Institute of Field Robotics, King Mongkut's University of Technology Thonburi
126 Pracha Uthit Rd, Bang Mot, Thung Khru, Bangkok, Thailand 10140
Senior Thesis FRAB (FIBO Robotics and Automation: Bachelor)

DYNAMIC JUMPING MECHANISM IN A ONE-LEGGED ROBOT FOR CROSSING WALLS

<ธรรมภณ พลายแดง, 65340500028>
<ปานศิริ พงศ์วงประเสริฐ, 65340500038>

บทคัดย่อ

โปรเจคนี้จัดทำขึ้นสำหรับศึกษาการจำลองการกระโดดข้ามสิ่งกีดขวางของหุ่น Single degree of freedom ในรูปแบบ 2 มิติ โดยมีใช้ damping เพื่อควบคุมการกระโดด และใช้การคำนวณตามหลัก Lagrangian mechanics เพื่อวิเคราะห์พลังงาน และแรงที่จำเป็นในการกระโดดให้ผ่านสิ่งกีดขวางไปได้ เมื่อทราบสัดส่วนโครงสร้างของหุ่น รวมทั้ง Parameters ต่าง ๆ และเพื่อศึกษา Kinematics สำหรับควบคุมตำแหน่ง นอกจากนี้ ยังติดตั้งส่วนหางเพื่อรักษาสมดุลทั้งขณะเคลื่อนที่ และหยุดนิ่ง เพื่อเพิ่มเสถียรภาพในการกระโดด

คำสำคัญ: Jump simulation, monopedal robot

1. บทนำ (Introduction)

ในปัจจุบัน วิทยาการหุ่นยนต์มีการพัฒนาไปอย่างรวดเร็ว หุ่นยนต์ไม่เพียงแต่ถูกออกแบบมาเพื่อทำงานในโรงงานอุตสาหกรรมเท่านั้น แต่ยังสามารถนำไปประยุกต์ใช้ในสถานการณ์ที่ซับซ้อน และท้าทายมากขึ้น เช่น การกู้ภัยในพื้นที่ที่ยากต่อการเข้าถึง การสำรวจพื้นที่อันตราย หรือการค้นหาผู้รอดชีวิตในสถานที่ที่มนุษย์ไม่สามารถเข้าถึงได้อย่างรวดเร็ว ในสถานการณ์เช่นนี้ หุ่นยนต์จำเป็นต้องมีความสามารถในการเคลื่อนที่ ผ่านสิ่งกีดขวาง และปรับตัวให้เหมาะสมกับสภาพแวดล้อมที่เปลี่ยนแปลงอยู่ตลอดเวลา

หนึ่งในเทคโนโลยีที่ได้รับการศึกษาอย่างมากในช่วงที่ผ่านมา คือ "หุ่นยนต์กระโดด" (Jumping Robot) ซึ่งเลียนแบบพฤติกรรมการกระโดดของสิ่งมีชีวิต เช่น แมลง หรือสัตว์เลี้ยงลูกด้วยนมขนาดเล็ก ที่สามารถกระโดดข้ามสิ่งกีดขวาง หรือเคลื่อนที่ไปยังที่สูงได้อย่างรวดเร็ว การเคลื่อนที่แบบกระโดดช่วยให้หุ่นยนต์สามารถหลบหลีกสิ่งกีดขวางได้ในเวลาสั้น และเข้าถึงพื้นที่ที่ยากต่อการเคลื่อนที่แบบล้อ หรือขาเดินปกติ

อย่างไรก็ตาม การออกแบบหุ่นยนต์ที่สามารถกระโดดข้ามสิ่งกีดขวางได้นั้น ต้องอาศัยการคำนวณที่ซับซ้อนในด้าน Dynamics และ Kinematics เพื่อให้หุ่นยนต์สามารถทรงตัวในอากาศ และลงสู่พื้นได้อย่างมั่นคง โดยเฉพาะเมื่อหุ่นยนต์ต้องกระโดดให้ได้ความสูง และทิศทางที่แม่นยำ นอกจากนี้ การควบคุมการเคลื่อนไหวของหาง หรือส่วนประกอบเสริมเพื่อรักษาสมดุลระหว่างการกระโดด และลงพื้นยังเป็นสิ่งจำเป็น

โครงการนี้จึงมีความสำคัญในการศึกษาการออกแบบ และการควบคุมการเคลื่อนที่ของหุ่นยนต์กระโดด โดยเน้นการใช้ Lagrangian mechanics ในการคำนวณพลังงาน และแรงที่จำเป็นในการกระโดด และใช้ Inverse Kinematics และ Trajectory Planning เพื่อกำหนดเส้นทาง และมุมการเคลื่อนไหวของข้อต่อ และหาง การศึกษาในโครงการนี้จะช่วยพัฒนาความรู้ความเข้าใจในการออกแบบ และควบคุมหุ่นยนต์ที่สามารถกระโดดข้ามสิ่งกีดขวางได้อย่างมีประสิทธิภาพ และสามารถนำไปประยุกต์ใช้ในสถานการณ์ที่ซับซ้อนในอนาคต

1.1 จุดประสงค์โครงการ

1) เพื่อศึกษา Dynamics ในปริภูมิ 2 มิติ สำหรับหุ่นกระโดด 1 DOF

2) เพื่อพัฒนา Toolbox สำหรับการหา Inverse kinematics ของปลายหางเพื่อที่จะคำนวณตำแหน่งหาง และการหา Forward kinematics เพื่อหาตำแหน่งการวางตำแหน่งที่จะรักษาสมดุล, คำนวณแรงบิดที่ต้องใช้ และพลังงานที่ต้องใช้ในการกระโดดตามหลัก Lagrangian mechanics และ Simulation ของหุ่น 1 DOF

3) เพื่อศึกษาระบบ Control ที่จะสามารถรักษาสมดุลทั้งขณะเคลื่อนที่ และหยุดนิ่ง

1.2 ขอบเขต

1) Parameters และ mode ของหุ่น

A. Parameters and scale

I. Parameters ของหุ่นในส่วนของ Scale จะถูก Fix ไว้อยู่แล้ว

II. แม้การส่งแรงจะมาจาก Spring หรือ SEA แต่การจำลองขาของหุ่นจะจำลองเป็นรูปแบบ link เดียวใน simulation

B. หุ่นจะมีด้วยกัน 2 Modes

I. Mode 1 สามารถปรับ Torque ของมอเตอร์เพื่อดูความสูงการกระโดดได้

II. Mode 2 สามารถปรับความสูงกำแพงเพื่อให้หุ่นสามารถจำลองการกระโดดข้ามกำแพงนั้นได้

2) จำลองการเคลื่อนที่ของหุ่นข้ามกำแพงของหุ่นเพื่อกระโดดข้ามสิ่งกีดขวาง โดยศึกษากลไกการทำงานของหุ่น SINGLE DEGREE OF FREEDOM (DOF) ในการเคลื่อนที่แบบ 2 มิติ

A. จังหวะเริ่มกระโดดของหุ่น

I. การคำนวณแรงบิด (Torque) และพลังงานที่ต้องใช้ในการกระโดดให้พ้นสิ่งกีดขวาง ตามหลักการของ Lagrangian mechanics เพื่อกำหนดแรง และพลังงานที่หุ่นต้องการในการกระโดด

