Structural Overview of the Quadruped Robot Report

Module A: Coordinate Systems and Leg Mechanics

1. Coordinate Systems Across Hierarchies
   - World Frame
   - Body Frame
   - Local Leg Frames
2. Leg Numbering Philosophy – XiLing Convention
3. Leg Mechanism and Motion Analysis
   - 3-Joint Leg Configuration
   - Forward and Inverse Kinematics Overview
   - Leg Diagram Interpretation

Module B: Forward and Inverse Kinematics for a Single Leg

1. Leg Architecture Breakdown
2. Movement Layer: Transformations and Joint Effects
3. Full-Chain Positional Computation
4. Analytical Form of Foot Position
5. Inverse Kinematics
   - Problem Context
   - Hip-Centered Frame Transformation
   - IK Decoupling Strategy
   - Geometry and Reachability
   - Derivations of q_1, q_2, q_3

Module C: Whole-Body Kinematics

1. Reference Frames
   - Types of Coordinate Systems
   - Homogeneous Transformation Matrix
   - World ↔ Body Conversion
2. Mapping Body to Leg Frames
3. Whole-Body Inverse Kinematics

- Local Frame Conversion
- Applying Single-Leg IK
- Full Forward Kinematics Chain

4. Body and Foot Trajectory Planning

- Pose and Foot Path Planning
- Stance vs. Swing Phases
- Interpolation Functions

Module D: Gait Formulation

1. Gait Fundamentals and Parameters

- Stride Length
- Cycle Time
- Duty Factor
- Gait Phasing

2. Trotting Gait Design

- Sequence and Phase Offsets
- Support Pattern

3. Foot Trajectory Design

- Stance Phase Trajectory
- Swing Phase with Elliptical Path
- Touchdown Planning

4. Stability Analysis

- Support Polygon
- Static Stability Margin (SSM)
- Dynamic Stability via ZMP
- Adaptive Step Timing

Module E: Trot Gait Implementation

1. Gait State Machine

2. Trot Gait Control Algorithm

- Initialization
- Cycle Update Loop
- Stability Monitoring

3. Summary of Key Stability Conditions and Controls

Module A:

1. Coordinate Systems Across Hierarchies

To interpret and control motion effectively, it's crucial to define the spatial frames used across the quadruped structure:

- **World (Global) Frame:** Anchored to the ground plane, this reference uses:
    - Forward (X), Leftward (Y), and Upward (Z) axes.
    - Ground truth variables marked with a **G** prefix.
- **Torso (Body) Frame:** Centered at the robot's geometric core:
    - X-axis leads the heading direction.
    - Y-axis extends to the robot's left.
    - Z-axis projects vertically upward.
    - Body variables use **B** prefix.
- **Local Leg Frames:** Each leg is assigned a coordinate system rooted at its hip joint:
    - **X-axis** points forward in rest pose.
    - **Y-axis** orientation depends on side (outward on left legs, inward on right).
    - **Z-axis** follows the right-hand rule (typically downward).

2. Leg Numbering Philosophy – XiLing Convention

Legs are indexed using a counter-clockwise sequence from the top view:

Leg ID	Label	Position
1	LF (Front-Left)	Top-left corner
2	LH (Rear-Left)	Bottom-left
3	RH (Rear-Right)	Bottom-right
4	RF (Front-Right)	Top-right corner

This numbering streamlines integration with common quadruped algorithms and published models like XiLing.

