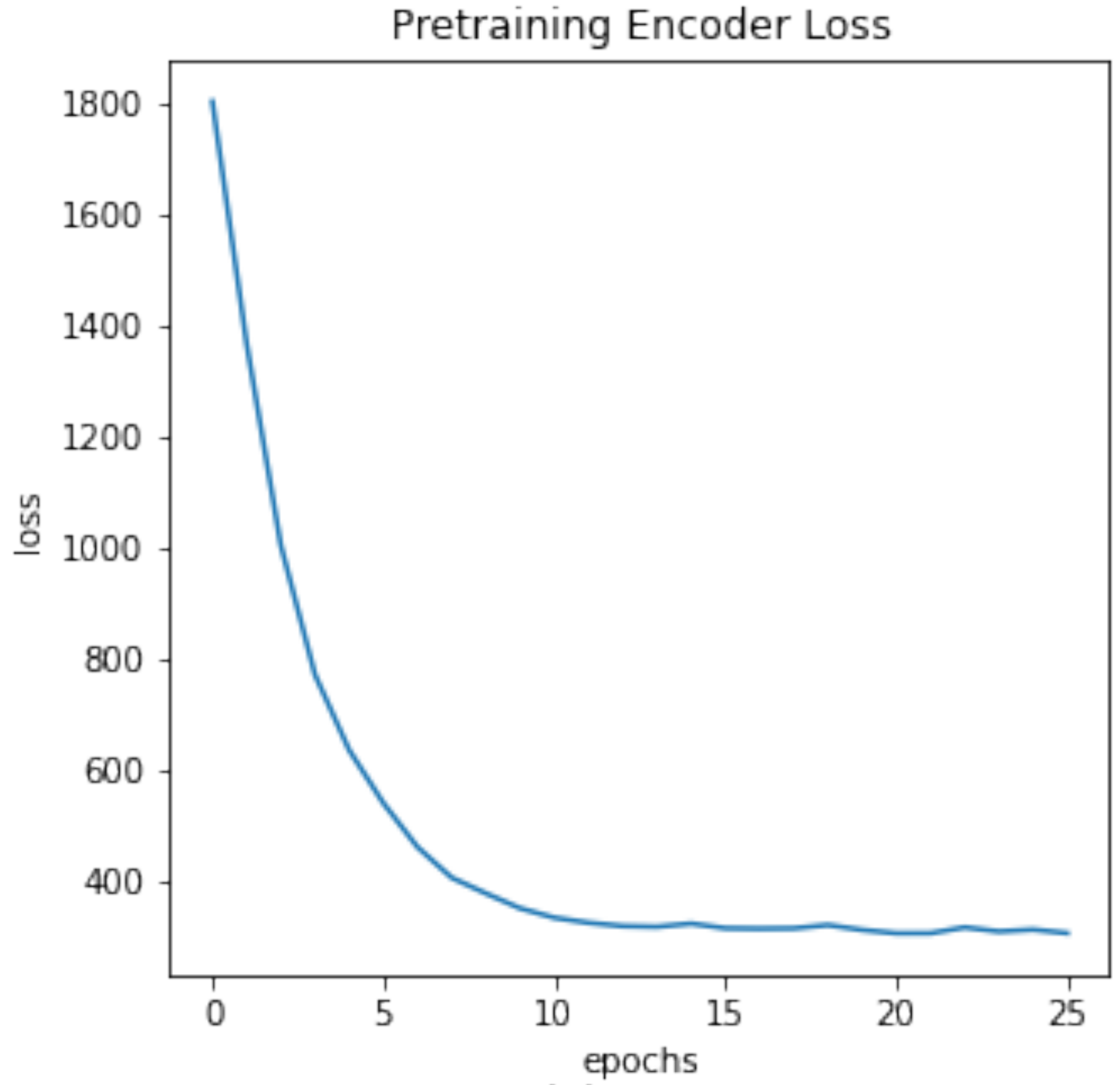
- S_t comprises of the following components:
  - Robot State:
    - Current Joint Angles
    - Difference between current and last joint positions
    - Orientation
    - Angular Velocity
    - Linear Acceleration
  - Motion State :
    - Desired Turning
    - Desired Speed along x and y axes