II. การคำนวณ Kinematics ตำแหน่งหาง

B. จังหวะลอยตัวกลางอากาศ

I. การลอยตัวกลางอากาศ โดยคำนวณ และวิเคราะห์ Trajectory ของหุ่นยนต์ รวมถึงการกำหนดตำแหน่ง และความเร็วขณะลอยตัว เพื่อให้การเคลื่อนที่มีเสถียรภาพ และควบคุมได้ เพื่อวิเคราะห์ Trajectory

II. การคำนวณมุม และตำแหน่งของหางเพื่อรักษาสมดุลของหุ่นยนต์ระหว่างการลอยตัว

III. การปรับมุม และตำแหน่งของหาง รวมถึงการใช้แรงจาก Reaction wheel เพื่อรักษาสมดุลระหว่างการลอยตัวในอากาศ และปรับทิศทางการลอยตัวให้คงเส้นทางที่ต้องการ

C. จังหวะที่หุ่นยนต์สัมผัสพื้น

I. การปรับตำแหน่ง มุมของขา และหางหลังจากสัมผัสพื้น เพื่อเตรียมความพร้อมในการกระโดดครั้งถัดไป โดยให้หุ่นยนต์อยู่ในมุม และทิศทางที่เหมาะสมสำหรับการกระโดดใหม่อย่างเสถียร

D. การจำลองการเคลื่อนที่ผ่านโปรแกรมจำลอง

I. การจำลองการเคลื่อนที่จะจำลองผ่าน Simscape Multibody / Python

2. ทบทวนวรรณกรรม และ ทฤษฎีที่เกี่ยวข้อง (Literature Review)

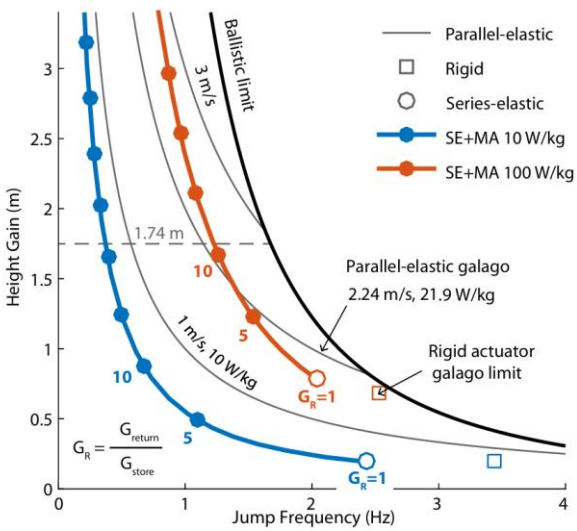
2.1 ROBOTIC VERTICAL JUMPING AGILITY VIA SERIES-ELASTIC POWER MODULATION

2.1.1 Designing a Robotic Galago

หมายเหตุ

- SE (Series-Elastic): หมายถึง Series-Elastic Actuator ซึ่งเป็นระบบขับเคลื่อนที่มีองค์ประกอบสปริงเชื่อมต่อโดยตรงกับมอเตอร์
- MA (Mechanical Advantage): หมายถึง อัตราทดกำลังทางกล
- GR (Gear Ratio): หมายถึง อัตราทดเกียร์

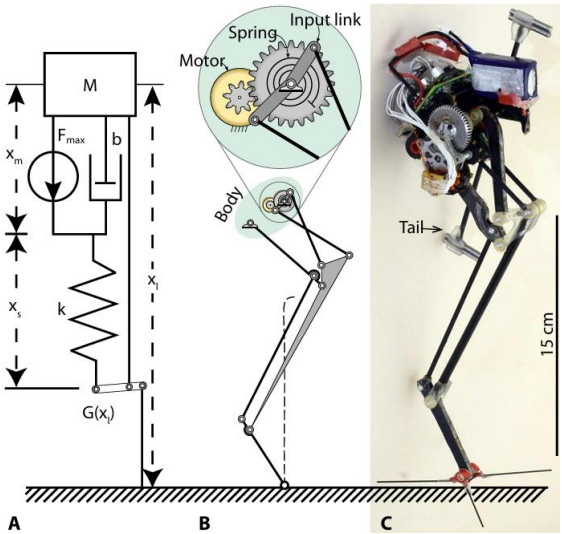
ในปัจจุบัน หุ่นยนต์ยังไม่สามารถทำการกระโดดแนวดิ่งได้อย่างคล่องตัวเทียบเท่ากับ Galago การศึกษาในครั้งนี้มุ่งเน้นไปที่การประเมิน Power Density (พลังงานสูงสุดต่อหน่วยมวล) ที่จำเป็นสำหรับระบบหุ่นยนต์เพื่อให้สามารถกระโดดได้ในระดับที่ใกล้เคียงกับ Galago โดยกำหนดมวลและความยาวของการยืดขาให้เท่ากับ Galago (0.25 กก. และ 0.15 ม.) และทำการศึกษาเกี่ยวกับตัวขับเคลื่อนประเภท Rigid, Parallel-Elastic, Series-Elastic, และ SE+MA ที่มี Power Density สองระดับคือ 10 และ 100 W/kg สำหรับ SE+MA ได้ถูกออกแบบให้มี *Mechanical Advantage* ที่แตกต่างกันในระหว่างการยืดขา เพื่อให้สามารถเก็บ และปล่อยพลังงานได้อย่างมีประสิทธิภาพสูงสุด ซึ่งแสดงเป็นอัตราส่วน GR (อัตราส่วนของ MA ตอนปลายเทียบกับตอนเริ่มต้น)



รูปที่ 1 The Effect of Actuation on Vertical Jumping Agility.

รูปที่ 1 แสดงให้เห็นว่าทั้ง Series-Elastic และ SE+MA สามารถกระโดดได้สูงกว่าระบบ Rigid ที่มี Power Density เท่ากัน เนื่องจาก Actuator สามารถส่งพลังงานไปยังองค์ประกอบยืดหยุ่นเพื่อเก็บพลังงาน และยืดเวลาการสัมผัสพื้น ทำให้เพิ่มพลังงานสุทธิได้ การเพิ่ม GR ช่วยให้ระบบ SE+MA กระโดดได้สูงขึ้น และเพิ่มความคล่องตัวในการกระโดดแนวดิ่ง

2.1.2 A Robotic Prototype Applying Power Modulation



รูปที่ 2 Power Modulation Modeled And Instantiated.

ต้นแบบหุ่นยนต์ Salto (รูปที่ 2C) ถูกพัฒนาขึ้นเพื่อทดสอบการควบคุมพลังงาน (Power Modulation) และผลกระทบต่อสมรรถนะการเคลื่อนที่ หุ่นยนต์นี้เป็นระบบกระโดดขาเดียวที่ใช้ Series-Elastic Actuator และหางควบคุมท่าทาง โดยออกแบบให้มีมวลเบาเพื่อลดพลังงานที่ใช้ต่อหน่วยความสูง และเพิ่มความทนทานต่อแรงกระแทก โดยค่าพารามิเตอร์ทางกายภาพของ Salto ที่แสดงรายละเอียดต่าง ๆ เช่น มวล และแรงบิด สรุปไว้ในตารางที่ 1

Table 1. Actuation strategies for designing a hypothetical robotic galago.