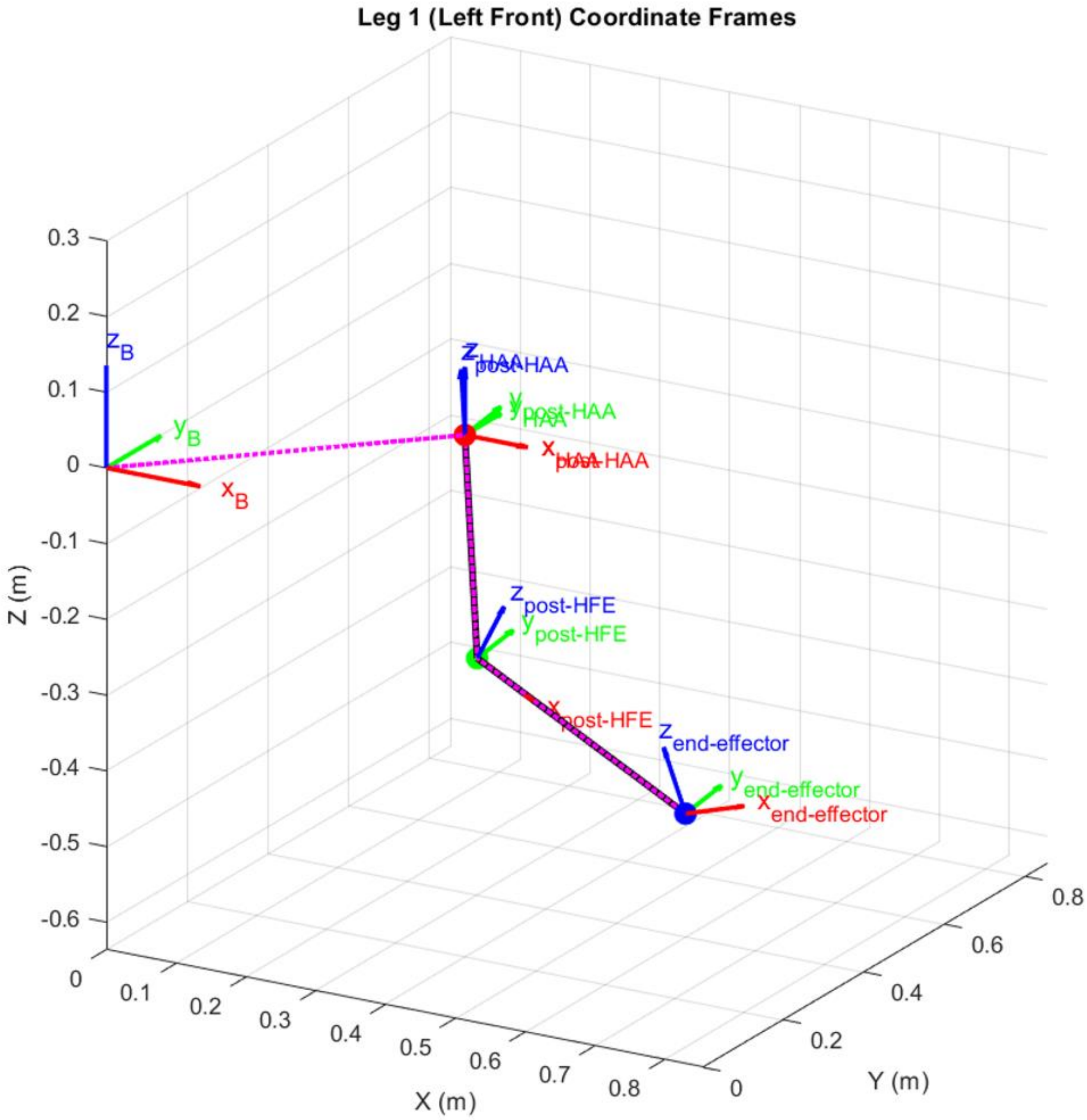
Leg Mechanism and Motion Analysis

This module addresses the core mechanical behavior of individual legs and how it contributes to global movement.

1. Mechanical Configuration of a 3-Joint Leg

Each limb comprises three degrees of freedom (DoF), achieved via revolute joints:

- **Joint 1: Hip Abduction/Adduction (HAA)**
    - Controls lateral sway.
    - Rotates about the **Y-axis** of the leg frame.
- **Joint 2: Hip Flexion/Extension (HFE)**
    - Moves the upper leg forward and backward.
    - Rotation follows the **Z-axis** after HAA.
- **Joint 3: Knee Flexion/Extension (KFE)**
    - Manages the extension of the lower leg.
    - Shares the axis alignment with HFE.



2. Kinematic Calculations

- **Forward Kinematics (FK):** Computes the foot position from known joint angles.

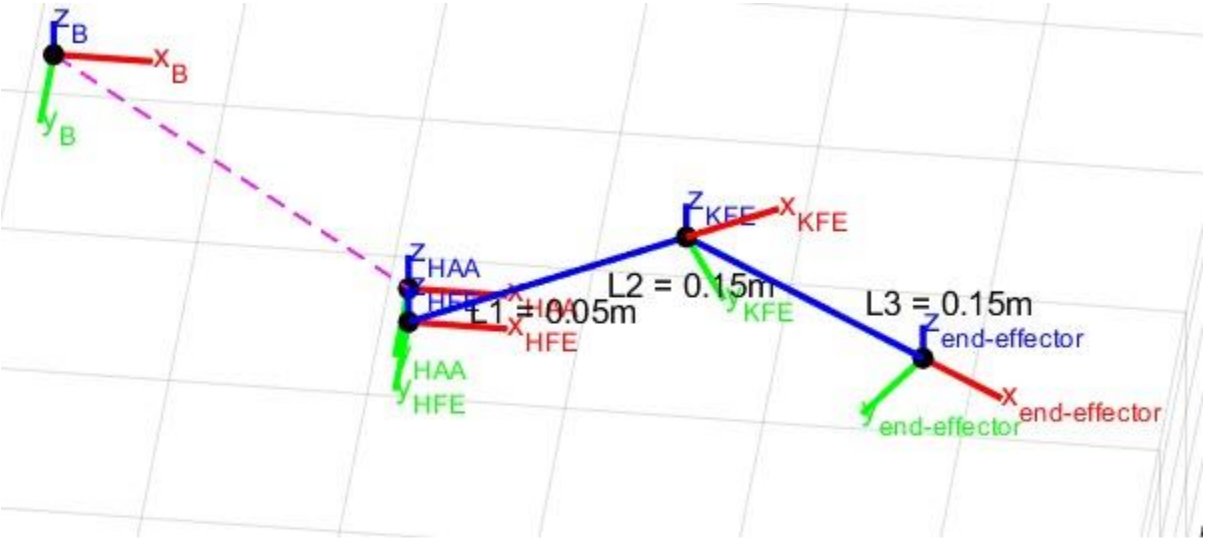
- **Inverse Kinematics (IK):** Determines required joint angles to achieve a desired foot location.