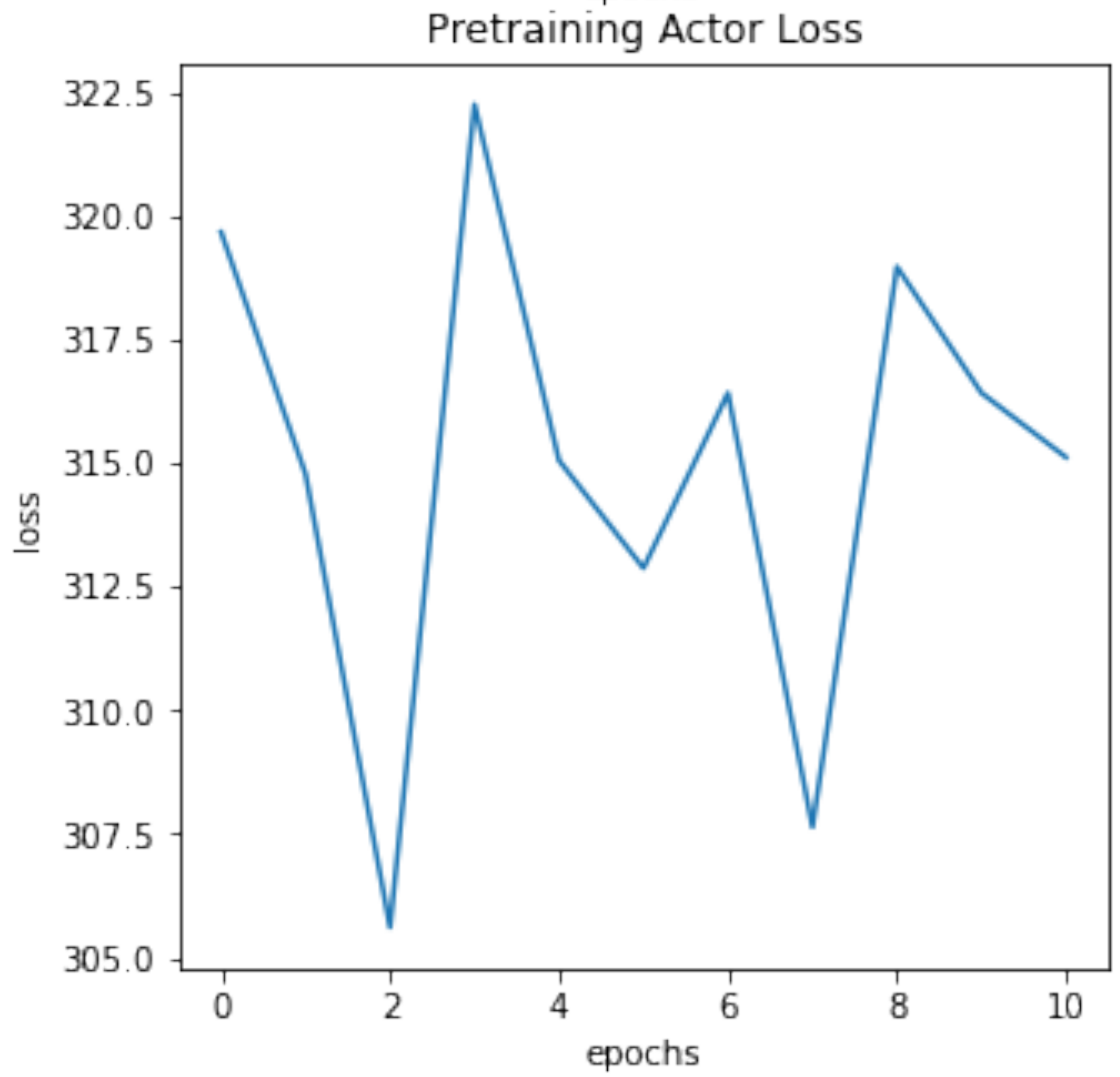


Reinforcement Learning

Actor Network : Architecture Limitations

- Proposed Actor Architecture has the following limitations:
    - The Network has no provisions for realising gait transition
    - The Network has no provisions for realising turning behaviour
    - The omega for the oscillator layer is sensitive to robot initial state, which should not be the case
    - Pre-training of the network fails to converges at a local minima instead of a global minima