	Galago (8)	Hypothetical rigid	Hypothetical series-elastic	Hypothetical parallel-elastic	Hypothetical SE +MA	EPFL Jumper	Salto
Mass (kg)	0.250	0.250	0.250	0.250	0.250	0.007	0.1000
Leg length (m)	0.15	0.15	0.15	0.15	0.15	0.1	0.15
Maximum jump height (m)	1.74	1.74	1.74	1.74	1.74	1.38	1.008
Jump frequency (Hz)	1.29	1.66	1.59	1.29	1.29	0.248	1.74
Vertical jumping agility (m/s)	2.24	2.89	2.78	2.24	2.24	0.34	1.75
Power density (W/kg)	92.7 (estimated)	343	325	21.9	90	50	137

ตารางที่ 1 Actuation Strategies for Designing a Hypothetical Robotic Galago.

ระบบใช้ Planar Eight-Bar Revolute Linkage (รูปที่ 2B) เพื่อสร้างโปรไฟล์ MA ซึ่งรองรับการควบคุมพลังงาน โดยเริ่มจาก MA ต่ำที่ 2.49 N/Nm ในช่วงเก็บพลังงาน และเพิ่มเป็นค่าเฉลี่ย 26.90 N/Nm ในช่วงปล่อยพลังงาน โดยมีค่า GR เท่ากับ 10.8 การออกแบบนี้ช่วยลดแรงสูงสุดในช่วงปล่อยพลังงาน และอัตราเร่งสูงสุดลงครึ่งหนึ่งเมื่อเปรียบเทียบกับระบบที่ใช้ MA คงที่

2.1.3 Synthesis of Leg Mechanism

กลไกขาแบบ SE+MA ได้รับการออกแบบโดยอิงจากวิธีการของ Plecnik Et Al. เพื่อเปลี่ยนแรงบิดจากสปริงให้กลายเป็นแรงปฏิกิริยาที่กระทำผ่านจุดศูนย์ถ่วงของต้นแบบ แนวทางนี้ได้สำรวจกลไกเชิงกลที่ประกอบด้วยจุดหมุนแบบ Revolute Joint และแก้ไขปัญหาค่าพารามิเตอร์ในพื้นที่เรขาคณิตที่ไม่เป็นเชิงเส้นสูงด้วยวิธีการสองขั้นตอน ได้แก่ การสำรวจการออกแบบเบื้องต้น และการปรับแต่งเชิงจลนศาสตร์ โดยการสร้างสมการพหุนาม และใช้ Bertini ในการหาคำตอบ จากนั้นนำตำแหน่งที่ดีที่สุดไปปรับจูนด้วย Kinematic Tuning เพื่อให้สอดคล้องกับข้อกำหนดที่ละเอียด เช่น

การปรับปรุงพฤติกรรม เช่น การหมุนลิงค์ป้อนกำลังในมุมกว้าง การกำหนดให้มี Mechanical Advantage (MA) เริ่มต้นที่ต่ำ และการควบคุมแรงปฏิกิริยาที่ปลายเท้าให้คงที่ตลอดระยะการเคลื่อนที่ที่เหลือ

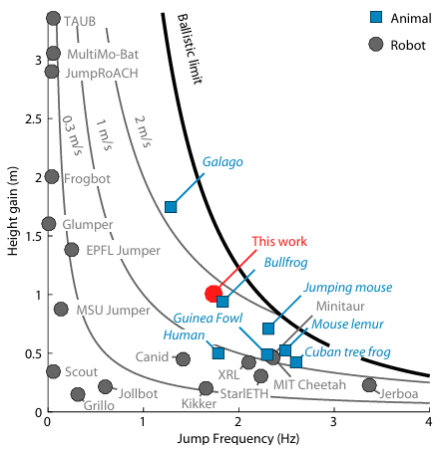
การจำลองแบบ Dynamic ของการกระโดดสูงสุดได้แสดงให้เห็นถึงแรงปฏิกิริยาภายใน และช่วยในการออกแบบต้นแบบที่ขับเคลื่อนด้วยสปริง แต่พบว่าการออกแบบ Six-Bar มีความยากในการปรับสมดุล จึงได้เพิ่มลิงค์อีกสองลิงค์ และทำการปรับแต่งอีกครั้งจนได้การออกแบบสุดท้ายตามที่แสดงใน (รูปที่ 2B)

2.1.4 Optimal Design of Jumping Systems

การออกแบบระบบกระโดดสำหรับ Series-Elastic และ *SE+MA* (รูปที่ 1) ถูกกำหนดโดยการระบุค่ามวล ความยาวการยืดขา Power Density และอัตราส่วน MA พร้อมทั้งปรับค่าตัวแปรเพื่อเพิ่มความคล่องตัว ตัวแปรที่ใช้ ได้แก่ แรงสูงสุด F_{max}, พลังงานที่เก็บสะสม G_{store}, ความแข็งของสปริง k, และจุดเปลี่ยนผ่านการควบคุม MA ค่าต่ำสุดของ Power Density ใน (ตารางที่ 1) ถูกกำหนดผ่านการค้นหาแบบกริด (Grid Search) โดยยึดตามข้อจำกัดความยาวสูงสุดของการยืดขา และความเร็วขณะกระโดดที่ต่ำสุดที่จำเป็นเพื่อบรรลุความสูงการกระโดดของ Galago

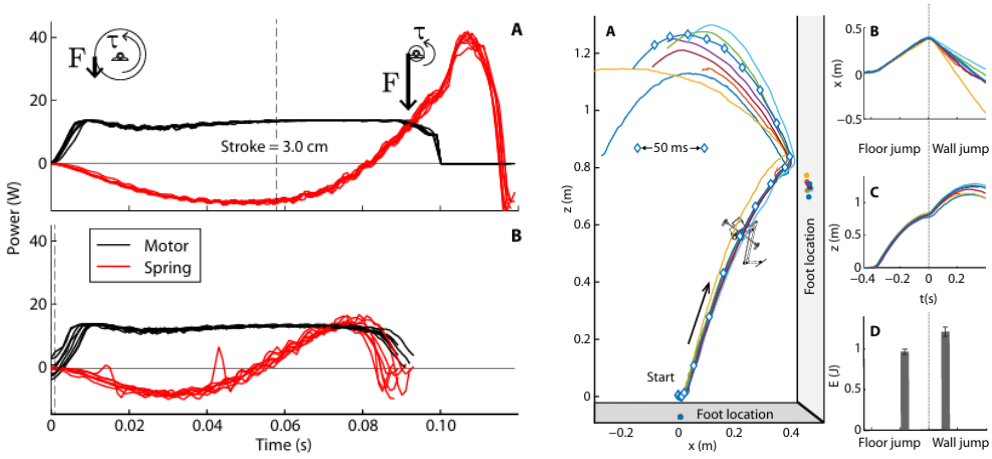
2.1.5 Discussion

การปรับพลังงานแบบ Series-Elastic ได้รับการทดสอบผ่านหุ่นยนต์ต้นแบบ โดยผลลัพธ์ที่ได้แสดงให้เห็นว่าสมรรถนะการกระโดดแนวดิ่งสูงสุดที่บันทึกไว้คือ 1.75 m/s ซึ่งสูงกว่าของ Bullfrog แต่ยังต่ำกว่าของ Galago ที่มีค่า 2.24 m/s การควบคุมพลังงานนี้ช่วยลดระดับ Power Density ที่จำเป็นสำหรับมอเตอร์ โดยพบว่าหากใช้ Actuator แบบ Rigid หรือ Series-Elastic โดยไม่มีการปรับค่า MA จะต้องใช้ Power Density สูงเกินไปเพื่อให้ได้สมรรถนะที่เทียบเท่ากับ Galago ในทางกลับกัน การเพิ่มการปรับค่า MA แบบช่วงคงที่ในระบบที่รวมกับ Series-Elastic Actuator จะทำให้สามารถลด Power Density ได้ถึง 3.6 เท่า ซึ่งช่วยให้สามารถสร้างหุ่นยนต์ที่มีความคล่องตัวใกล้เคียงกับ Galago ได้



รูปที่ 3 Height Gain and Jump Frequency for Animal and Untethered Robotic Systems.

