

國 立 中 央 大 學

資 訊 工 程 學 系  
碩 士 論 文

使用大型語言模型進行機器控制指令的自動化生  
成

Automated Generation of Machine Control  
Commands Using Large Language Models

研 究 生：蔡時富

指 導 教 授：蘇木春 博 士

中 華 民 國 一 百 一 十 三 年 六 月

# 使用大型語言模型進行機器控制指令的自動化生成

## 摘要

本研究探討了如何透過大型語言模型 (Large Language Model, LLM)，將自然語言轉換為程式碼來控制機器。研究內容涵蓋背景知識、文獻回顧、研究方法、以及實驗設計與結果。

在背景知識和文獻回顧部分，首先介紹了大型語言模型的研究現況、智慧機器與人工智慧物聯網的應用場景，以及 3D 列印技術的發展現況。接著，回顧了大型語言模型在程式碼生成與機器控制上的應用、運動學研究與機器人控制的相關文獻，以及 3D 列印近年來的發展與其應用場景。

研究方法部分描述了硬體設計流程，包括模型設計軟體、檔案輸出格式、3D 列印的使用，以及馬達與開發版的介紹。軟體設計流程方面，介紹了運動模擬環境、順向運動學和逆向運動學的開發，與大型語言模型應用程式介面的使用，最後，系統架構章節詳細介紹了系統架構圖、系統流程圖等整體程式框架。

實驗設計與結果部分包含三個實驗。分別為機械臂的基本控制、機械臂應用於畫圖與機械臂在自動運輸車上的應用，其中展示了機械結構設計圖、函數設計、下達指令的格式、實驗過程縮圖以及最後的實驗成效總結。

而實驗結果顯示，使用大型語言模型生成程式碼來控制機器的方式擁有相當高的準確度，尤其在有較明確的機器函式庫的前提下，更能透

過少量的指令輸入，獲得高品質生成效率和準確度。然而，隨著硬體的增多和系統的複雜性增加，也面臨了一些需要克服的機械性失誤。未來的研究將繼續優化系統的穩定性和精確度，進一步提升其應用價值。

**關鍵字：**自動程式碼生成、機器控制、自然語言、大型語言模型、3D 列印、智慧機器

# Automated Generation of Machine Control Commands Using Large Language Models

## Abstract

This study explores how to convert natural language into code to control machines using a Large Language Model (LLM). The research covers background knowledge, literature review, research methods, and experimental design and results.

In the background knowledge and literature review section, we first introduce the current research status of large language models, the application scenarios of smart machines and the Internet of Things (IoT) in artificial intelligence, and the development status of 3D printing technology. We then review the applications of large language models in code generation and machine control, kinematics research and robot control, as well as the application scenarios of 3D printing in smart machines.

The research methods section describes the hardware design process, including model design software, file output formats, the use of 3D printing, and the introduction of motors and development boards. The kinematics development aspect covers the development of a motion simulation environment, forward kinematics, and inverse kinematics. The section on the selection of large language models includes an introduction to LLM APIs and their applications. Finally, the system architecture section provides a detailed introduction to the system architecture diagram, system flowchart, and the overall program frame-

work.

The experimental design and results section includes three experiments: basic control of the robotic arm, the application of the robotic arm in drawing, and the application of the robotic arm on an automated transport vehicle. This section presents mechanical structure design diagrams, function design, command format examples, and experimental results.

The experimental results show that using large language models to generate code for controlling machines achieves a high level of accuracy. Particularly with a well-defined machine function library, high-quality generation efficiency and accuracy can be achieved with minimal command input. However, as the number of hardware components increases and the system complexity grows, some mechanical errors have been encountered. Future research will continue to optimize the stability and accuracy of the system, further enhancing its application value.

**Keywords:** Auto Generation, Machine Control, Large Language Models, 3D Printing, Smart Machines

# 誌謝

誌謝...