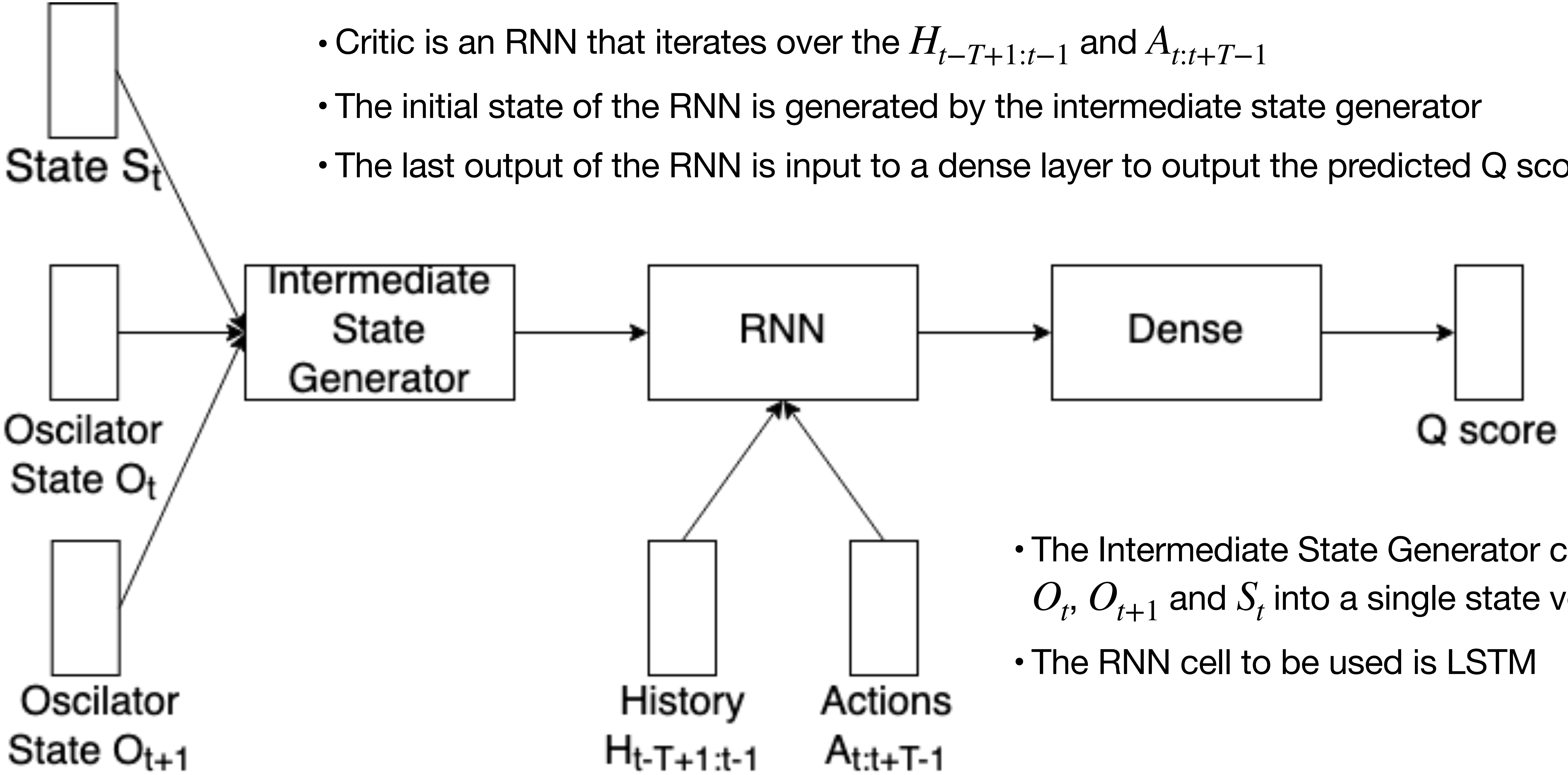




Reinforcement Learning

Critic Network

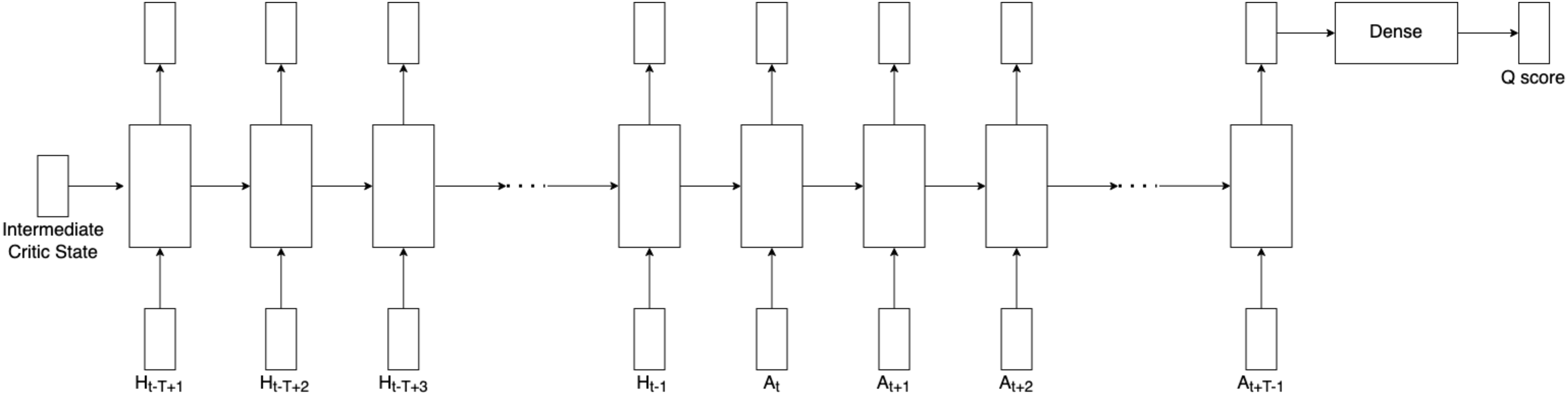
- Critic is an RNN that iterates over the $H_{t-T+1:t-1}$ and $A_{t:t+T-1}$
- The initial state of the RNN is generated by the intermediate state generator
- The last output of the RNN is input to a dense layer to output the predicted Q scores



- The Intermediate State Generator combines O_t, O_{t+1} and S_t into a single state vector
- The RNN cell to be used is LSTM

