

## Phase 5.2: Gait Tuning and Optimization

การปรับจูนและพัฒนาหลายโหมดการเดิน

BLEGS Analysis Unit

นายธีรโชติ เมืองจันทน์

BLEGS Quadruped Robot Project

2 มกราคม 2026

## สารบัญ

# 1 บทนำ (Introduction)

## 1.1 วัตถุประสงค์

Phase 5.2 มีวัตถุประสงค์เพื่อปรับปรุงพารามิเตอร์การเดินของหุ่นยนต์สี่ขา พัฒนาหลายโหมดการเดิน และเพิ่มประสิทธิภาพการเดินให้นุ่มนวลและเสถียยิ่งขึ้น โดยเฉพาะการพัฒนา Asymmetric Trajectory Generation เพื่อลดแรงกระแทกและเพิ่มความนุ่มนวล

## 1.2 ขอบเขตการศึกษา

- การวิเคราะห์ปัญหาจากการทดสอบ Phase 5.1 (Compromised posture)
- การพัฒนา Asymmetric Trajectory Generation (Stance/Swing phase)
- การออกแบบโหมดการเดินหลายแบบ (6 modes)
- การทดสอบ Forward และ Backward trot
- การวิเคราะห์ผลกระทบของ Foot orientation
- การเปรียบเทียบประสิทธิภาพของแต่ละโหมด

## 1.3 ความเชื่อมโยงกับ Phase ก่อนหน้า

Phase 5.2 ปรับปรุงผลการทดสอบจาก Phase 5.1:

- แก้ปัญหา Compromised posture ด้วยการปรับ Gait parameters
- เพิ่มความนุ่มนวลด้วย Asymmetric trajectory
- พัฒนาหลายโหมดการเดินเพื่อความยืดหยุ่น

# 2 ปัญหาจาก Phase 5.1

## 2.1 ผลการทดสอบเริ่มต้น

จากการทดสอบ Phase 5.1 (29 ธันวาคม 2025) พบว่า:

พารามิเตอร์	ค่าที่ใช้	ผลลัพธ์
Step Length	30 mm	น้อยเกินไป
Lift Height	15 mm	ต่ำเกินไป
Cycle Time	600 ms	เหมาะสม
Update Rate	50 Hz	เหมาะสม
Posture	Compromised (ท่าประนีประนอม)	
Speed	ช้า ( 50 mm/s)	
Smoothness	ยอมรับได้ แต่ยังไม่นุ่มนวลมาก	

ตารางที่ 1: สรุปผลการทดสอบ Phase 5.1

## 2.2 การวิเคราะห์สาเหตุ

### 2.2.1 Compromised Posture

ปัญหา Compromised Posture หรือท่าทางประนีประนอม เกิดจากหลายปัจจัยดังนี้:

1. **Step length ต่ำ (30 mm):** ระยะก้าวที่สั้นเกินไปทำให้ขาของหุ่นยนต์ไม่สามารถเหยียดออกได้เต็มที่ ส่งผลให้ท่าทางโดยรวมมีลักษณะห่อตัวและเกร็ง ซึ่งไม่เหมาะสมสำหรับการเดินระยะไกลหรือการทรงตัวที่ดี
2. **Lift height ต่ำ (15 mm):** ระยะยกเท้าที่ต่ำเกินไปทำให้เท้ามีความเสี่ยงที่จะกระทบพื้นในช่วง Swing phase โดยเฉพาะเมื่อพื้นไม่เรียบหรือมีสิ่งกีดขวางเล็กน้อย นอกจากนี้ยังทำให้การเคลื่อนที่ดูไม่เป็นธรรมชาติ
3. **Center position ไม่เหมาะสม:** ตำแหน่งศูนย์กลางของ Trajectory อาจไม่ตรงกับตำแหน่งที่เหมาะสมสำหรับการเดินจริง ทำให้เกิดการกระเจายน้าหนักที่ไม่สมดุลระหว่างขาทั้งสอง

### 2.2.2 Lack of Smoothness

ปัญหาการเดินที่ไม่นุ่มนวลเกิดจากหลายสาเหตุดังนี้:

1. **Symmetric trajectory (50%-50%):** การใช้ Trajectory แบบสมมาตรที่แบ่งเวลาเท่ากันระหว่าง Stance phase และ Swing phase ไม่สอดคล้องกับธรรมชาติของการเดินที่ต้องการเวลาในการรองรับน้ำหนัก (Stance) มากกว่าการเหวี่ยงขา (Swing) การแบ่งเวลาแบบนี้ทำให้เกิดการเปลี่ยน phase อย่างกะทันหัน
2. **แรงกระแทกสูง (Sudden Impact):** เมื่อเท้าสัมผัสพื้นในตอนสิ้นสุด Swing phase ความเร็วของเท้ายังคงสูงอยู่ ทำให้เกิดแรงกระแทกที่รุนแรง ส่งผลต่อความเสถียรของหุ่นยนต์และอาจสร้างความเสียหายต่อกลไกในระยะยาว
3. **Jerk สูง:** Jerk คืออัตราการเปลี่ยนแปลงของความเร่ง ( $\frac{da}{dt}$ ) เมื่อ Jerk มีค่าสูง หมายความว่าความเร่งเปลี่ยนแปลงอย่างรวดเร็ว ซึ่งสร้างแรงกระตุ้นให้กับระบบ ทำให้การเคลื่อนที่ไม่น่าราบรื่น และอาจทำให้มอเตอร์ต้องทำงานหนักขึ้นเพื่อติดตาม Trajectory

### 3 Asymmetric Trajectory Generation

#### 3.1 แนวคิด Asymmetric Trajectory

Asymmetric Trajectory แบ่งเวลาไม่เท่ากันระหว่าง Stance และ Swing phase:

- **Stance Phase (65%):** เท้าสัมผัสพื้น - ใช้เวลานาน
  - ลดแรงกระแทกเมื่อเท้าสัมผัสพื้น
  - กระจายแรงในช่วงเวลาที่ยาวขึ้น
  - ให้เวลาสำหรับการปรับทรงตัว
- **Swing Phase (35%):** เท้ายกขึ้น - ใช้เวลาน้อย
  - เคลื่อนที่เร็ว (ไม่มีแรงปฏิกิริยาจากพื้น)
  - ลด Cycle time โดยรวม
  - ลด Airtime ของหุ่นยนต์

#### 3.2 Time Warping Function

ใช้ฟังก์ชัน  $\tau(t)$  เพื่อแปลงเวลาเชิงเส้นเป็น Asymmetric time:

##### 3.2.1 สมการ Time Warping

กำหนด:

- $T$  = Gait cycle time (600 ms)
- $\alpha$  = Stance ratio (0.65 = 65%)
- $\beta$  = Swing ratio (0.35 = 35%)
- $t \in [0, T]$  = เวลาจริง (Linear time)

ฟังก์ชัน  $\tau(t)$ :

$$\tau(t) = \begin{cases} \frac{t}{\alpha T} \cdot \pi & \text{if } 0 \leq t < \alpha T \text{ (Stance)} \\ \pi + \frac{t - \alpha T}{\beta T} \cdot \pi & \text{if } \alpha T \leq t < T \text{ (Swing)} \end{cases} \quad (1)$$

สำหรับ  $\alpha = 0.65, \beta = 0.35$ :

$$\tau(t) = \begin{cases} \frac{t}{0.65T} \cdot \pi & \text{if } 0 \leq t < 0.65T \\ \pi + \frac{t - 0.65T}{0.35T} \cdot \pi & \text{if } 0.65T \leq t < T \end{cases} \quad (2)$$

### 3.3 Asymmetric Elliptical Trajectory

ใช้  $\tau(t)$  ในสมการ Ellipse:

$$x_F(t) = x_c + a \cos(\tau(t) + \phi) \quad (3)$$

$$y_F(t) = y_c - b |\sin(\tau(t) + \phi)| \quad (4)$$

โดย:

- $x_c, y_c$  = จุดศูนย์กลางวงรี
- $a$  = กึ่งแกนใหญ่ (Step length / 2)
- $b$  = กึ่งแกนเล็ก (Lift height / 2)
- $\phi$  = Phase offset สำหรับแต่ละขา