อีกทางเลือกหนึ่งคือการใช้ Parallel-Elastic Actuator ตามที่แสดงใน (รูปที่ 3) ซึ่งต้องการ Power Density น้อยกว่า SE+MA แต่มีข้อจำกัดในการควบคุมแรงในขาระหว่างการกระโดด หุ่นยนต์ Salto สามารถหดขาได้ที่ความเร็วสูงสุดของมอเตอร์เพื่อเตรียมตัวชนผนัง ซึ่งกลไกแบบ Parallel-Elastic ไม่สามารถทำได้อย่างมีประสิทธิภาพ



รูปที่ 4 และ 5 Spring and Motor Power During Jumps. และ Trajectories of the robotic wall jump experiments. ตามลำดับ

กลไกขาแบบ SE+MA แสดงให้เห็นถึงประสิทธิภาพ และความเสถียร ตามที่แสดงในกราฟพลังงานใน (รูปที่ 4) และการเบี่ยงเบนของความสูงเล็กน้อยในการทดสอบการกระโดดแนวดิ่ง การเบี่ยงเบนของเส้นทางการเคลื่อนที่ใน (รูปที่ 5) ไม่ได้เกิดจากข้อบกพร่องของกลไก แต่เกิดจากการขาดระบบควบคุมแบบ Closed-Loop ในการควบคุมการทำงาน

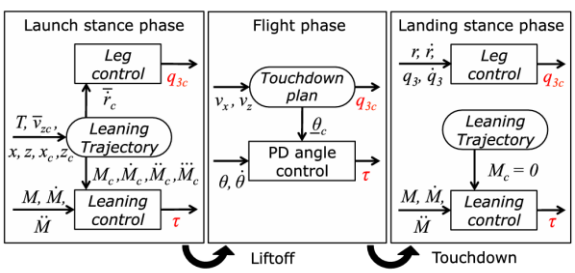
หุ่นยนต์สามารถกระโดดชนผนังเพื่อเพิ่มความสูง ซึ่งแสดงให้เห็นถึงการใช้คุณสมบัติของสิ่งแวดล้อมเพื่อขยายพื้นที่ทำงาน อย่างไรก็ตาม ข้อจำกัดหลักของการทดลองคือหุ่นยนต์สามารถเคลื่อนไหวได้เฉพาะในระนาบด้านหน้า-หลัง (Sagittal Plane) เท่านั้น การทดลองดำเนินการบนพื้นผิวแข็งที่มีการยึดเกาะดี และยังขาดการรับรู้สิ่งแวดล้อมหรือท่าทางสัมพัทธ์ของหุ่นยนต์ ทำให้การกระโดดชนผนังเป็นแบบ Open-Loop ที่ตั้งค่าพารามิเตอร์ไว้ล่วงหน้า

2.2 Precision Robotic Leaping and Landing using Stance-Phase Balance

2.2.1 Motivation and Principles

เป้าหมายคือการออกแบบให้หุ่นยนต์สามารถกระโดดได้ระยะไกล และแม่นยำ พร้อมลงจอดอย่างมั่นคงบนพื้นที่รองรับขนาดเล็ก โดยระหว่างการยืน (Stance Phase) การเคลื่อนที่ของหุ่นยนต์แบ่งเป็นสองส่วนคือ การเอน (Leaning) ของจุดศูนย์ถ่วง (CG) และการเคลื่อนที่ในทิศทางรัศมีรอบจุดรองรับ ซึ่งแบ่งการควบคุมเป็นการควบคุมการเอน และการควบคุมขาตามลำดับ ทั้งนี้ การเคลื่อนที่ในอากาศของหุ่นยนต์จะอยู่ในระนาบแนวดิ่ง เช่นเดียวกับการเคลื่อนไหวระหว่างการยืน

ในช่วงก่อนการกระโดด (Launch Stance Phase), การควบคุมการเอน และขาจะกำหนดท่าทางเพื่อให้หุ่นยนต์ออกตัวในเส้นทางที่ต้องการ ส่วนในช่วงการบิน (Flight Phase), แผนการสัมผัสพื้นจะกำหนดท่าทางการลงจอด และในช่วงยืนหลังการลงจอด (Landing Stance Phase), การควบคุมการเอนจะช่วยรักษาสมดุล ขณะที่การควบคุมขาจะชะลอการเคลื่อนที่ (รูปที่ 6)



รูปที่ 6 Control Block Diagram for Launch Flight and Lading.

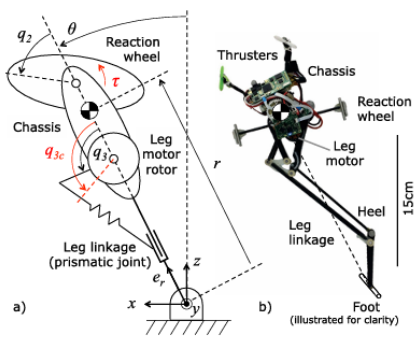
หุ่นยนต์ในช่วงวิถีโค้ง (Ballistic Flight Phase) ไม่มีการควบคุมการเคลื่อนที่ของ CG ในอากาศ จึงต้องตั้งค่าการปล่อยตัวอย่างแม่นยำ โดยเงื่อนไขการปล่อยตัวที่กำหนดให้มีความเร็วเชิงมุมของ CG และโมเมนตัมเชิงมุมสุทธิเป็นศูนย์ช่วยลดความคลาดเคลื่อนในการปล่อยตัว โดยจะแสดงให้เห็นว่ามีความแม่นยำมากกว่าการวิ่งแบบ SLIP ในกรณีที่มีข้อผิดพลาดที่มุมขา

การลงจอดที่สมดุลบนพื้นแคบจำเป็นต้องมีมุม และความเร็วที่เหมาะสม ซึ่งขีดจำกัดเหล่านี้ถูกคำนวณใน 2.2.6 Touchdown Plan ทั้งนี้การควบคุมมุมการเอนมีความสำคัญอย่างยิ่งต่อการปล่อยตัว และการลงจอด

2.2.2 Robot Model and Parameters

หุ่นยนต์ Salto-1P ประกอบด้วยสามส่วนหลัก ได้แก่ Chassis ล้อหมุนควบคุมทิศทาง (Reaction Wheel), และมอเตอร์ขา แต่ละส่วนมีโมเมนต์ความเฉื่อยรอบแกน Y เป็น (I_1), (I_2), และ (I_3) ตามลำดับ (รูปที่ 7) โดยข้อต่อแบบหมุนขนานกับแกน Y เชื่อมต่อแชสซีกับล้อหมุนควบคุม และมอเตอร์ขา ทำให้การหมุนของข้อต่อ (q_2) และ (q_3) ไม่ส่งผลต่อจุดศูนย์ถ่วง CG ระบบขาทำงานเป็นข้อต่อปริซึมแบบไร้มวล เชื่อมระหว่างแชสซี และเท้าตามแนวแกน (e_r)