Reinforcement Learning

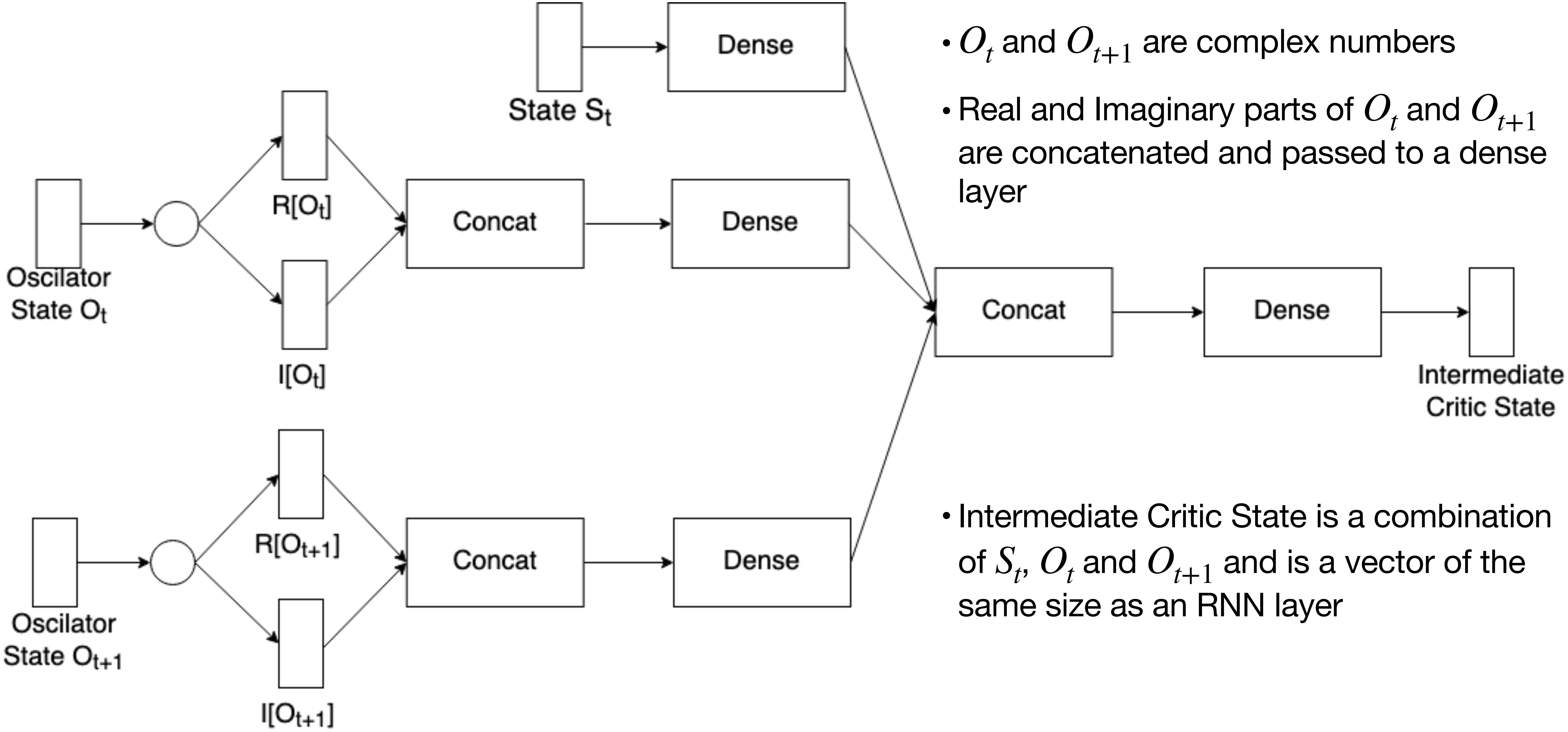
Critic Network : RNN



The Q Score predicted by the RNN takes into account the actions over the time window of $2T-1$ actions around a timestep t given S_t, O_t and O_{t+1}

Reinforcement Learning

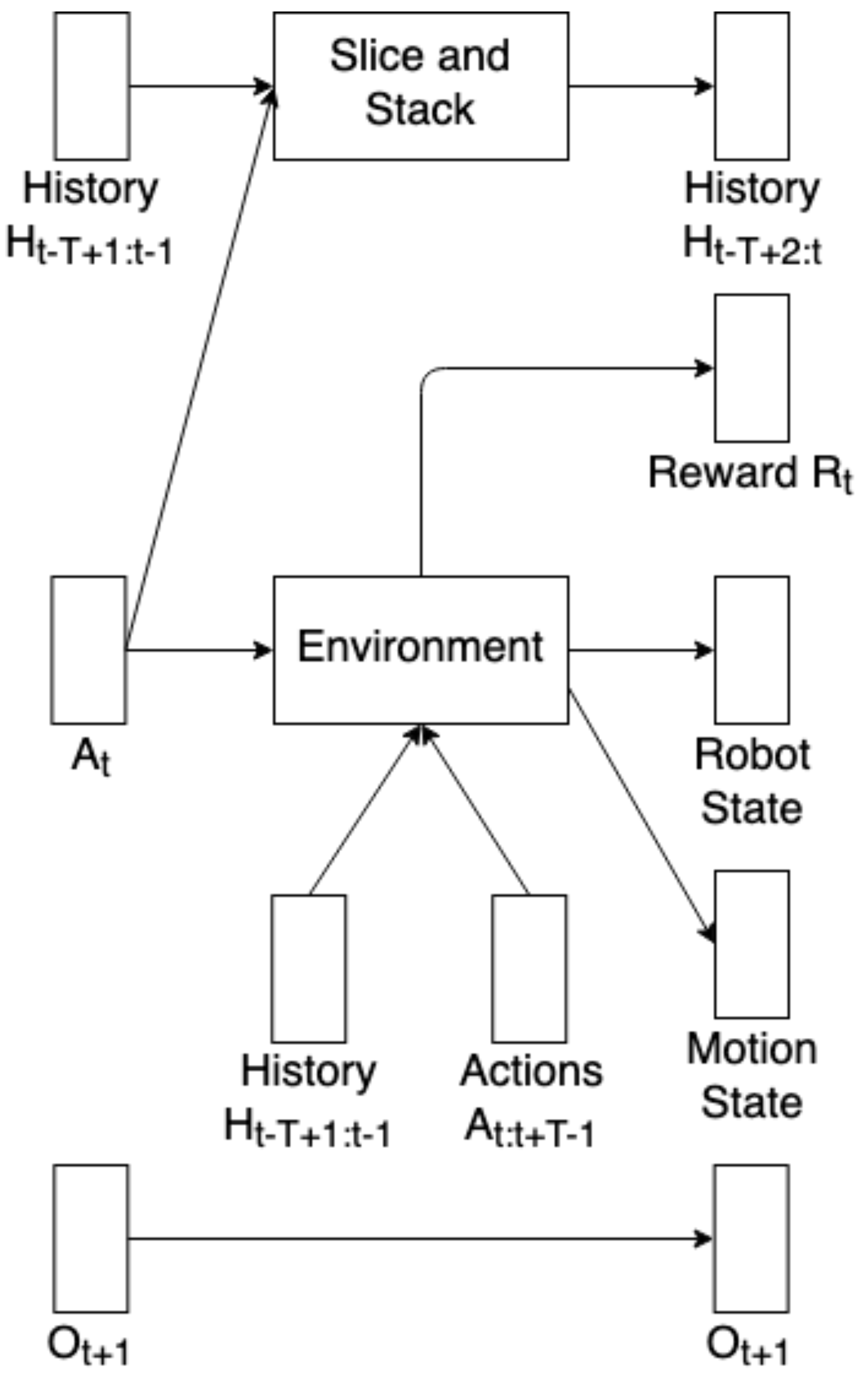
Critic Network : Intermediate State Generator