Both FK and IK are vital for walking, obstacle avoidance, and terrain adaptation.

3. Leg Diagram Interpretation (Leg_1)

Key geometric insights from the leg's visual representation:

- **Body Frame** is well-aligned at the torso center.
- **HAA Joint Frame** has the proper lateral offset and orientation.
- **HFE Frame** lies below HAA and supports natural forward-backward motion.
- **Foot (End-Effector)** frame is located at ground contact with maintained leg-relative orientation.



Module B:

Understanding of Forward Kinematics for a Single Quadruped Leg

In robotic leg design, computing how each joint contributes to the foot's final position in 3D space is a foundational step. Instead of marching through transformations linearly, this revised layout groups ideas based on function: **structural anatomy**, **movement influence**, and **positional outcomes**.

---

1. Leg Architecture Breakdown – Understanding What Moves What

To build a kinematic model, we first need to understand the **physical layout** of the leg and the **degrees of motion** it allows.

Components of the Leg (Leg 1 - Left Front)

- **l_1**: Segment from hip abduction/adduction (HAA) to hip flexion/extension (HFE) — *hip link*
- **l_2**: From HFE to knee joint (KFE) — *thigh link*
- **l_3**: From knee to foot — *shank link*

Rotational Joints

Each joint introduces a specific rotation:

Joint	Notation	Axis	Function
HAA	q_1	Y	Outward/inward swing
HFE	q_2	Z	Forward-backward leg motion
KFE	q_3	Z	Extends or folds the lower leg

The hip is anchored in the body frame at an offset:

$$(x_{hip},\ y_{hip},\ z_{hip})$$

---

2. Movement Layer: Translating Rotations to Transformations

To capture how joint rotations shift the leg in space, we chain transformation matrices. Think of each transformation as a "movement operation" from one segment to the next.

Transformation Blocks (Grouped by Action Type)

1. **Positioning the Hip Joint (Fixed Offset)**
   Move from body center to hip:

$$T_{B \to HAA} = \text{Translation}(x_{hip}, y_{hip}, z_{hip})$$

2. **HAA Rotation (q_1)**
   Rotate around **Y-axis** of leg frame.
3. **Descend to HFE Joint**
   Translate along Z by $-l_1$.
4. **HFE Rotation (q_2)**
   Rotate around Z-axis after HAA.
5. **Reach Knee Joint**
   Translate forward along X by l_2.
6. **KFE Rotation (q_3)**
   Rotate again about the Z-axis.

7. **Final Stretch to Foot**
   Translate by l_3 along X to reach the end-effector (foot).

Each of these steps is represented by a **4×4 transformation matrix**, and their cumulative multiplication gives the final pose of the foot.

---

3. Full-Chain Positional Computation

The total transformation from the body center to the foot is:

$$T_{B \to FOOT} = T_{B \to HAA} \cdot T_{HAA\_ROT} \cdot T_{HAA \to HFE} \cdot T_{HFE\_ROT} \cdot T_{HFE \to KFE} \cdot T_{KFE\_ROT} \cdot T_{KFE \to FOOT}$$

Matrix Chain Computation Simplified (Modular)

Step 1: Combine First Segment

$$T_{B \to HFE} = T_{B \to HAA} \cdot T_{HAA\_ROT} \cdot T_{HAA \to HFE}$$

Step 2: Expand to Knee Joint

$$T_{B \to KFE} = T_{B \to HFE} \cdot T_{HFE\_ROT} \cdot T_{HFE \to KFE}$$

Step 3: Final Transformation

$$T_{B \to FOOT} = T_{B \to KFE} \cdot T_{KFE\_ROT} \cdot T_{KFE \to FOOT}$$

---

4. Final Result – Analytical Form of Foot Position

By multiplying all transformations and simplifying, we derive the foot coordinates relative to the body:

$$x_{foot} = x_{hip} - l_1\sin(q_1) + l_2\cos(q_1)\cos(q_2) + l_3\cos(q_1)\cos(q_2 + q_3)$$

$$y_{foot} = y_{hip} + l_2\sin(q_2) + l_3\sin(q_2 + q_3)$$

$$z_{foot} = z_{hip} - l_1\cos(q_1) - l_2\sin(q_1)\cos(q_2) - l_3\sin(q_1)\cos(q_2 + q_3)$$

This form gives you **direct spatial mapping** from joint angles to the foot location, which is the essence of forward kinematics.