# 目錄

	頁次
<b>摘要</b>	<b>i</b>
<b>Abstract</b>	<b>iii</b>
<b>誌謝</b>	<b>v</b>
<b>目錄</b>	<b>vi</b>
<b>一、 緒論</b>	<b>1</b>
1.1 研究動機 .....	1
1.2 研究目的 .....	2
1.3 論文架構 .....	2
<b>二、 背景知識以及文獻回顧</b>	<b>4</b>
2.1 背景知識 .....	4
2.1.1 大型語言模型的研究現況 .....	4
2.1.2 智慧機器與人工智慧物聯網的應用場景 .....	6
2.1.3 3D 列印技術的發展現況 .....	7
2.2 文獻回顧 .....	8
2.2.1 大型語言模型及其在程式碼生成與機器控制上的應用 .....	8
2.2.2 運動學研究與機器人控制 .....	9
2.2.3 3D 列印應用於機器人製作的相關文獻 .....	14

<b>三、 研究方法</b>	<b>15</b>
3.1 硬體設計流程 .....	15
3.1.1 模型設計軟體：Autodesk Fusion 360.....	15
3.1.2 檔案輸出格式：STL (Stereolithography) .....	16
3.1.3 3D 列印機：Creality K1 MAX .....	17
3.1.4 馬達與開發版介紹 .....	19
3.2 運動學開發 .....	22
3.2.1 運動模擬環境 .....	22
3.2.2 順向運動學 .....	23
3.2.3 逆向運動學 .....	23
3.3 大型語言模型開發 .....	23
3.3.1 OpenAI 與 GPT 模型 .....	23
3.3.2 模型調用流程 .....	24
3.4 系統架構 .....	25
3.4.1 系統架構與流程 .....	25
<b>四、 實驗設計與結果</b>	<b>27</b>
4.1 實驗一：機械臂的基本控制 .....	27
4.1.1 機械結構設計圖 .....	27
4.1.2 函數設計 .....	30
4.1.3 下達指令的格式範例 .....	31
4.1.4 實驗結果 .....	34
4.2 實驗二：將機械臂用於畫圖 .....	36
4.2.1 機械結構設計圖 .....	36
4.2.2 函數設計 .....	39
4.2.3 下達指令的格式範例 .....	39
4.2.4 實驗結果 .....	41

4.3 實驗三：機械臂在自動運輸車上的應用 .....	44
4.3.1 機械結構設計圖 .....	44
4.3.2 函數設計 .....	47
4.3.3 下達指令的格式範例 .....	48
4.3.4 實驗結果 .....	50
<b>五、總結</b>	<b>52</b>
5.1 未來展望 .....	52
<b>參考文獻</b>	<b>53</b>

# 圖 目 錄

## 頁次

2.1 順向運動學示例圖 . . . . .	10
2.2 幾何求解逆向運動學示例圖 . . . . .	12
3.1 Autodesk Fusion 360 Logo . . . . .	16
3.2 3D Systems Logo . . . . .	17
3.3 Creality K1 MAX . . . . .	18
3.4 開發板與馬達縮圖 . . . . .	21
3.5 系統架構圖 . . . . .	26
4.1 機械臂版本一設計圖紙第一頁 (單位:mm) . . . . .	27
4.2 機械臂版本一設計圖紙第二頁 (單位:mm) . . . . .	28
4.3 機械臂版本一設計圖紙第三頁 (單位:mm) . . . . .	28
4.4 機械臂版本一設計圖紙第四頁 (單位:mm) . . . . .	29
4.5 機械臂版本一設計圖紙第五頁 (單位:mm) . . . . .	29
4.6 實驗一：實驗過程縮圖 . . . . .	35
4.7 機械臂版本二設計圖紙第一頁 (單位:mm) . . . . .	36
4.8 機械臂版本二設計圖紙第二頁 (單位:mm) . . . . .	37
4.9 機械臂版本二設計圖紙第三頁 (單位:mm) . . . . .	37
4.10 機械臂版本二設計圖紙第四頁 (單位:mm) . . . . .	38
4.11 機械臂版本二設計圖紙第五頁 (單位:mm) . . . . .	38
4.12 實驗二：實驗過程縮圖 . . . . .	43