ฐานเท้าถูกปรับให้เป็นแท่งขนานกับแกน Y เพื่อทำหน้าที่เป็นข้อต่อแบบหมุนภายใต้เงื่อนไขที่แรงปฏิกิริยากับพื้นอยู่ภายในกรวยแรงเสียดทาน มวลรวมของหุ่นยนต์คือ m และ CG อยู่ห่างจากเท้าหุ่นระยะ r โดย r_{min} เป็นระยะน้อยสุดในท่านั่ง และ r_{max} เป็นระยะสูงสุดในท่ายืด หุ่นยนต์มีการควบคุมหลักสองจุดคือ แรงบิดที่ข้อต่อ (q_2) และคำสั่งไปยังมอเตอร์ขาแบบ Series-Elastic ซึ่งต่อกับสปริงบิดผ่านกล่องเกียร์ 25:1 (เกียร์, สปริง, และข้อต่อแสดงใน รูปที่ 7) การควบคุมมอเตอร์ใช้การควบคุม PD เพื่อรักษามุม (q_{3c}) นอกจากนี้ หุ่นยนต์ยังมีแรงขับ Aero Dynamic สองจุดเพื่อช่วยคงทิศทางการเคลื่อนที่ให้อยู่ในระนาบ X-Z



รูปที่ 7 (A) Planar Model of The Robot in Stance Phase. The Foot behaves like a pin joint when the robot is on the ground.

(B) Image of Salto-1P with its components labeled.

2.2.3 Leaning Plant Model

แบบจำลองการเอนของหุ่นยนต์ใช้โมเมนตัมเชิงมุม (L) รอบฐานเท้า ซึ่งเป็นโซ่ของตัวรวมสัญญาณ (Integrators) ทำให้การออกแบบตัวควบคุมเป็นปัญหาการวางตำแหน่งขั้วที่ง่ายขึ้น โดยใช้โมเมนตัมเชิงมุมแบบปรับขนาด ($M = \frac{L}{mrg}$) แทน L

สำหรับ Salto-1P เมื่อ r และ q_3 ถูกล็อก การเคลื่อนที่ของหุ่นยนต์จะเทียบเท่ากับ Reaction Wheel Pendulum (RWP) ที่กำหนดโดยพารามิเตอร์สองตัว ได้แก่ ค่าคงตัวเวลาในการเอียง (T_t) และตัวคูณความเร็วเชิงมุมของล้อควบคุม (G_w) :

$$T_t = \sqrt{\frac{mr^2+I_1+I_2+I_3}{mrg}} \quad G_w = \sqrt{\frac{I_2}{mr^2+I_1+I_2+I_3}}$$

$$\begin{aligned}
T_t &= Time\ constant\ of\ toppling \\
m &= total\ mass \\
r &= Displacement\ from\ CG\ and\ foot\ base \\
I_1 &= Chassis\ Mol \\
I_2 &= Reaction\ Wheel\ Mol \\
I_3 &= Leg\ motor\ rotor\ Mol \\
I_3 &= total\ mass \\
g &= Gravitational\ acceleration \\
G_w &= anglar\ velocity\ gain
\end{aligned}$$

พลศาสตร์การหมุนของหุ่นยนต์อธิบายได้ด้วยสมการการเคลื่อนที่ของ RWP:

$$\begin{aligned}
H_{11}\ddot{\theta} + H_{12}\ddot{q}_2 &= mrgsin(\theta) \\
H_{21}\ddot{\theta} + H_{12}\ddot{q}_2 &= r
\end{aligned} \qquad (1)$$

$H11, H12, H21 = Inertia\ and\ coupling\ terms\ based\ on\ mass\ distribution\ and\ system\ structure.$

$$\begin{aligned}
\theta &= Angle\ related\ to\ the\ pendulum's\ position \\
\ddot{\theta} &= Angular\ acceleration\ of\ the\ angle\ \theta \\
\ddot{q}_2 &= Angular\ acceleration\ related\ to\ a\ second\ rotational\ component \\
m &= total\ mass \\
r &= Displacement\ from\ CG\ and\ foot\ base \\
m &= total\ mass \\
g &= Gravitational\ acceleration
\end{aligned}$$

โดยที่ $H_{11} = mrgT_t{}^2$ และ $H_{12} = H_{21} = H_{22} = -G_w H_{11}$ ที่ RWP สามารถรักษาสมดุลในตำแหน่งที่นิ่งได้

2.2.4 Leaning Control

Leaning Control มีหน้าที่จัดการ Angular Momentum และ Body Angle ของหุ่นยนต์ในช่วงการ Launch และ Landing Stance Phases โดยอิงจากทฤษฎีการทรงตัว ซึ่งได้รับการทดสอบกับ RWP (Reaction Wheel Pendulum) ชื่อ Tippy คอนโทรลเลอร์นี้ถูกดัดแปลงให้ติดตามคำสั่ง Angular Momentum แทนที่จะติดตามตำแหน่งของ Actuated Joint เหมือนในงานก่อนหน้า

คอนโทรลเลอร์ใช้ตัวแปรสถานะ M , $\dot{M}$, และ $\ddot{M}$ ซึ่งสามารถคำนวณได้จากพารามิเตอร์ T_t และ G_w สมการการควบคุมการเอนที่ใช้ Full-State Feedback ถูกกำหนดดังนี้:

$$\dddot{M} = k_{dd}\,\ddot{M} + k_d\dot{M} + k_m\,(M - u) \qquad (2)$$

$$r = Disent\ from\ CG\ and\ foot\ base$$
$$M = angular\ momentum\ or\ the\ system\ state.$$
$$\dot{M} = Angular\ velocity\ of\ M.$$
$$\ddot{M} = Angular\ acceleration\ of\ M.$$
$$\dddot{M} = the\ rate\ of\ change\ of\ angular\ acceleration.$$
$$u = The\ desired\ command\ or\ target\ value\ for\ M.$$
$$k_m = Gain\ for\ the\ value\ of\ M$$
$$k_d = Gain\ for\ the\ value\ of\ \dot{M}$$
$$k_{dd} = Gain\ for\ the\ value\ of\ \ddot{M}$$

โดย u คือคำสั่ง Angular Momentum ที่ต้องการ ในการกำหนดค่า Feedback (k_{dd}, k_d, k_m) ระบบจะอ้างอิงตาม Closed-Loop Poles λ_i เพื่อให้ได้การตอบสนองตามต้องการ

ผลลัพธ์ของคอนโทรลเลอร์การเอนสามารถแปลงเป็นคำสั่ง Torque (τ) สำหรับ Actuated Joint โดยใช้สมการการเคลื่อนที่ของหุ่นยนต์ อย่างไรก็ตาม Balance Controllers นี้ไวต่อการประเมินค่าทิศทางแนวตั้งซึ่งอาจได้รับผลกระทบจากการลอยของเซ็นเซอร์ เช่น IMU เพื่อแก้ไขปัญหานี้ ในช่วงการ Stance จะใช้ Balance Offset Observer เพื่อแก้ไขค่าความเบี่ยงเบนสะสม