Inverse Kinematics for a Quadruped Robot Leg

Module 1: Problem Context and Given Information

Objective:
Given a desired 3D foot position (x_foot, y_foot, z_foot), calculate the joint angles q1, q2, q3 for a single leg of the quadruped.

Known Parameters:
- Position of the hip joint in body frame: (x_hip, y_hip, z_hip)
- Segment lengths:
  l1: hip to thigh
  l2: thigh to knee
  l3: knee to foot

Pre-derived Forward Kinematics:
These equations express foot position as a function of joint angles:
x_foot = x_hip - l1*sin(q1) + l2*cos(q1)*cos(q2) + l3*cos(q1)*cos(q2 + q3)
y_foot = y_hip + l2*sin(q2) + l3*sin(q2 + q3)
z_foot = z_hip - l1*cos(q1) - l2*sin(q1)*cos(q2) - l3*sin(q1)*cos(q2 + q3)

Module 2: Shifting to a Hip-Centered Frame

To simplify calculations, we define the foot position relative to the hip origin:
x' = x_foot - x_hip
y' = y_foot - y_hip
z' = z_foot - z_hip

Module 3: Decoupling the IK Problem

Inverse kinematics for this leg is decomposed into two sub-problems:
1. Solve for q1: The angle controlling the leg's lateral swing (abduction/adduction).

2. Solve for q2 and q3: The planar bending of the leg in the sagittal (XZ) plane.

Module 4: Determining Reach and Geometry

We introduce geometric simplifications to understand reachability:
- r = sqrt(x'^2 + z'^2)
- d = sqrt(x'^2 + y'^2 + z'^2)
- Valid workspace condition: |l2 - l3| < d < l2 + l3

Module 5: Deriving q1 (Hip Abduction/Adduction)

Define ρ = l2*cos(q2) + l3*cos(q2 + q3)

Forward kinematics from hip frame:
x' = -l1*sin(q1) + ρ*cos(q1)
z' = -l1*cos(q1) - ρ*sin(q1)

Combine:
x'*sin(q1) + z'*cos(q1) = -l1

Module 6: Solving for q1

Define angle ϕ = atan2(z', x')

Then:
x'*sin(q1) + z'*cos(q1) = sqrt(x'^2 + z'^2) * sin(q1 + ϕ)
Solving:
q1 = arcsin(-l1 / sqrt(x'^2 + z'^2)) - ϕ

Alternate numerically stable form:
q1 = atan2(-x'*l1 - z'*sqrt(r^2 - l1^2), -z'*l1 + x'*sqrt(r^2 - l1^2))
Where r = sqrt(x'^2 + z'^2)

Solving for q3: Knee Joint Flexion/Extension

Using the leg plane formed by l2 and l3:

Effective leg distance:
s = sqrt(ρ^2 + y'^2)
ρ = x'*cos(q1) - z'*sin(q1)

Law of Cosines:
s^2 = l2^2 + l3^2 + 2*l2*l3*cos(q3)
q3 = arccos((s^2 - l2^2 - l3^2) / (2*l2*l3))

Explicit:

q3 = arccos(((x'*cos(q1) - z'*sin(q1))^2 + y'^2 - l2^2 - l3^2) / (2*l2*l3))

Solving for q2: Hip Flexion/Extension

Break into two angles:
α = atan2(y', ρ) with ρ = x'*cos(q1) - z'*sin(q1)
β = arcsin((l3*sin(q3)) / s) where s = sqrt((x'*cos(q1) - z'*sin(q1))^2 + y'^2)

Final:
q2 = α - β

Explicit:
q2 = atan2(y', x'*cos(q1) - z'*sin(q1)) - arcsin((l3*sin(q3)) / sqrt((x'*cos(q1) - z'*sin(q1))^2 + y'^2))

Inverse Kinematics Equations

Given relative foot position (x', y', z'):

$q_1 = \arcsin(-l_1/r) - \text{atan2}(z', x')$ where $r = \sqrt{(x'^2 + z'^2)}$

$q_3 = \arccos(((x'\cos(q_1) - z'\sin(q_1))^2 + y'^2 - l_2^2 - l_3^2)/(2l_2l_3))$

$q_2 = \text{atan2}(y', x'\cos(q_1) - z'\sin(q_1)) - \arcsin((l_3\sin(q_3))/\sqrt{((x'\cos(q_1) - z'\sin(q_1))^2 + y'^2)}$