4.13 機械臂版本三設計圖紙第一頁 (單位:mm) . . . . .	44
4.14 機械臂版本三設計圖紙第二頁 (單位:mm) . . . . .	45
4.15 機械臂版本三設計圖紙第三頁 (單位:mm) . . . . .	45
4.16 機械臂版本三設計圖紙第四頁 (單位:mm) . . . . .	46
4.17 機械臂版本三設計圖紙第五頁 (單位:mm) . . . . .	46
4.18 機械臂版本三設計圖紙第六頁 (單位:mm) . . . . .	47
4.19 實驗三：實驗過程縮圖 . . . . .	51

# 表 目 錄

頁次

4.1 實驗結果：Please catch the ball to the top of the tower. . . . .	34
4.2 實驗結果：Draw a square. . . . .	41
4.3 實驗結果：Draw a triangle. . . . .	41
4.4 實驗結果：Draw a star. . . . .	41
4.5 實驗結果：Draw a house. . . . .	42
4.6 實驗結果：Please grab the red block, and then grab the blue block. . . . .	50

# 一、緒論

## 1.1 研究動機

在現今蓬勃發展的人工智慧領域中，隨著高效能運算技術 (High Performance Computing, HPC) [1] 的快速發展，許多大型語言模型 (Large Language Model, LLM) [2]，如 OpenAI 的 Chat-GPT [3]、Meta 的 Llama2 [4]、Google 的 Gemini [5] 的問世，至今已經深刻改變了我們對於人工智慧與其應用前景的認知。這些大型模型的出現不僅擴大了我們對於人工智慧應用的想像，同時突破了人工智慧技術在各個不同領域中應用的可能，目前大型語言模型已被廣泛的應用在客服、教育、編輯寫作、程式開發與多媒體創作領域，成為了人們生活與工作中的一大幫助。此外，對於這些模型的需求不斷增長，也促使了各大科技公司不斷投入資源，提供更加強大且多樣化的大型語言模型，使的目前的人工智慧發展方向更加明朗。

本研究旨在探索大型語言模型在自動程式碼生成 (Code Generation) [6] 領域的能力，特別是在控制機器方面的應用。眾所周知，目前許多研究集中於微調模型本身，以提升其在單一任務上的效能。然而我們認為，將大型語言模型應用於控制機器，是一個極具挑戰性且有前景的研究方向。通過將人類的自然語言指令轉換為程式碼，機器可以更加靈活地被控制，進一步擴大了人與機器之間的交互性，並使得使用者能夠以更加直觀和自然的方式與機器進行互動。

這項研究的重要性在於其對於智慧機器技術發展和應用的潛在影響。

首先，此研究不但可以大幅降低機器在軟體層面的開發與維護成本，從而推動智慧機器技術的普及和應用。更重要的是，利用大型語言模型能很好的理解自然語言指令的特性，我們可以製作出更加貼近個人需求，且互動性更高的智慧機器，從而滿足不同使用者的個人化需求，並將這些機器廣泛應用於人的生活和工作中，為人們帶來更大的便利和效益。

## 1.2 研究目的

本研究旨在探索大型語言模型生成程式碼的潛力，與在控制機器方面的應用。本研究預計達成以下目標：

- 探索大型語言模型在自動程式碼生成中的應用，並達到使用自然語言控制機器的目的。
- 驗證使用大型語言模型將自然語言指令轉換為程式碼的可行性和實際應用效果。
- 自行組裝設計機器，並透過 3D 列印技術實現更為客製化的機型製作。
- 分析和評估所開發的系統在不同應用場景下的實際效果和應用價值。
- 提供相關技術和方法的研究成果，為自然語言控制機器的相關研究和應用提供實證基礎和技術範本。

## 1.3 論文架構

本論文分為五個章節，其架構如下：

第一章、緒論，敘述本論文之研究目的、動機以及架構。

第二章、背景知識以及文獻回顧，敘述本研究之背景知識如大型語言模型的研究現況、智慧機器與人工智慧物聯網的應用場景、以及融合3D列印技術的機器設計，並探討目前已有的相關研究。