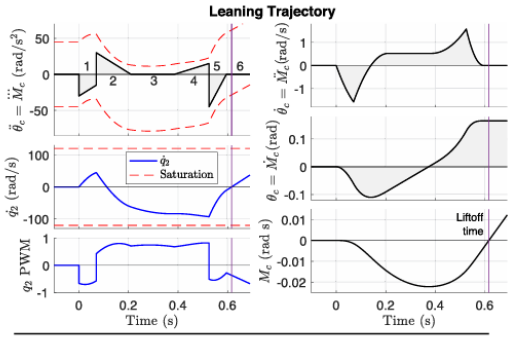
2.2.5 Leaning Trajectory

หุ่นยนต์ต้องทำความเร็วในแนวดิ่ง ($\bar{v}_z$) และแนวนอน ($\bar{v}_x$) ของจุดศูนย์กลางมวล (CG) ในช่วงยกตัว (Liftoff) เพื่อเข้าสู่วิถีโค้งแบบ Ballistic Parabola ซึ่งจะช่วยให้สามารถลงจอดในตำแหน่งที่กำหนดได้ การควบคุมเส้นทางการเคลื่อนที่ถูกตั้งค่าให้มุมเอน (θ) ค่อยๆ เอียงไปถึงมุมเป้าหมาย (θ_c) ขณะที่ขาของหุ่นยนต์ยังคงย่อต่ำสุด ($r = r_{min}$) จากนั้นจะขยายออกจนถึงความเร็วในช่วงยกตัว ($\bar{r}_c$) ทำให้ความเร็วในแนวนอน ($\bar{v}_z = \bar{r}sin\,(\bar{\theta})$) และความเร็วในแนวดิ่ง ($\bar{v}_z = \bar{r}cos\,(\bar{\theta})$) เส้นทางการเคลื่อนที่นี้จะช่วยแยกการควบคุมการเอน และการควบคุมขาให้ทำงานอย่างเป็นอิสระ

เส้นทางการเคลื่อนที่นี้ออกแบบมาเพื่อตอบโจทย์ 4 ข้อหลัก ได้แก่:

- มุมเอนถึงค่าที่ต้องการ ($\bar{\theta} = \bar{\theta}_c$)
- ความเร็วเชิงมุม ($\dot{\bar{\theta}} = \dot{\bar{q}}_2 = 0$)
- โมเมนตัมเชิงมุม ($\bar{L} = 0$)
- มอเตอร์ของล้อควบคุม(q_2) ไม่เข้าสู่สถานะอิ่มตัว

การออกแบบเส้นทางการเคลื่อนที่ ($\theta_c(t) = \dot{M}_c(t)$) ใช้เป็นฟังก์ชัน Piecewise Cubic โดยมีพารามิเตอร์เป็นอัตราเร่งเชิงมุม (a) (rad/s^2) และเวลา $T(s)$ ดังแสดงใน รูปที่ 8 เส้นทางการเคลื่อนที่นี้ถูกเลือกเพราะมีการแก้เชิงวิเคราะห์ง่าย และรองรับพารามิเตอร์ที่หลากหลาย



Segment	Time	$\dddot{M}_c$ initial	$\dddot{M}_c$
1	$0 \le t < T$	$-a$	$0.5a/T$
2	$T \le t < 3T$	a	$-0.5a/T$
3	$3T \le t < 5.5T$	0	0
4	$5.5T \le t < 7.5T$	0	$0.25a/T$
5	$7.5T \le t < 8.5T$	$-1.5a$	$1.5a/T$
6	$8.5T \le t$	0	0

รูปที่ 8 Leaning Trajectory for Launch with Parameters $a = 30 rad/s\ T = 0.07s$. Intended liftoff time is indicated by the purple line.

เส้นทางการเคลื่อนที่นี้คงที่ที่ ($\bar{\theta}_c$) ที่เวลา $8.5T$ วินาที โดยยกตัวเมื่อ ($M = \dot{q}_2 = 0$) ที่เวลา $\frac{317}{36}T$ วินาที หากยกตัวคลาดเคลื่อน อาจส่งผลให้ค่า $\bar{L}$ และ $\dot{\bar{q}}_2$ มีความเบี่ยงเบนจากศูนย์ แต่ $\dot{\bar{\theta}}$ จะยังคงอยู่ที่ $\bar{\theta}_c$

ในการกระโดดไปยังเป้าหมาย Jump Planner จะคำนวณมุมเอน และอัตราเร่งเชิงมุม (a) โดยใช้ตำแหน่งปัจจุบันของเท้า (x, z) และตำแหน่งเป้าหมาย (x_c, z_c) ตามสมการ:

$$\bar{v}_{xc} = \frac{(x_c - x) g \bar{v}_{zc}}{2 r_{max} g + \bar{v}_{zc} \sqrt{\bar{v}_{zc}^{\,2} - 2(z_c - z) g + \bar{v}_{zc}^{\,2}}} \qquad (3)$$

$x_c = Horizontal\ positions, the\ target\ or\ desired\ position.$

$\boldsymbol{x} = Horizontal\ positions, the\ current\ position.$

$g = Gravitational\ acceleration.$

$\bar{v}_{zc} = Vertical\ component\ of\ velocity.$

$r_{max} = Maximum\ Displacement\ from\ CG\ and\ foot\ base.$

$z_c = Vertical\ positions, z_c\ possibly\ being\ the\ target\ or\ desired\ height.$

$z = the\ current\ height.$

$$\bar{\theta}_c = atan\left(\frac{\bar{v}_{xc}}{\bar{v}_{zc}}\right)$$

$\bar{\theta}_c = The\ calculated\ angle, likely\ representing\ the\ angle\ of\ the\ trajectory$
$or\ motion\ direction\ in\ relation\ to\ the\ horizontal\ or\ vertical\ axis.$

$\bar{v}_{xc} = Vertical\ component\ of\ velocity.$

$\bar{v}_{zc} = Vertical\ component\ of\ the\ velocity.$

$$a = \frac{8}{9T^2}\bar{\theta}_c$$

$\bar{\theta}_c = The\ calculated\ angle, likely\ representing\ the\ angle\ of\ the\ trajectory$
$or\ motion\ direction\ in\ relation\ to\ the\ horizontal\ or\ vertical\ axis.$

$\bar{v}_{xc} = This\ could\ represent\ an\ acceleration\ or\ some\ other\ parameter$
$that\ scales\ with\ the\ angle\ \bar{\theta}_c.$

$T = time\ period\ or\ duration.$

นอกจากนี้การทำ Small Angle Approximation จะถูกใช้ในการคำนวณตำแหน่งของ CG ระหว่างยกตัว

2.2.6 Touchdown Plan

การควบคุมในช่วง Flight จะตั้งค่าสภาวะเริ่มต้นสำหรับการลงจอด โดยปรับมุม และความยาวขาในขณะ Touchdown ให้เหมาะสมกับความเร็วตอนลงพื้น เพื่อป้องกันไม่ให้ระบบ Leaning Control สูญเสียสมดุล

1. ความยาวขาในขณะ Touchdown: หลังจากที่หุ่นยนต์ถึงจุดสูงสุดของการกระโดด ขาจะเริ่มยืดออกในขณะที่ตกลงมา ความยาว (r) ควรตั้งค่าให้พอเหมาะเพื่อให้ขาถึง (r_{min}) โดยไม่กระแทกแรงเกินไป ซึ่งจะช่วยให้หุ่นยนต์ทรงตัวได้ดีขึ้น (2.2.7 Series-Elastic Leg Control)
2. มุมขาในขณะ Touchdown: มุมขา (θ) ควรตั้งค่าให้เหมาะสมเพื่อลดความพยายามในการควบคุมหลังการลงจอด โดยมุมที่ตั้งไว้อย่างเหมาะสม(θ_{eff}^{+}) จะช่วยให้หุ่นยนต์ยังคงใกล้กับวิถีที่ต้องการหลังจากการกระแทก ซึ่งเทียบเท่ากับการทำให้มุม Offset เป็นศูนย์หลัง Impact:

$$\theta_{eff}^{+} = \theta + \dot{\theta}T_t = 0 \qquad (4)$$

สมการอนุรักษ์โมเมนตัมเชิงมุมขณะ Touchdown:

$$(mr^2 + I_1)\dot{\theta}^+ = -mr\left(v_x^- sin(\theta) - v_z^- cos(\theta)\right) + I_1\dot{\theta}^- \quad (5)$$