Module C:

Whole-Body Kinematics for Quadruped Robots: Modular and Analytical Overview

Section 1: Reference Frames and Their Definitions

1.1 Types of Coordinate Systems

In the kinematic model of quadruped robots, three major coordinate systems are considered:

- **World Frame (W)**
  - Fixed origin in the environment
  - X-axis: forward
  - Y-axis: left
  - Z-axis: up

- **Body Frame (B)**
  - Origin at the center of the robot's body
  - X-axis: robot forward
  - Y-axis: left
  - Z-axis: upward

- **Leg Frame (Li)** for each leg $i \in \{1, 2, 3, 4\}$
  - Origin at the hip joint
  - Alignment follows the body frame when in the neutral pose

1.2 Homogeneous Transformation Matrix

A 4x4 transformation matrix T encodes both rotation and translation from one frame to another:

$$T = \begin{bmatrix} R_{W,B} & p_{W,B} \\ 0 & 1 \end{bmatrix}$$

Where:
- $R_{W,B}$ is a 3x3 rotation matrix from body to world frame.
- $p_{W,B}$ is a 3x1 vector locating the body in the world frame.

1.3 Representation of Body Pose

Position Vector:
$$p_{W,B} = [x_B, y_B, z_B]^T$$

Orientation using Euler angles:
$$R_{W,B} = R_z(\psi) * R_y(\theta) * R_x(\phi)$$

Where:
- $R_x(\phi)$: Rotation around X-axis
- $R_y(\theta)$: Rotation around Y-axis
- $R_z(\psi)$: Rotation around Z-axis

1.4 Conversions Between World and Body Frames

Forward (Body → World):
$$T_{W,B} = \begin{bmatrix} R_{W,B} & p_{W,B} \\ 0 & 1 \end{bmatrix}$$

Inverse (World → Body):
$$T_{B,W} = \begin{bmatrix} R_{W,B}^T & -R_{W,B}^T * p_{W,B} \\ 0 & 1 \end{bmatrix}$$

Where $R_{W,B}^T$ is the transpose of the rotation matrix.

1.5 Mapping Body to Individual Legs

Each leg i has transformation from body to leg frame Li:

$$T_{B,Li} = \begin{bmatrix} R_{B,Li} & p_{B,Li} \\ 0 & 1 \end{bmatrix}$$

- $R_{B,Li}$: rotation from body to leg (often identity)
- $p_{B,Li}$: position of leg i's hip joint in body frame

Leg attachment points (for a typical rectangular robot body):

- Left Front (L1): $[\ l_b/2,\ w_b/2,\ 0\ ]$
- Left Hind (L2): $[-l_b/2,\ w_b/2,\ 0\ ]$
- Right Hind (L3): $[-l_b/2,\ -w_b/2,\ 0\ ]$
- Right Front(L4): $[\ l_b/2,\ -w_b/2,\ 0\ ]$

Section 2: Whole-Body Inverse Kinematics

The purpose of whole-body IK is to compute joint angles for each leg based on:
- Desired foot locations in the world frame
- Current pose of the body (position + orientation)

2.1 Transforming Foot Positions to Local Frames

To find foot position in leg's local frame Li:

$$p_{Li,Fi} = R_{B,Li}^T * [\ R_{W,B}^T * (p_{W,Fi} - p_{W,B}) - p_{B,Li}\ ]$$

Where:
- $p_{W,Fi}$ is the foot position in world frame
- $p_{W,B}$ is body center in world frame
- $p_{B,Li}$ is position of hip joint in body frame
- $R_{B,Li}^T$ converts from body to leg frame

2.2 Applying Single-Leg Inverse Kinematics

Once $p_{Li,Fi}$ is computed, we apply inverse kinematics for each leg i:

$$q_{1i} = \arcsin(-l1 / r_i) - \operatorname{atan2}(z'_i, x'_i)$$

$$q_{3i} = \arccos(((x'_i * \cos(q_{1i}) - z'_i * \sin(q_{1i}))^2 + y'_i{}^2 - l2^2 - l3^2) / (2 * l2 * l3))$$

$$q_{2i} = \operatorname{atan2}(y'_i, x'_i * \cos(q_{1i}) - z'_i * \sin(q_{1i})) - \arcsin((l3 * \sin(q_{3i})) / \operatorname{sqrt}((x'_i * \cos(q_{1i}) - z'_i * \sin(q_{1i}))^2 + y'_i{}^2))$$

Where $r_i = \operatorname{sqrt}(x'^2_i + z'^2_i)$ and (x'_i, y'_i, z'_i) are foot positions in local leg frame Li