第三章、研究方法，說明本研究細節，如模型設計圖、開發板與硬體的使用、機器運動學開發、與大型語言模型的連動等。

第四章、實驗設計與結果，展示機器的實際運作結果，產生的程式碼與準確率等資訊。

第五章、總結，對於研究結果進行總結，並討論研究的未來展望。

## 二、背景知識以及文獻回顧

### 2.1 背景知識

#### 2.1.1 大型語言模型的研究現況

在自然語言處理（Natural Language Processing, NLP）[7]領域，大型語言模型的發展經歷了多個重要階段，這些階段的發展對於自然語言處理技術的進步和應用具有深遠的影響。以下是一些與自然語言處理相關的里程碑：

- 早期階段：統計模型為當時主要的方法，但這些模型在處理複雜的語言時存在一定的局限性，特別是對於語境理解和生成的能力有限。
- 神經語言模型（Neural Language Model, NLM）[8]：神經語言模型的出現標誌著自然語言處理技術轉為使用類神經網路作為模型架構，但由於當時硬體計算能力和儲存空間的限制，這些模型的規模和性能十分有限。
- 變換器模型 (Transformer Model) [9]：變換器模型的提出是自然語言處理領域的一個重要突破，它引入了自注意機制 (Self-Attention)，使模機能夠更好的捕捉序列中的長程依賴性，從而取得了顯著的性能提升。

- BERT ( Bidirectional Encoder Representations from Transformers ) [10] :  
BERT 是一個雙向的 Transformer 模型，通過預訓練和微調，在多項自然語言處理任務上表現卓越。
- 大型語言模型 (Large Language Model, LLM) : 在過去的幾年中，大型語言模型取得了巨大的進展，這些模型擅長理解和生成類人語言。

大型語言模型的發展是自然語言處理領域的一項重要突破，其基於大量文本數據（例如維基百科、網頁文本）進行無監督預訓練。透過預測下一個詞或填充遮罩的詞來學習語言結構，大型語言模型通常由多個變換器模型層疊加而成。這些變換器層不僅有助於模型學習不同的抽象詞彙，同時也借助自注意力機制，使其能夠有效地處理長文本序列。

最近，多模態模型 (Large Multimodal Models, LMM) 的出現使得現今的人工智慧技術不僅能夠處理文本資訊，還可以處理如圖像、音訊、影片等多種模式。這樣的多模態整合不僅能更全面地理解和生成內容，更大幅增加了人工智能在現實生活中應用場景，如自動駕駛 [11]、藝術創作等 [12]。而一些多模態模型也由於同時使用文本和圖像資料進行了聯合訓練，獲得了比大型語言模型更好的成效。

未來，大型語言模型的發展將面臨許多挑戰。其中包括模型的更好可解釋性，即如何讓模型的決策過程更加透明和可理解；更好的預訓練策略，以提高模型的效能和泛化能力；多模態整合的深入研究，以進一步拓展模型的應用範圍；以及更大模型規模的探索，以應對日益複雜的語言任務和多模態任務需求。

總之，大型語言模型的未來發展將在自然語言處理、多模態應用和人工智能的其他領域中發揮重要作用，並將為人類社會帶來更加便利的服務與體驗。

### 2.1.2 智慧機器與人工智慧物聯網的應用場景

人工智慧物聯網（Artificial Intelligence of Things, AIOT）[13] 技術目前已廣泛應用於許多領域，以下是一些典型的應用範例：