- O_t and O_{t+1} are complex numbers
- Real and Imaginary parts of O_t and O_{t+1} are concatenated and passed to a dense layer
- Intermediate Critic State is a combination of S_t, O_t and O_{t+1} and is a vector of the same size as an RNN layer

Reinforcement Learning
Environment

- History consists of the action from the previous $T-1$ timesteps
- History is initialised with $T-1$ repetitions of the initial joint positions
- The Environment produces the Robot State and Motion State given A_t
- Reward R_t is produced by the environment given History $H_{t-T+1:t-1}$ and Actions $A_{t:t+T-1}$
- The Oscillator State produced is passed on unchanged
- $H_{t-T+1:t-1}$ is sliced and stacked with A_t to get $H_{t-T+2:t}$ which is the history for the timestep $t+1$



Training Procedure

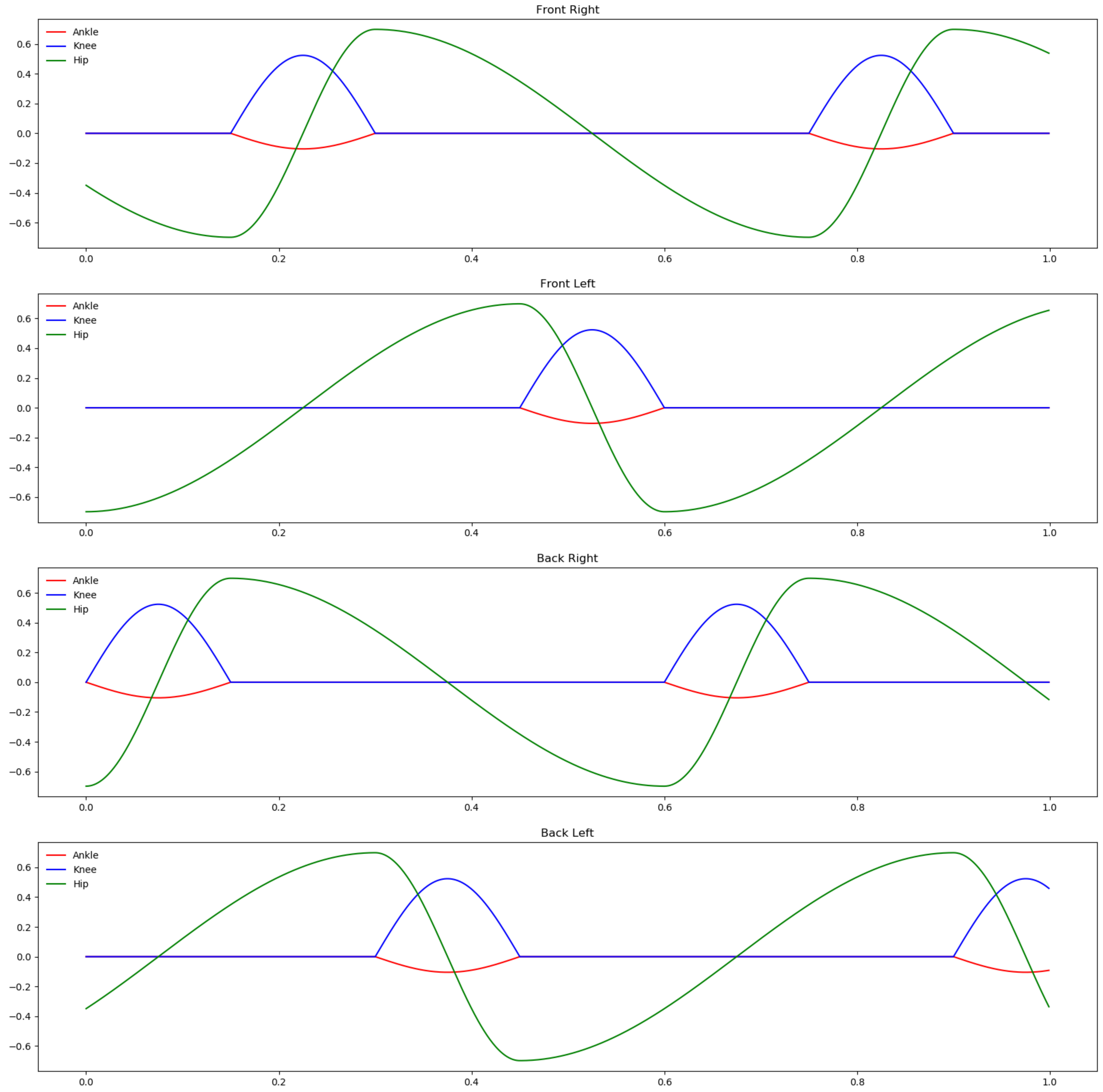
Multi Stepped Training Procedure

- Training of the gait generation network in two steps:
    - Pre-training of the Actor Network
    - Reinforcement Learning of DDPG
- Two stepped process to ensure quicker and smoother convergence to an optima during Reinforcement Learning
- The Pre-training steps provides information to the model about the process of gait, whereas Reinforcement Learning allows for fine-tuning of the knowledge from pre-training based on real world experience

Training Procedure

Multi Stepped Training

- The gait pattern used for pre-training is normalised
- Pre-training only using creep gait pattern
- Gait Transition and other gaits are expected to be learnt through Reinforcement Learning



Pre-training

Back-propagation Schemes

- Due to the optimisation of the neural network being a multi objective optimisation problem
- This leads to conflicts in updates causing the optimisation to not converge