$mr^2 = Moment\ of\ inertia\ term\ involving\ the\ mass\ \ and\ radius.$

$I_2 = Moment\ of\ inertia\ of\ the\ leg\ or\ body\ segment.$

$\dot{\theta} = Angular\ velocity\ of\ the\ leg.$

$v_x\ and\ v_z\ = Components\ of\ the\ velocity.$

$The\ superscripts\ +\ and\ -\ indicate\ values\ immediately\ after\ and\ before\ the\ impact, respectively.$

หลังจากทำให้สมการเรียบง่ายโดยใช้สมมติฐาน และการประมาณมุมเล็ก ค่ามุมที่ต้องการสำหรับการ Touchdown (θ_c) จะเป็น:

$$\theta_c = -\frac{T_t v_x^-}{r - T_t v_z^-} \quad (6)$$

$T_t = A\ time\ constant\ or\ parameter\ related\ to\ the\ touchdown\ phase.$

$r = Radius\ or\ leg\ length\ at\ touchdown.$

$v_x^-\ and\ v_z^-\ = \ Horizontal\ and\ vertical\ velocities\ before\ impact.$

ข้อผิดพลาดในการประเมินความเร็วจะส่งผลต่อมุม Touchdown โดยความผิดพลาดความเร็วแนวนอนสูงสุดที่ยอมรับได้สำหรับหุ่นยนต์ SALTO-1P คือประมาณ 0.20 m/s

2.2.7 Series-Elastic Leg Control

การเคลื่อนที่ของจุดศูนย์กลางมวล (CG) ในแนวแกน (r) ของ Salto-1P มีลักษณะคล้ายกับการเคลื่อนที่ของมวลบนรางเชิงเส้น อย่างไรก็ตาม Salto-1P ใช้การควบคุมแบบ Series-Elastic Power Modulation ที่ไม่เป็นเชิงเส้น โดยมีการควบคุมผ่านตัวควบคุมที่อิงพลังงาน ในช่วงการยกตัว Leg Control จะเพิ่มความเร็วเชิงรัศมีที่ต้องการ ($\bar{r}_c$) โดยการหมุนมอเตอร์ขาไปยังมุมที่ตั้งไว้ (q_{3c}) ซึ่งสัมพันธ์กับ ($\bar{r}_c$) ดังสมการต่อไปนี้:

$$\frac{1}{2}k(\frac{1}{G}(q_{3c} - q_o))^2 = \frac{1}{2}m\bar{r}_c^{\,2} \quad (7)$$

$G = 25: Gear\ ratio.$

$k = \ Spring\ constant.$

$q_o = Initial\ motor\ position\ for\ energy\ balance\ or\ neutral\ position.$

$q_{3c} = command\ or\ target\ position\ for\ the\ motor\ that\ adjusts\ the\ elastic\ force\ applied\ by\ the\ leg.$

$m = total\ mass\ /\ mass\ of\ the\ system.$

$\bar{r}_c = Desired\ leg\ velocity.$

โดยที่ $G = 25$ เป็นอัตราทดของเกียร์, k และ q_o เป็นพารามิเตอร์ของสปริงที่ใช้คำนวณการลดแรงเนื่องจากการส่งผ่านและความไม่เชิงเส้นของสปริง ซึ่งทำให้ได้:

$$q_{3c} = G\sqrt{\frac{m}{k}}\bar{r}_c + q_o \tag{8}$$

สำหรับ SALTO-1P ค่าที่วัดได้จริงคือ $q_{3c} = 17\frac{rad}{s\ m}\bar{r}_c + 18.5\ rad$ โดย SALTO-1P จะเริ่มยกตัวประมาณ 0.14 วินาที หลังจากเปิดใช้งานมอเตอร์ขา เพื่อให้มอเตอร์ทำงานล่วงหน้าก่อนถึงช่วงเวลายกตัวที่ตั้งใจไว้ที่ $\frac{317}{36}\ T = 0.14$ วินาที ในช่วงที่ขาสัมผัสพื้นหลังจากการลอยขึ้น ความยาวขาจะถูกตั้งค่าเป็น:

$$q_{3c} = -10\ v_z + 25 \tag{9}$$

โดยจะเพิ่ม r เมื่อความเร็วในแนวแกน $|v_z|$ เพิ่มขึ้น

ในระหว่างการลงจอด Salto-1P ใช้ Closed-Loop Force Control สำหรับมอเตอร์ขาแบบ Series-Elastic เพื่อให้ระบบทำงานในลักษณะของตัวหน่วง โดยตั้งค่าหน่วงไว้ที่ประมาณ $1.5\ N\ s\ m^{-1}$ เพื่อช่วยลดแรงกระแทกให้เรียบขณะสัมผัสพื้น โดยการบีบอัด r กลับไปยังค่าขั้นต่ำ r_{min}

3. เนื้อหาในรายวิชาที่เกี่ยวข้อง

1) Transformation of Coordinate Frame

2) Forward Kinematics

3) Inverse Kinematics

4) Differential Kinematics

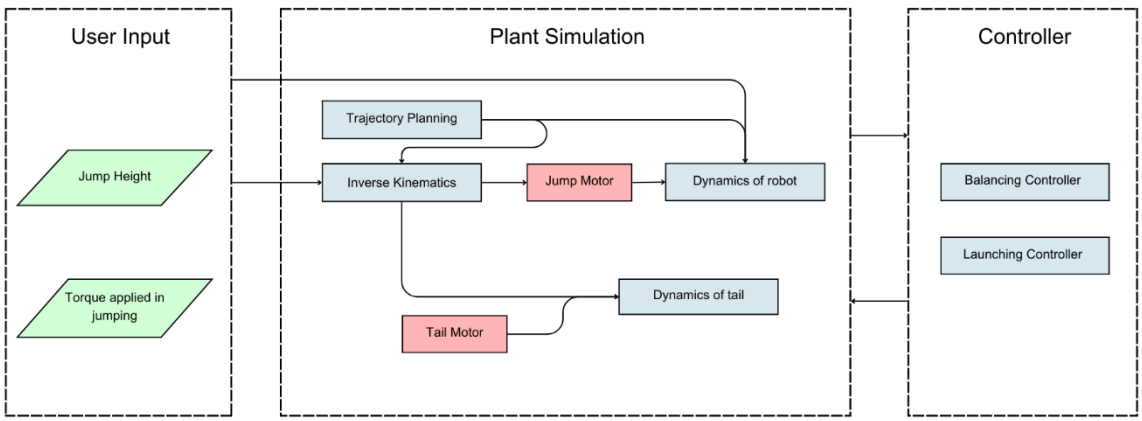
5) Statics Force

6) Dynamics

7) Trajectory Gen

8) การทำ Robot Modelling ผ่าน Simscape Multibody

4. System Diagram / System Overview (Function and Argument)



1. Input

   a. Jump Height: Parameter Input นี้จะกำหนดความสูงที่ User ต้องการ ที่หุ่นยนต์ควรไปถึงระหว่างการกระโดด ข้อมูลนี้จะถูกส่งไปยัง Inverse Kinematics Block ซึ่งจะส่งผลต่อตำแหน่ง และมุมของข้อต่อที่จำเป็นในการกระโดดข้ามความสูงดังกล่าว เพื่อสร้าง Trajectory Plane ที่ Smooth และไปยังความสูงดังกล่าว

   b. Torque Applied in Jumping: Input นี้จะใส่แรงบิดหรือแรงเริ่มต้นที่จะใช้เพื่อเริ่มการกระโดด ช่วยในการกำหนดแรงกระโดดที่จำเป็น และสามารถใช้คำนวณ Dynamics ของหุ่นยนต์ได้ด้วย ค่าแรงบิดนี้จะ Interact กับ Inverse Kinematics เพื่อให้แน่ใจว่าการกระโดดจะดำเนินการด้วยแรงที่เหมาะสม และจะสร้าง Trajectory Plane เพื่อให้แน่ใจว่าแรงที่ใช้สอดคล้องกับเส้นทางการเคลื่อนที่ที่เสถียร และสมดุล