- 工業 4.0(Industry 4.0) [14]：工業領域是人工智慧物聯網技術應用的主要領域之一。機械臂在製造業中的應用越來越普遍。它們可以用於組裝、焊接、物料處理等高重複工作。人工智慧物聯網技術是智慧工廠實現生產過程的自動化和智能化的關鍵，這有助於提高生產效率並減少人力成本，從而提高了整個製造業的競爭力。
- 物流和倉儲：物流和倉儲管理是另一個智慧機器和人工智慧物聯網技術的重要應用場景。無人搬運車 (Automated Guided Vehicle, AGV) [15] 在物流和倉儲管理中扮演著關鍵角色。它們可以自動運送物品，根據預定路徑進行導航，不僅能夠實現倉庫內物品的快速移動和準確分配，同時也降低了人力成本，尤其是在大型倉庫和物流中心，對於智慧機器的需求更加顯著。
- 醫療保健 [16]：智慧機器和人工智慧物聯網技術在醫療保健領域也有著重要的應用。手術機器人是其中一個典型的例子，它可以幫助外科醫生不受時間與地點的影響進行精確的手術操作，同時兼顧了手術的準確性、安全性與急迫性三個重要需求。此外，智慧機器人還可以應用於康復治療和輔助生活等方面，給予病患在進行康復訓練時能獲得更為良好的體驗。

綜上所述，人工智慧物聯網技術現今已廣泛應用於工業自動化、物流倉儲管理以及醫療保健等領域，不僅提高了生產效能，同時也改善了人們的生活品質和健康狀態。隨著技術的不斷發展和創新，人工智慧物聯網技術將繼續在各個領域發揮著重要作用，推動產業往智慧化、自動化的方向發展。

### 2.1.3 3D 列印技術的發展現況

受惠於電腦輔助設計（Computer Aided Design, CAD）[17]、電腦輔助製造（Computer-Aided Manufacturing, CAM）[18] 和電腦數值控制加工（Computer Numerical Control, CNC）[19] 等技術的蓬勃發展。使快速成形技術 [20] (Rapid Prototyping, RP) 在現在的工業設計與製造的研究領域中，已越來越成熟。此技術時常用於快速生成零件模型的製造技術，它通過電腦控制，將材料進行堆疊加工，生成立體實品，因此又稱為積層製造技術（Additive Manufacturing, AM）[21]，而 3D 列印 (3 Dimensional Printing, , 3DP) [22] 則是此技術的一種具體應用。以下為 3D 列印技術的大致發展歷程：

- 早期發展：在 1981 年，日本名古屋市工業研究所的小玉秀男提出了兩種利用光固化高分子的 3D 列印方法。隨後，美國 3D 系統公司的 Chuck Hull 發展了一套快速成形系統，稱為立體快速成形 (Stereolithography, SLA)，利用紫外線雷射來固化光聚合物。
- 商業化時期：1986 年，3D 系統公司在美國通過了第一個快速成形設備的專利，並在 1988 年開發出第一台商業化的快速成形系統。1989 年，美國麻省理工學院申請了第一個 3D 列印的專利技術。早期的技術專利主要掌握在大型快速成形系統公司手中，但直到 2010 年，這些專利陸續到期，3D 列印的應用開始蓬勃發展。
- 應用發展時期：從 2010 年開始，3D 列印的市場產值逐年提升，應用層面也愈加廣泛，現今的 3D 列印技術已被廣泛應用於建築、工業設計、汽車、航太、醫療生技、服飾、飾品、地理資訊和食品等產業。一些劃時代的成品包括 3D 列印的房屋 [23]、無人機 [24]、汽車 [25]、人工血管 [26] 和各式食品 [27] 都已在近年陸續問世。

3D 列印作為快速成形技術技術的一種具體應用，經歷了從早期發展到

商業化時期再到應用發展時期的演進過程。從 1980 年代的技術創新到 2010 年後的市場爆發，3D 列印直到現在依然在不斷擴展應用範圍，隨著技術的不斷發展，3D 列印將持續在各個領域中發揮重要作用，為各行各業帶來更多創新與可能性。

## 2.2 文獻回顧

### 2.2.1 大型語言模型及其在程式碼生成與機器控制上的應用

Vaithilingam 等人 [28] 為了探討大型語言模型作為程式碼生成工具的可用性，招募了 24 名擁有不同程式設計經驗的參與者，使用 Copilot，這個基於大型語言模型的程式碼生成工具，執行一系列不同難度的程式設計任務，而研究人員則從旁紀錄每個參與者的操作、用時與使用 Copilot 的心得。而研究發現，大多數參與者都喜歡在進行程式設計時使用 Copilot，因為它能夠提供一個有用的起點，讓使用者能節省許多思考與搜索的時間，能直接使用 Copilot 紿予的程式為基礎繼續進行修改，然而，有些參與者也同時在理解、編輯 Copilot 生成的程式碼片段時遇到了困難，導致拖延了任務完成的進度。此研究最後總結了一些關於這類程式碼生成工具的改進方向，鼓勵使用者不應簡單的將 Copilot 視為一個能一步到位程式碼產生工具，而應該為理解和驗證生成的程式碼，進而去探索更多解決方案和任務解方。