2.2 Full Forward Kinematics Chain

To find the position of a foot in the world frame, we apply a chain of coordinate transformations as follows:

$$p_{W,Fi} = T_{W,B} \cdot T_{B,Li} \cdot [p_{Li,Fi}; 1]$$

Where:
- $p_{W,Fi}$ = Foot position in the world frame
- $T_{W,B}$ = Transformation from body frame to world frame
- $T_{B,Li}$ = Transformation from leg frame to body frame
- $p_{Li,Fi}$ = Foot position in local leg frame

Step-by-step derivation:

1. Leg Frame → Body Frame:
- The local foot position is first rotated into body frame using $R_{B,Li}$
- Then translated using the leg offset $p_{B,Li}$
- Result: $p_{B,Fi} = R_{B,Li} \cdot p_{Li,Fi} + p_{B,Li}$

2. Body Frame → World Frame:
- This result is then rotated to the world frame using $R_{W,B}$
- And translated using body position $p_{W,B}$
- Final form: $p_{W,Fi} = R_{W,B} \cdot p_{B,Fi} + p_{W,B}$

Combining both transformations:

$$
\begin{aligned}
p_{W,Fi} &= R_{W,B} \cdot (R_{B,Li} \cdot p_{Li,Fi} + p_{B,Li}) + p_{W,B} \\
&= R_{W,B} \cdot R_{B,Li} \cdot p_{Li,Fi} + R_{W,B} \cdot p_{B,Li} + p_{W,B}
\end{aligned}
$$

This breakdown clearly shows the impact of both the body's pose and leg placement on the world-space position of each foot.

---

3. Whole-Body Inverse Kinematics

Whole-body IK solves for joint angles given:

- Desired foot positions in world frame
- Current pose of robot's body in world frame

3.1 Transformation of Foot Positions

We transform desired foot positions to each leg's local frame for inverse kinematics:

$$p_{Li,Fi} = R_{B,Li}^T \cdot (R_{W,B}^T \cdot (p_{W,Fi} - p_{W,B}) - p_{B,Li})$$

Where:
- $p_{W,Fi}$ = Foot position in world frame
- $p_{W,B}$ = Body center in world frame
- $p_{B,Li}$ = Position of leg i's hip joint in body frame
- $R_{W,B}$ = Rotation from body to world frame
- $R_{B,Li}$ = Rotation from leg to body frame (typically identity)

3.2 Single Leg Inverse Kinematics

With the foot in local frame, use inverse kinematics equations:

$$q_{1i} = \arcsin(-l1 / r_i) - \operatorname{atan2}(z'_i, x'_i)$$

$$q_{3i} = \arccos(((x'_i \cdot \cos(q_{1i}) - z'_i \cdot \sin(q_{1i}))^2 + y'_i^2 - l2^2 - l3^2) / (2 \cdot l2 \cdot l3))$$

$q_{2i} = \mathrm{atan2}(y'_i, x'_i \cdot \cos(q_{1i}) - z'_i \cdot \sin(q_{1i})) - \arcsin((l3 \cdot \sin(q_{3i})) / \sqrt{(x'_i \cdot \cos(q_{1i}) - z'_i \cdot \sin(q_{1i}))^2 + {y'_i}^2})$

Where:
- (x'_i, y'_i, z'_i) are foot coordinates in leg frame
- $r_i = \sqrt{{x'_i}^2 + {z'_i}^2}$

---

4. Body Pose Control

To move a quadruped robot along a specified path, we must:
1. Define how the body should move (trajectory of the body)
2. Ensure leg movements maintain stability

This section introduces how to plan a body trajectory and coordinate foot movements to maintain balance.

4.1 Body Trajectory Representation

The robot's body pose in the world is represented by its position and orientation over time.

Position:
$p_{W,B}(t) = [x_B(t), y_B(t), z_B(t)]^T$

Orientation:
$\Phi(t) = [\phi(t), \theta(t), \psi(t)] \rightarrow$ roll, pitch, yaw

These variables define the spatial posture of the robot's torso over time.

We can also represent this motion with a homogeneous transformation matrix:

$$T_{W,B}(t) = \begin{bmatrix} R_{W,B}(t) & p_{W,B}(t) \\ 0 & 1 \end{bmatrix}$$

Where $R_{W,B}(t)$ is a time-varying rotation matrix representing body orientation.