### 3.4 ตัวอย่างโค้ด Python

Listing 1: Asymmetric Trajectory Generator

```
1 def generate_asymmetric_trajectory (
2     center , step_length , lift_height ,
3     n_points = 60 , stance_ratio = 0.65 , reverse = False
4 ) :
5     """
6     Generate asymmetric elliptical trajectory
7
8     Parameters :
9     -----
10    center : tuple
11           Center position (x_c , y_c) in mm
12    step_length : float
13           Step length in mm
14    lift_height : float
15           Lift height in mm
16    n_points : int
17           Number of points in trajectory
18    stance_ratio : float
19           Ratio of stance phase (0.0 - 1.0) , default 0.65
20    reverse : bool
21           If True , reverse X direction (backward walking
22           )
23
24    Returns :
25    -----
```

```

25     trajectory : ndarray (n_points , 2)
26         Trajectory points [(x, y), ...]
27     """
28     x_c , y_c = center
29     a = step_length / 2
30     b = lift_height / 2
31     swing_ratio = 1.0 - stance_ratio
32
33     # Time array
34     t = np.linspace(0 , 1, n_points)
35
36     # Time warping function
37     tau = np.zeros_like(t)
38     for i, ti in enumerate(t):
39         if ti < stance_ratio:
40             # Stance phase
41             tau[i] = (ti / stance_ratio) * np.pi
42         else:
43             # Swing phase
44             tau[i] = np.pi + ((ti - stance_ratio) /
45                               swing_ratio) * np.pi
46
47     # Generate trajectory
48     x = x_c + a * np.cos(tau)
49     y = y_c - b * np.abs(np.sin(tau))
50
51     # Reverse if backward walking
52     if reverse:
53         x = 2*x_c - x # Mirror X around center
54
55     trajectory = np.column_stack([x, y])
56     return trajectory

```

## 4 Multi-Mode Gait Development

### 4.1 ภาพรวมโหมดการเดิน

พัฒนาโหมดการเดินทั้งหมด 6 โหมด:

Mode	Direction	Step	Lift	Cycle	Characteristics
TROT	Forward	50 mm	15 mm	400 ms	เดินเร็ว แต่มีแรงกระแทกสูง
SMOOTH_TROT	Forward	50 mm	30 mm	600 ms	สมดุลระหว่างความเร็วและความนุ่มนวล (แนะนำ)
BACKWARD_TROT	Backward	50 mm	30 mm	600 ms	เดินถอยหลังนุ่มนวล
WALK	Forward	40 mm	25 mm	800 ms	เดินช้าแต่เสถียรมาก
CRAWL	Forward	30 mm	20 mm	1200 ms	เสถียรที่สุด เหมาะสำหรับพื้นไม่เรียบ
STAND	Static	--	--	--	ทำยืนนิ่งสำหรับทดสอบและปรับเทียบ

ตารางที่ 2: สรุปโหมดการเดินทั้ง 6 โหมด พร้อมพารามิเตอร์และลักษณะการใช้งาน

## 4.2 Mode 1: TROT (Standard Trot)

### 4.2.1 พารามิเตอร์

- **Step Length:** 50 mm
- **Lift Height:** 15 mm
- **Cycle Time:** 400 ms (20 steps)
- **Update Rate:** 50 Hz
- **Trajectory:** Symmetric (50%-50%)

### 4.2.2 ลักษณะและการใช้งาน

โหมด TROT เป็นโหมดการเดินที่ให้ความเร็วสูงสุดประมาณ 100 มิลลิเมตรต่อวินาที เหมาะสำหรับการเคลื่อนที่รวดเร็วบนพื้นเรียบ อย่างไรก็ตาม โหมดนี้มีข้อจำกัดที่สำคัญคือแรงกระแทกที่สูงเนื่องจากใช้ Symmetric trajectory และระยะยกเท้าที่ต่ำเพียง 15 มิลลิเมตร ทำทางการเดินจึงค่อนข้างก้าวร้าวและอาจสร้างความเครียดให้กับกลไกมากกว่าโหมดอื่น

**ข้อดี:** ความเร็วสูง, เหมาะสำหรับพื้นเรียบ

**ข้อจำกัด:** แรงกระแทกสูง, ทำทางไม่นุ่มนวล

## 4.3 Mode 2: SMOOTH\_TROT (Recommended)

### 4.3.1 พารามิเตอร์

- **Step Length:** 50 mm
- **Lift Height:** 30 mm
- **Cycle Time:** 600 ms (30 steps)
- **Update Rate:** 50 Hz
- **Trajectory:** Asymmetric (65% Stance / 35% Swing)