- To alleviate this conflicting effect the following schemes of optimisations to be experimented with:
    - Entire Actor through signal MSE and Motion Encoder through ω and μ MSE both in the same step
    - Complex MLP through signal MSE and Motion Encoder through ω and μ MSE both in the same step
    - Motion Encoder through ω and μ MSE in step 1 followed by entire Actor through signal MSE and Motion Encoder through ω and μ MSE in step 2
    - Motion Encoder through ω and μ MSE in step 1 followed by entire Actor through signal MSE in step 2

References

1. Masuri, Ariel; Medina, Oded; Hacohen, Shlomi; Shvalb, Nir (2020) ‘Gait and Trajectory Optimization by Self-Learning for Quadrupedal Robots with an Active Back Joint’

2. Jia, Yan; Lua, Xiao; Han, Baoling; Liang, Guanhao; Zhao, Jiaheng; Zhao, Yuting (2018) ‘Stability Criterion for Dynamic Gaits of Quadruped Robot’

3. Kramer, Oliver; Gong, Daoxiong; Yan, Jie; Zuo, Guoyu (2010) ‘A Review of Gait Optimization Based on Evolutionary Computation’

4. Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver & Daan Wierstra (2016) ‘Continuous Control with Deep Reinforcement Learning’

5. Bose, Sourabh & Huber, Manfred. (2017). ‘Training Neural Networks with Policy Gradient’

6. S. Auddy, S. Magg & S. Wermter (2019) "Hierarchical Control for Bipedal Locomotion using Central Pattern Generators and Neural Networks”

7. Doo Re Song, Chuanyu Yang, Christopher McGreavy, Zhibin Li (2018) “Recurrent Deep Deteministic Policy Gradient Method for Bipedal Locomotion on Rough Terrain Challenge”

8. Deep Reinforcement Learning

References

9. Shouwen Fan, Min Sun (2007) “Gait Parameters Optimization and Real-Time Trajectory Planning for Humanoid Robots”