4.2 Foot Trajectory Generation

Each foot must track a trajectory relative to the body. Its movement alternates between two phases:

- **Stance Phase**: foot remains on the ground (supports body)
- **Swing Phase:** foot is lifted and moves to next position

Stance Phase – Foot is Fixed on Ground (in world frame):

$p_{W,Fi}(t) = p_{W,Fi}(t_{contact})$

In body frame (because body moves while foot is planted):

$p_{B,Fi}(t) = R_{W,B}(t)^T \cdot (p_{W,Fi}(t_{contact}) - p_{W,B}(t))$

Swing Phase – Foot Moves to New Location (in body frame):

$p_{B,Fi}(t) = p_{B,Fi}(t_{liftoff}) + s(t) \cdot (p_{B,Fi}(t_{touchdown}) - p_{B,Fi}(t_{liftoff})) + h(t) \cdot [0, 0, h_{max}]$

Where:
- $s(t)$ interpolates horizontal movement ($0 \to 1$)
- $h(t)$ is vertical lift (sinusoidal or parabolic)
- h_{max} is max swing height

This guarantees a smooth foot path that avoids collision and supports balance.

Understanding the Motion Functions

1. $s(t)$ – Horizontal motion scaling

   - Determines horizontal foot progress from lift to touch.

   - Typically follows trapezoidal velocity profile.

2. $h(t)$ – Vertical motion

   - Controls height profile of foot during swing.

   - Often sinusoidal: $h(t) = h_{max} \cdot \sin(\pi t)$

   - Ensures lift and smooth landing

Module D:

Gait Formulation for Quadruped Robots

1. Gait Fundamentals and Parameters

1.1 Key Gait Parameters

Stride Length (S): The stride length (S) is the distance covered in one complete locomotion cycle and is given by:

$S = v_d * T_c$ (Equation 28)

Where:

- S is stride length (meters)
- v_d is desired forward velocity (meters/second)
- T_c is cycle time (seconds)

Cycle Time (T_c): Cycle time represents the duration of one complete gait cycle:

$T_c = n_s * T_s$ (Equation 29)

Where:
- n_s is the number of steps in a cycle (typically 4 for a walking gait, 2 for a trotting gait)
- T_s is the duration of a single step (seconds)

Duty Factor: The duty factor (β) represents the fraction of the cycle time a foot remains in contact with the ground:

$\beta = T_{stance} / T_c$ (Equation 30)

Where:
• β is the duty factor (dimensionless, between 0 and 1)
• T_{stance} is the stance phase duration (seconds)

Gait Phasing: Each leg has a phase offset (ϕi) that defines its timing relative to the overall gait cycle:

$\phi_i = t_i / T_c$ (Equation 31)

Where:
• ϕ_i is the phase of leg i (dimensionless, between 0 and 1)
• t_i is the time offset for leg i (seconds)

---

2. Trotting Gait Pattern Design

2.1 Trot Gait Sequence

• Pair 1: Left Front (LF) & Right Hind (RH)
• Pair 2: Right Front (RF) & Left Hind (LH)

2.2 Phase Relationships

For a trot gait, the phase offsets are:

$\phi_{LF} = 0$ (Equation 32)
$\phi_{RH} = 0$ (Equation 33)
$\phi_{RF} = 0.5$ (Equation 34)
$\phi_{LH} = 0.5$ (Equation 35)

This means that one diagonal pair is always half a cycle (180°) out of phase with the other, ensuring continuous support.

2.3 Duty Factor Selection

For a trot gait: $0.3 \le \beta \le 0.5$ (Equation 43)

Where:
• $\beta = 0.5$ produces a walk-like trot (more stable, slower)
• $\beta = 0.3$ produces a run-like trot (more dynamic, faster)

2.4 Support Pattern

The support pattern matrix S(t) indicates which legs are on the ground at time t:

$S(t) = [S_{LF}(t), S_{LH}(t), S_{RH}(t), S_{RF}(t)]$ (Equation 36)

Where for each leg i:
• $S_i(t) = 1$ if leg i is in stance phase at time t (on the ground)
• $S_i(t) = 0$ if leg i is in swing phase at time t (off the ground)

For duty factor $\beta=0.5$, the support pattern is:

• First half of cycle ($t \in [0, 0.5 \cdot Tc]$):
$S(t) = [1, 0, 1, 0]$ (Equation 37)
(LF and RH on the ground, RF and LH in the air)

• Second half of cycle ($t \in [0.5 \cdot Tc, Tc]$):
$S(t) = [0, 1, 0, 1]$ (Equation 38)
(RF and LH on the ground, LF and RH in the air)

---

3. Foot Trajectory Generation

Each foot follows a carefully planned trajectory to ensure smooth and stable locomotion.