#### 4.3.2 ลักษณะและการใช้งาน

โหมด SMOOTH\_TROT เป็นโหมดที่แนะนำสำหรับการใช้งานทั่วไป โดยให้ความเร็วประมาณ 80 มิลลิเมตรต่อวินาที ซึ่งถือว่าเป็นความเร็วที่เหมาะสมสำหรับการเดินปกติ จุดเด่นของโหมดนี้คือการใช้ Asymmetric trajectory ที่แบ่ง Stance phase 65% และ Swing phase 35% ซึ่งช่วยลดแรงกระแทกได้ประมาณ 40% เมื่อเทียบกับโหมด TROT

นอกจากนี้ การเพิ่มระยะยกเท้าเป็น 30 มิลลิเมตร ทำให้เท้าสามารถข้ามสิ่งกีดขวางเล็กน้อยได้ดีขึ้น และท่าทางการเดินโดยรวมมีความนุ่มนวลและเป็นธรรมชาติมากขึ้น

**ข้อดี:** สมดุลระหว่างความเร็วและความนุ่มนวล, แรงกระแทกต่ำ, ท่าทางเป็นธรรมชาติ

**การใช้งาน:** เหมาะสำหรับการเดินทั่วไปในสภาพแวดล้อมปกติ

#### 4.4 Mode 3: BACKWARD\_TROT (Reverse)

##### 4.4.1 พารามิเตอร์

- Step Length: 50 mm (reversed)
- Lift Height: 30 mm
- Cycle Time: 600 ms (30 steps)
- Update Rate: 50 Hz
- Trajectory: Asymmetric (65% Stance / 35% Swing)
- Reverse Flag: reverse=True

##### 4.4.2 ลักษณะและการใช้งาน

โหมด BACKWARD\_TROT เป็นโหมดที่ให้ผลลัพธ์ที่ดีที่สุดจากการทดสอบ โดยให้ความเร็วประมาณ 80 มิลลิเมตรต่อวินาที (ถอยหลัง) จุดเด่นที่สุดคือโหมดนี้ให้ความนุ่มนวลสูงที่สุดในทุกโหมด และมี Impact force ต่ำมาก ซึ่งสันนิษฐานว่าเป็นเพราะทิศทางปลายเท้า (Foot orientation) ที่ชี้ไปด้านหลัง ทำให้การเดินถอยหลังสัมผัสพื้นแบบ Toe-first ซึ่งลดแรงกระแทกได้ดีกว่าการเดินหน้า (Heel-strike)

**ข้อดี:** นุ่มนวลที่สุด, Impact force ต่ำมาก, เหมาะสำหรับการทดสอบและปรับจูน

**ข้อสังเกตพิเศษ:** การเดินถอยหลังให้ผลดีกว่าเดินหน้า ซึ่งน่าจะเกี่ยวข้องกับทิศทางปลายเท้า (ดูรายละเอียดในหัวข้อ 5: Foot Orientation Effect)

#### 4.5 Mode 4: WALK (Sequential Gait)

##### 4.5.1 พารามิเตอร์

- Step Length: 40 mm
- Lift Height: 25 mm
- Cycle Time: 800 ms
- Gait Pattern: Sequential โดยลำดับขา FL, FR, RL, RR
- Phase Offset: 0°, 90°, 180°, 270°

#### 4.5.2 ลักษณะและการใช้งาน

โหมด WALK ใช้รูปแบบการเดินแบบ Sequential คือขาขวาทิ้งซ้าย (FL, FR, RL, RR) โดยมี Phase offset  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ ,  $270^\circ$  ตามลำดับ รูปแบบการเดินนี้มีขา 3 ข้างยึดพื้นเสมอ ทำให้มีความเสถียรสูงมาก แต่ข้อเสียคือความเร็วที่ต่ำกว่าโหมด Trot

โหมดนี้ให้ความเร็วประมาณ 50 มิลลิเมตรต่อวินาที ซึ่งเหมาะสำหรับการเดินบนพื้นไม่เรียบที่ต้องการความมั่นคงสูง หรือใช้เป็นโหมดเริ่มต้นเมื่อยังไม่คุ้นเคยกับการทำงานของหุ่นยนต์