2. Plant Simulation

ในส่วนนี้จำลองการเคลื่อนไหวของหุ่นยนต์ รวมถึงการกระโดด การทรงตัวในอากาศ และการลงจอด แต่ละส่วนประกอบ ใช้ Output ของ Trajectory Plan เพื่อทำตาม Path ที่กำหนดไว้ ทำให้การเคลื่อนไหวแม่นยำ และควบคุมได้มากขึ้น

a) Trajectory Planning: คำนวณเส้นทางการเคลื่อนที่ที่ต้องการสำหรับทั้งช่วงการกระโดด และการทรงตัวของหุ่นยนต์ โดยใช้ Parameters Input (Jump Height or Torque applied in jumping) จะกำหนด Path
ที่เหมาะสมที่สุดสำหรับตำแหน่งของข้อต่อ ความเร็ว และความเร่ง ข้อมูลเส้นทางนี้จะถูกส่งไปยัง Inverse Kinematics Block และ Dynamics of Robot เพื่อเป็น Guide ในการเคลื่อนที่

b) Inverse Kinematics: รับข้อมูล Path จาก Trajectory Planning Block จากนั้นคำนวณตำแหน่งข้อต่อ และมุมตามเส้นทางนี้ ผลลัพธ์ที่ได้คือชุดการกำหนดค่าข้อต่อที่สอดคล้องกับตำแหน่ง และมุมที่วางแผนไว้ของเส้นทาง เพื่อให้แน่ใจว่าการกระโดดจะราบรื่น

c) Dynamics of Robot: ใช้ข้อมูลเส้นทางเพื่อจำลอง Dynamics ของหุ่นยนต์ โดยเฉพาะอย่างยิ่งในช่วงการกระโดด Jump Motor จะถูกควบคุมให้สอดคล้องกับเส้นทางที่วางแผนไว้ โดยจะปรับแรงและแรงบิดเพื่อให้เคลื่อนที่ตามเส้นทางได้อย่างราบรื่น นอกจากนี้ Block นี้ยังรับ Feedback และส่ง Output กับ Balancing Controller Block อีกด้วย

d) Dynamics of Tail: ทำงานร่วมกับเส้นทางที่วางแผนไว้เพื่อความสมดุล Tail Motor จะถูกควบคุมเพื่อรักษาเสถียรภาพตลอดเส้นทาง ทั้งในกลางอากาศ และเมื่อลงจอด Feedback จาก Controller จะถูกนำมาคำนวณ และส่ง Output กลับไปเพื่อการปรับสมดุลจะปรับส่วนหางเพื่อให้แน่ใจว่าสมดุลตามเส้นทางการเคลื่อนที่

3. Controller

a. Balancing Controller: ใช้ Trajectory Data เป็น Reference สำหรับการเคลื่อนไหวที่เสถียร โดยจะดู Feedback จาก Dynamics of Robot and Dynamics of Tail blocks โดยจะปรับ Jump Motor และ Tail Motor เพื่อรักษาสมดุล และ Track Path การเคลื่อนที่ได้อย่างแม่นยำ หากเกิดการเบี่ยงเบน ตัวควบคุมจะแก้ไขท่าทางของหุ่นยนต์เพื่อปรับให้ตรงกับเส้นทางการเคลื่อนที่ที่วางแผนไว้

b. Launching Controller: มีหน้าที่ควบคุมการกระโดดของหุ่นยนต์ ซึ่งรวมถึงการออกแรงที่ข้อต่อ และการตั้งค่าพารามิเตอร์การกระโดดเพื่อให้หุ่นยนต์สามารถกระโดดได้ตามความสูงที่กำหนด โดยคอนโทรลเลอร์นี้จะใช้ข้อมูลจาก Trajectory Planning และ Inverse Kinematics เพื่อกำหนดแรงบิด (Torque) และทิศทางที่ต้องใช้ในขณะกระโดด การควบคุมจะทำงานร่วมกับ Jump Motor เพื่อปรับทอร์คในการกระโดดให้เหมาะสมกับความสูงที่ตั้งไว้ใน User Input นอกจากนี้ ยังต้องประสานงานกับ Dynamics of robot และ Dynamics of tail เพื่อให้การกระโดดเป็นไปอย่างราบรื่น และหุ่นยนต์สามารถรักษาการทรงตัวในอากาศได้

5. ผลการศึกษาที่คาดหวัง

1) สามารถวิเคราะห์ Inverse Kinematics และ Dynamics ของหุ่นยนต์ ได้

2) สามารถวิเคราะห์การเคลื่อนที่ของหุ่นยนต์ที่สามารถกระโดดข้ามสิ่งกีดขวางได้อย่างมีประสิทธิภาพ

3) สามารถพัฒนาระบบควบคุมเพื่อรักษาสมดุลในระหว่างการกระโดด และการลงสู่พื้น

4) สามารถ Simulation การเคลื่อนที่ของหุ่นที่สามารถกระโดดได้

6. รายละเอียดโครงการ

ลำดับ	การดำเนินงาน	สัปดาห์ที่ 1	สัปดาห์ที่ 2	สัปดาห์ที่ 3	สัปดาห์ที่ 4	สัปดาห์ที่ 5	สัปดาห์ที่ 6
1	ศึกษาทฤษฎีเกี่ยวกับการเคลื่อนไหว และการทรงตัวของหุ่นกระโดด						
2	จัดทำ Proposal สำหรับโครงการหุ่นกระโดด						
3	ศึกษา และพัฒนาการคำนวณ Kinematics และ Dynamics ของหุ่นกระโดด						
4	สร้าง และปรับปรุงการ Visualization การเคลื่อนไหวของหุ่นกระโดดในรูปแบบ 2 มิติ						
5	ปรับปรุงการควบคุม และความเสถียรของหุ่นกระโดดขณะลอยตัว และลงพื้น						
6	ปรับแก้ไข สรุปผลการศึกษา และส่งงาน						

7. เอกสารอ้างอิง (References)

[1] Haldane, D. W., Plecnik, M. M., Yim, J., & Fearing, R. S. (2016). Robotic vertical jumping agility via series-elastic power modulation. *Science Robotics, 1*(1), eaag2048. https://doi.org/10.1126/scirobotics.aag2048

[2] Yim, J., Plecnik, M. M., & Fearing, R. S. (2020). Precision jumping limits from flight-phase control in Salto-1P. *IEEE Robotics and Automation Letters, 5*(2), 1287–1294. https://people.eecs.berkeley.edu/~ronf/PAPERS/JYim-RAL20.pdf