3.1 Stance Phase Trajectory

During the stance phase, the foot remains on the ground while the body moves forward. The foot trajectory is given by:

$p_{B,Fi}^{stance}(t) = [x_0 - (t - t_0) \cdot v_d, y_0, 0]$ (Equation 39)

Where:

- x_0, y_0 are the initial foot coordinates in the body frame

- t_0 is the time at stance phase start

- v_d is the desired forward velocity

3.2 Swing Phase Trajectory

During swing, the foot lifts off, moves forward, and lands smoothly. A half-elliptical trajectory is commonly used:

$p_{B,Fi}^{swing}(t) = [x_s + (x_e - x_s) \cdot s(t), y_0, h_{max} \cdot \sin(\pi \cdot s(t))]$ (Equation 40)

Where:

- x_s is the swing start x-position

- x_e is the swing end x-position

- $s(t)$ is a normalized timing parameter from 0 to 1

- h_{max} is the maximum foot height during swing

$s(t) = (t - t_s) / (t_e - t_s)$ (Equation 41)

3.3 Foot Placement Planning

The touchdown position of each foot affects stability. A simple rule is:

$x_{TD,i} = k_p \cdot (x_{COM} + v_x \cdot k_v) + l_i$ (Equation 42)

$y_{TD,i} = k_p \cdot y_{COM} + w_i$ (Equation 43)

Where:

- $x_{TD,i}$, $y_{TD,i}$ are touchdown coordinates for foot i
- x_{COM}, y_{COM} are the current COM positions
- v_x is current body velocity
- k_p is position feedback gain
- k_v is velocity feedback gain
- l_i, w_i are nominal foot positions relative to the body center

---

Stability and Gait Control in Quadruped Robots

1. Support Polygon

At any given moment, the region on the ground defined by the convex hull of all feet currently touching the ground is referred to as the support polygon. This polygon is computed using the positions of those feet that are in contact with the surface, ensuring a stable base for the robot.

Mathematically, it can be represented as:
$P(t) = \text{ConvexHull}(\{p_{W,Fi,xy}(t) \mid S_i(t) = 1\})$

- $p_{W,Fi,xy}(t)$: Position of the i-th foot in world coordinates.
- $S_i(t)$: An indicator variable equal to 1 if the i-th foot is in contact with the ground; 0 otherwise.
- Convex hull includes all grounded feet to form the support polygon.

2. Static Stability Margin (SSM)

Static Stability Margin is a metric used to evaluate how close the robot's center of mass (COM) is to the edge of the support polygon. It is computed as the minimum perpendicular distance from the COM's ground projection to any edge of the support polygon:

$SSM(t) = \min\{d(p_{W,COM,xy}(t), e_j) \mid e_j \in \text{Edges}(P(t))\}$

- $d(p, e)$: Perpendicular distance from a point to an edge.
- $p_{W,COM,xy}(t)$: Ground projection of the COM.
- e_j: Edge of the support polygon.

For stability to be ensured, the margin must always be positive:
$SSM(t) > 0 \ \forall t$

3. Dynamic Stability - Zero Moment Point (ZMP)

To maintain dynamic stability during motion, the Zero Moment Point (ZMP) must lie within the boundaries of the support polygon. The ZMP is calculated as:

$p_{ZMP}(t) = p_{COM,xy}(t) - h_{COM} * a_{COM,xy}(t) / g$

- h_{COM}: Height of COM from the ground.
- $a_{COM,xy}(t)$: Horizontal acceleration of COM.
- g: Acceleration due to gravity (9.81 m/s^2).

To ensure dynamic stability:
$p_{ZMP}(t) \in P(t)\ \forall\ t$

4. Gait Timing Adjustment

To support balance during walking, the step duration can be modified dynamically based on the stability margin. The adjusted step timing is computed using:

$$T_s(t) = T_{s,nominal} * [1 + k_s * (SSM_{nominal} - SSM(t))]$$

- $T_s(t)$: Adjusted step time.
- $T_{s,nominal}$: Base step duration.
- k_s: Adaptation factor (typically 0.1 to 0.5).
- $SSM_{nominal}$: Desired stability margin.

When the margin is low, step time increases; when margin is high, step time decreases to maintain dynamic balance.

Module E:

5. Trot Gait Implementation

5.1 Gait State Machine

To control the motion of each leg during trotting, a state machine is used with these main phases:
- Initial: All legs are grounded.
- Lift-off: The leg transitions to swing.
- Swing: The leg moves to its next ground contact point.
- Touchdown: The leg lands on the designated location.
- Stance: The leg maintains ground contact while body moves forward.

5.2 Trot Gait Algorithm

The trot gait algorithm involves initialization and continuous cycle updates:
1. Initialization:
   - Define gait parameters and phase offsets.
   - Mark all legs in stance mode.
2. Per cycle update:
   - Update time and compute normalized phase.
   - For each leg:
     - Identify current phase (swing/stance).
     - Calculate transitions and respective positions.
     - Use inverse kinematics to compute joint angles.
3. Stability Monitoring:
   - Update support polygon and stability margin.
   - Adjust gait parameters as needed.

Summary

- COM must remain inside the support polygon to ensure static stability.
- ZMP staying within the support polygon guarantees dynamic stability.
- Step timing adapts based on how close the robot is to losing stability.
- Trot gait relies on structured state machines and phase management.
- Inverse kinematics enables precise leg control for terrain adaptability.