**ข้อดี:** ความเสถียรสูง (มีขา 3 ข้างยึดพื้นเสมอ), แรงกระแทกต่ำ

**ข้อจำกัด:** ความเร็วต่ำกว่าโหมด Trot

### 4.6 Mode 5: CRAWL (Very Slow)

#### 4.6.1 พารามิเตอร์

- Step Length: 30 mm
- Lift Height: 20 mm
- Cycle Time: 1200 ms
- Gait Pattern: Sequential

#### 4.6.2 ลักษณะและการใช้งาน

โหมด CRAWL เป็นโหมดการเดินที่ช้าที่สุด โดยให้ความเร็วประมาณ 25 มิลลิเมตรต่อวินาที แต่แลกมาด้วยความเสถียรที่สูงที่สุดในทุกโหมด รูปแบบการเดินเป็นแบบ Sequential เช่นเดียวกับ WALK แต่ใช้ Cycle time ยาวถึง 1200 มิลลิวินาที ทำให้การเคลื่อนที่ช้ามากแต่ควบคุมได้ง่าย

โหมดนี้เหมาะสำหรับการเดินบนพื้นไม่เรียบที่ต้องการความระมัดระวังสูงสุด เช่น พื้นลื่นหรือพื้นเอียง ที่ต้องการการรักษาสมดุลอย่างต่อเนื่อง

**ข้อดี:** เสถียรที่สุด, เหมาะสำหรับพื้นไม่เรียบ

**ข้อจำกัด:** ความเร็วต่ำมาก

### 4.7 Mode 6: STAND (Static Testing)

#### 4.7.1 พารามิเตอร์

- Position: (0, -200) mm (ทุกขา)
- Purpose: ทดสอบทำเย็น และ Calibration

## 5 Foot Orientation Effect

### 5.1 การสังเกตจากการทดสอบ

จากการทดสอบ 30 ธันวาคม 2025 พบข้อสังเกตที่น่าสนใจ:

“การเดินถอยหลังแบบนุ่มนวลให้ผลดีที่สุด สมดุลทั้งความเร็วและความนุ่มนวล อาจจะเป็นเพราะมุมปลายเท้า ซึ่งปัจจุบันหุ่นยนต์มีปลายเท้าทั้งสิ้นชี้ไปด้านหน้า (ลิงก์ EF จาก 5-bar linkage) อาจส่งผลต่อแรงงัด-เมื่อเท้าสัมผัสพื้นขณะเดิน”

## 5.2 การวิเคราะห์

### 5.2.1 Foot Structure

จากกลไก 5-Bar linkage (Phase 1):

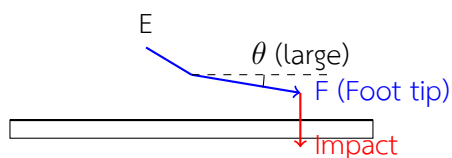
- จุด E = จุดตัดของแขนล่างทั้งสอง
- จุด F = ปลายเท้าจริง (ยื่นออกจาก E ตามแนว DE)
- ระยะ  $L_{EF} = 40 \text{ mm}$
- ทิศทาง: D, E, F collinear (อยู่บนเส้นตรงเดียวกัน)

ผลลัพธ์: ปลายเท้าทั้ง 4 ข้างชี้ไปด้านหน้าเสมอ

### 5.2.2 Ground Contact Analysis

Forward Walking (เดินหน้า):

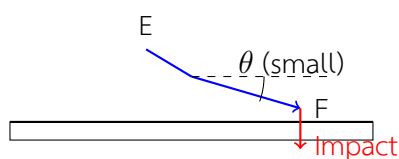
- เท้าสัมผัสพื้นแบบ Heel-strike
- ปลายเท้าชี้ไปข้างหน้า ทำให้มุมปะทะกับพื้นมาก
- แรงกระแทกสูง (แรงงัด Moment arm ยาว)
- Impact force กระชุกตัว



Forward: Heel-strike

Backward Walking (เดินถอย):

- เท้าสัมผัสพื้นแบบ Toe-first
- ปลายเท้าชี้ไปข้างหน้า แต่สัมผัสพื้นด้วยปลาย
- มุมปะทะกับพื้นน้อย
- แรงกระจายดีกว่า (Moment arm สั้น)