Sai 等人 [29] 探討了 ChatGPT 在機器人應用中的實驗研究，並提出了一種策略，結合了提示工程的設計原則和機器控制函式，使 ChatGPT 能透過與人對話、解析 XML 標記以及合成程式碼，完成機器人領域的一系列任務，其中例如導航、控制無人機與機械臂，研究重點在於不同的對話策略，在執行各種類型的機器人任務時，是否有成效上的差別。其中還介紹了一個名為 PromptCraft 的開源研究工具，其中包含一個平臺，研究人員可以共同上傳並投票選出最優的對話策略，以及一個集成

了 ChatGPT 的機器人模擬器，讓使用者更容易進行 ChatGPT 與機器人之間的相關研究。

Liang 等人[30]的研究提出了名為「程式即為策略」(Code as Policies)的方法，使用大型語言模型直接生成可以作為控制機器人策略的程式。實驗結果表明，使用大型語言模型生成的程式碼，在某些實驗中的確可以成功地控制機器人執行各種指令，包括某些新的、且沒見過的指令，不過隨著任務難度的複雜化，實驗中隨之降低的準確率也展示出了，現今大型語言模型在語意推理和空間幾何推理領域的發揮是有限的。不過，這篇論文提出的方法仍為機器人控制領域帶來了新的思路，也展示了機器控制與大型語言模型結合後的技術前景。

由文獻可以得知，大型語言模型在機器控制領域與其他眾多領域擁許多可能性。

### 2.2.2 運動學研究與機器人控制

機器人與運動學的研究早已行之有年，以下使用 [31] 提供的定義對本問題做一個簡單解釋：

- 順向運動學 (Forward Kinematics)：順向運動學是一個關鍵概念，它使我們能透過關節角度計算出機械臂末端的位置。這對於控制機械手臂非常重要，因為它使我們能透過馬達角度確定機械臂的每個關節點的空間座標。
- 逆向運動學 (Inverse Kinematics)：逆向運動學是正向運動學的反向問題。它試圖根據機械臂末端的位置，計算出每個關節應該轉動的角度。這是一個更困難的問題，因為它涉及到解方程組或使用數值方法來找到關節角度。逆向運動學在機械手臂的路徑規劃和避障中至關重要。

在順向運動學中，Denavit-Hartenberg 為最常用的計算方式。此方法

使用四個參數來計算力臂頂點座標：

- $a_i$ ：力臂長度 (Link length)，指在第  $i$  個力臂上，從第  $i$  個關節的旋轉軸到第  $i + 1$  個關節的旋轉軸之間的距離，此參數沿著第  $i$  個力臂的  $Z_i$  軸測量。
- $\alpha_i$ ：扭轉角 (Twist Angle)，兩個力臂的連接處稱為節點，指從第  $i$  個節點到第  $i + 1$  個節點之間的扭轉角度，此參數沿著第  $i$  個節點  $X_i$  軸測量。
- $d_i$ ：節點偏移 (Link offset)，指從第  $i$  個節點到第  $i + 1$  個節點之間的距離，此參數通常是變量。
- $\theta_i$ ：節點角度 (Joint angle)，指第  $i$  個旋轉軸的旋轉角度，此參數通常是變量。