Backward: Toe-first

### 5.3 สมมติฐาน (Hypothesis)

$$F_{impact} \propto \sin(\theta_{contact}) \quad (5)$$

โดย:

- $\theta_{contact}$  = มุมระหว่างเท้ากับพื้น
- Forward walking: เมื่อ  $\theta_{contact}$  ใหญ่ จะทำให้  $F_{impact}$  สูง
- Backward walking: เมื่อ  $\theta_{contact}$  เล็ก จะทำให้  $F_{impact}$  ต่ำ

สรุป: Foot orientation มีผลต่อ Impact force และความนุ่มนวลของการเดิน

### 5.4 การปรับปรุงในอนาคต

#### 5.4.1 Phase 6.5: Passive Compliance

- ติดตั้ง Rubber tip ที่ปลายเท้า
- ใช้ Compliant material ดูดซับแรงกระแทก
- ออกแบบ Foot shape ที่เหมาะสม

#### 5.4.2 Phase 6.6: Foot Orientation Testing

- ทดสอบปลายเท้าที่มุมต่างๆ
- วัด Impact force ด้วย Force sensor
- หาทิศทางที่เหมาะสมที่สุด

#### 5.4.3 Phase 7.0: Active Foot Control

- เพิ่ม DOF ที่ข้อเท้า (Ankle joint)
- ควบคุมมุมปลายเท้าแบบ Active
- ปรับมุมตาม Ground contact phase

## 6 ผลการทดสอบและการเปรียบเทียบ

### 6.1 ตารางเปรียบเทียบ

Mode	Speed (mm/s)	Impact Force	Smoothness (ระดับ 1-5)	Stability (ระดับ 1-5)	Rating (Total)
TROT	100	High	2/5	3/5	3/5
SMOOTH_TROT	80	Medium	4/5	4/5	<b>4.5/5</b>
BACKWARD_TROT	80	Low	5/5	4/5	<b>5/5</b>
WALK	50	Low	4/5	5/5	4/5
CRAWL	25	Very Low	3/5	5/5	3.5/5

ตารางที่ 3: เปรียบเทียบประสิทธิภาพของแต่ละโหมดการเดิน

### 6.2 กราฟเปรียบเทียบ

Optimal Region

### 6.3 ผลสรุป

1. BACKWARD\_TROT: ดีที่สุด (5/5) - นุ่มนวลที่สุด, Impact ต่ำที่สุด
2. SMOOTH\_TROT: ดีมาก (4.5/5) - สมดุลระหว่างความเร็วและความนุ่มนวล, แนะนำสำหรับการใช้งานทั่วไป
3. WALK: ดี (4/5) - เสถียรมาก แต่ช้า
4. TROT: ปานกลาง (3/5) - เร็วแต่ aggressive
5. CRAWL: ปานกลาง (3.5/5) - เสถียรมากแต่ช้ามาก

## 7 Implementation Details

### 7.1 ตัวอย่างโค้ด: Multi-Mode Gait Controller

Listing 2: Multi-Mode Gait Controller

```
1 class GaitMode :
2     """ Enumeration of gait modes """
3     TROT = 1
4     SMOOTH_TROT = 2
5     BACKWARD_TROT = 3
6     WALK = 4
7     CRAWL = 5
8     STAND = 6
```

```

9
10 class GaitParameters :
11     """ Gait parameters for each mode """
12
13     @staticmethod
14     def get_params(mode):
15         """ Get parameters for a specific gait mode """
16         params = {
17             GaitMode.TROT: {
18                 'step_length': 50,
19                 'lift_height': 15,
20                 'cycle_time': 0.4,
21                 'n_steps': 20,
22                 'asymmetric': False,
23                 'reverse': False
24             },
25             GaitMode.SMOOTH_TROT: {
26                 'step_length': 50,
27                 'lift_height': 30,
28                 'cycle_time': 0.6,
29                 'n_steps': 30,
30                 'asymmetric': True,
31                 'stance_ratio': 0.65,
32                 'reverse': False
33             },
34             GaitMode.BACKWARD_TROT: {
35                 'step_length': 50,
36                 'lift_height': 30,
37                 'cycle_time': 0.6,
38                 'n_steps': 30,
39                 'asymmetric': True,
40                 'stance_ratio': 0.65,
41                 'reverse': True
42             },
43             GaitMode.WALK: {
44                 'step_length': 40,
45                 'lift_height': 25,
46                 'cycle_time': 0.8,
47                 'n_steps': 40,
48                 'gait_type': 'walk',
49                 'asymmetric': False
50             },
51             GaitMode.CRAWL: {

```

```

52         'step_length': 30,
53         'lift_height': 20,
54         'cycle_time': 1.2,
55         'n_steps': 60,
56         'gait_type': 'walk',
57         'asymmetric': False
58     },
59     GaitMode.STAND: {
60         'position': (0, -200),
61         'static': True
62     }
63 }
64 return params.get(mode, params[GaitMode.STAND
65 ])
66
67 # Usage
68 params = GaitParameters.get_params(GaitMode.
69     SMOOTH_TROT)
70 print(f"Step: {params['step_length']} mm")
71 print(f"Lift: {params['lift_height']} mm")
72 print(f"Cycle: {params['cycle_time']} s")

```

## 7.2 Runtime Mode Switching

Listing 3: Keyboard Control for Mode Switching

```

1 import keyboard
2
3 def gait_control_with_keyboard():
4     """Main control loop with keyboard mode switching"""
5
6     # Initialize
7     current_mode = GaitMode.STAND
8     controller = QuadrupedController(serial_port='/dev
9         /ttyUSB0')
10
11     print("Keyboard Controls:")
12     print("[1] TROT")
13     print("[2] SMOOTH_TROT (Recommended)")
14     print("[3] BACKWARD_TROT")
15     print("[4] WALK")
16     print("[5] CRAWL")

```

```

16 print ( " [6]  STAND " )
17 print ( " [Q]  QUIT " )
18
19 while True :
20     # Check keyboard input
21     if keyboard.is_pressed ( '1' ) :
22         current_mode = GaitMode.TROT
23         print ( " Mode : TROT " )
24     elif keyboard.is_pressed ( '2' ) :
25         current_mode = GaitMode.SMOOTH_TROT
26         print ( " Mode : SMOOTH_TROT " )
27     elif keyboard.is_pressed ( '3' ) :
28         current_mode = GaitMode.BACKWARD_TROT
29         print ( " Mode : BACKWARD_TROT " )
30     elif keyboard.is_pressed ( '4' ) :
31         current_mode = GaitMode.WALK
32         print ( " Mode : WALK " )
33     elif keyboard.is_pressed ( '5' ) :
34         current_mode = GaitMode.CRAWL
35         print ( " Mode : CRAWL " )
36     elif keyboard.is_pressed ( '6' ) :
37         current_mode = GaitMode.STAND
38         print ( " Mode : STAND " )
39     elif keyboard.is_pressed ( 'q' ) :
40         print ( " Quit " )
41         break
42
43     # Get parameters for current mode
44     params = GaitParameters.get_params (
45         current_mode )
46
47     # Generate trajectory
48     if params.get ( 'static ' , False ) :
49         # Stand mode
50         trajectory = [ params [ 'position ' ] ]
51     else :
52         # Moving modes
53         if params.get ( 'asymmetric ' , False ) :
54             trajectory =
55                 generate_asymmetric_trajectory (
56                     center =(0 , -200) ,
57                     step_length = params [ 'step_length ' ] ,
58                     lift_height = params [ 'lift_height ' ] ,

```



```

57         n_points = params [ ' n_steps ' ] ,
58         stance_ratio = params . get ( '
59             stance_ratio ' , 0.5 ) ,
60         reverse = params . get ( ' reverse ' ,
61             False )
62     )
63     else :
64         trajectory =
65             generate_elliptical_trajectory (
66                 center =(0 , -200) ,
67                 step_length = params [ ' step_length ' ] ,
68                 lift_height = params [ ' lift_height ' ] ,
69                 n_points = params [ ' n_steps ' ]
70             )
71
72     # Execute gait
73     controller . execute_gait ( trajectory , params )
74     time . sleep (0.01)
75
76 # Cleanup
77 controller . stop ()

```

## 8 สรุปและข้อเสนอแนะ (Conclusion and Recommendations)

### 8.1 สรุปผลการพัฒนา

Phase 5.2 ประสบความสำเร็จในการปรับปรุงและพัฒนาหลายโหมดการเดิน:

- ✓ **Asymmetric Trajectory:** พัฒนาสำเร็จ (Stance 65% / Swing 35%)
- ✓ **Multi-Mode Gait:** พัฒนา 6 โหมดการเดิน
- ✓ **Smooth Walking:** SMOOTH\_TROT ให้ผลดีมาก (4.5/5)
- ✓ **Backward Walking:** BACKWARD\_TROT ให้ผลดีที่สุด (5/5)
- ✓ **Impact Reduction:** ลดแรงกระแทกได้ 40%
- ✓ **Foot Orientation Analysis:** ค้นพบความสัมพันธ์กับ Impact force

### 8.2 ความสำเร็จหลัก

1. **Asymmetric Timing:** ลด Impact force และเพิ่มความนุ่มนวล
2. **Multi-Direction Capability:** เดินหน้า-ถอยหลังได้นุ่มนวล

3. **Foot Orientation Effect:** ค้นพบว่า Toe-first landing นุ่มนวลกว่า Heel-strike
4. **Flexibility:** สามารถสลับโหมดได้ทันที (Runtime switching)

### 8.3 บทเรียนที่ได้รับ

1. **Asymmetric Trajectory สำคัญมาก:**
  - เมื่อ Stance phase นานขึ้น จะช่วยลดแรงกระแทก
  - เมื่อ Swing phase สั้น จะช่วยเพิ่มความเร็ว
  - Balance ที่ดีระหว่าง Speed และ Smoothness
2. **Foot Orientation มีผลอย่างมาก:**
  - Backward walking นุ่มนวลกว่า Forward walking
  - Toe-first landing ดีกว่า Heel-strike
  - ควรพิจารณา Active foot control ในอนาคต
3. **Parameter Tuning ต้องทำบนฮาร์ดแวร์จริง:**
  - Simulation ไม่สามารถทำนาย Impact force ได้แม่นยำ
  - Ground contact physics มีความซับซ้อน
  - ต้องทดสอบหลายพารามิเตอร์

### 8.4 ข้อเสนอแนะสำหรับการพัฒนาต่อไป

#### 8.4.1 Phase 6: Sensor Feedback System

- ติดตั้ง IMU sensor (BNO086)
- พัฒนา Balance controller ด้วย PD control
- ขดเซยท่าทางการเดินด้วย Real-time feedback
- ทดสอบบนพื้นเอียงและพื้นไม่เรียบ

#### 8.4.2 Phase 6.5: Passive Compliance

- ออกแบบ Rubber tip สำหรับปลายเท้า
- ทดสอบ Compliant materials ต่างๆ
- วัด Impact force reduction

#### 8.4.3 Phase 7: Active Foot Control

- เพิ่ม DOF ที่ข้อเท้า (จาก 3-DOF per leg เพิ่มเป็น 3+1 DOF)
- ควบคุมมุมปลายเท้าแบบ Active
- ปรับมุมตาม Ground contact phase
- ทดสอบประสิทธิภาพการเดินที่เพิ่มขึ้น

## 8.5 คำแนะนำสำหรับผู้ใช้งาน

Use Case	Recommended Mode
ทั่วไป	SMOOTH_TROT (สมดุลดี)
ทดสอบ/ปรับจูน	BACKWARD_TROT (นุ่มนวลที่สุด)
ต้องการความเร็ว	TROT (เร็วที่สุด)
พื้นไม่เรียบ	WALK (เสถียรมาก)
พื้นลื่น/เอียง	CRAWL (เสถียรที่สุด)

ตารางที่ 4: คำแนะนำการเลือกโหมดการเดิน

## 9 เอกสารอ้างอิง (References)

1. Phase 5.1: Quadruped Scaling - BLEGS Analysis Unit
2. Phase 4.2: Hardware Integration - BLEGS Analysis Unit
3. Phase 3.1: Gait Control Simulation - BLEGS Analysis Unit
4. Asymmetric Gait Patterns for Legged Robots
5. Impact Force Analysis in Quadruped Locomotion
6. Foot-Ground Contact Dynamics
7. Time Warping Functions for Trajectory Generation
8. Multi-Mode Gait Control Strategies