每個關節都使用一個獨立座標系來決定這些參數。

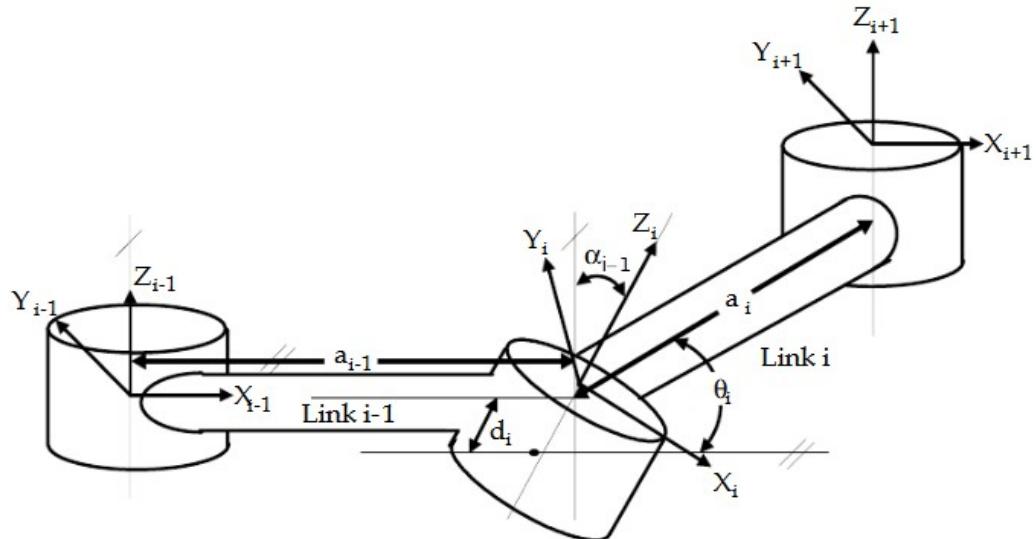


圖 2.1: 順向運動學示例圖

如上圖所示，沿  $X_{i-1}$  測得的  $Z_{i-1}$  到  $Z_i$  的距離記  $a_{i-1}$ ，沿  $X_i$  測得的  $Z_{i-1}$  和  $Z_i$  之間的角度為  $\alpha_{i-1}$ ，沿  $Z_i$  測得的  $X_{i-1}$  到  $X_i$  的距離記  $d_i$ ，沿

$Z_i$  測得的  $X_{i-1}$  到  $X_i$  的角度記  $\theta_i$ 。

如此可求得變換矩陣  ${}_{i-1}^i T$ ：

$$\begin{aligned}
 {}_{i-1}^i T &= R_x(\alpha_{i-1}) D_x(a_{i-1}) R_z(\theta_i) Q_i(d_i) \\
 &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha_{i-1} & -\sin \alpha_{i-1} & 0 \\ 0 & \sin \alpha_{i-1} & \cos \alpha_{i-1} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_{i-1} \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & 0 \\ \sin \theta_i & \cos \theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 &= \begin{bmatrix} \cos \theta_i & -\sin \theta_i & 0 & a_{i-1} \\ \sin \theta_i \cos \alpha_{i-1} & \cos \theta_i \cos \alpha_{i-1} & -\sin \alpha_{i-1} & -\sin \alpha_{i-1} d_i \\ \sin \theta_i \sin \alpha_{i-1} & \cos \theta_i \sin \alpha_{i-1} & \cos \alpha_{i-1} & \cos \alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}
 \end{aligned}$$

而逆向運動學問題，實驗中使用了幾何求解 (Geometric solution approach)，與策略梯度方法 (Policy-Gradient solution approach) [32] 來開發機械臂的逆向控制。

幾何求解方法的原理是將機械手臂的空間幾何問題分解為多個平面幾何問題，適用於機械結構較為簡單的機器人，如下圖所示：

可由以下算式機算頂點  $P$  的分量  $(P_x, P_y)$ 。

$$p_x = l_1 \cos \theta_1 + l_2 \cos \theta_{12}$$

$$p_y = l_1 \sin \theta_1 + l_2 \sin \theta_{12}$$

由於， $\cos \theta_{12} = \cos \theta_1 \cos \theta_2 - \sin \theta_1 \sin \theta_2$ ， $\sin \theta_{12} = \sin \theta_1 \cos \theta_2 - \cos \theta_1 \sin \theta_2$ ，所以可得：

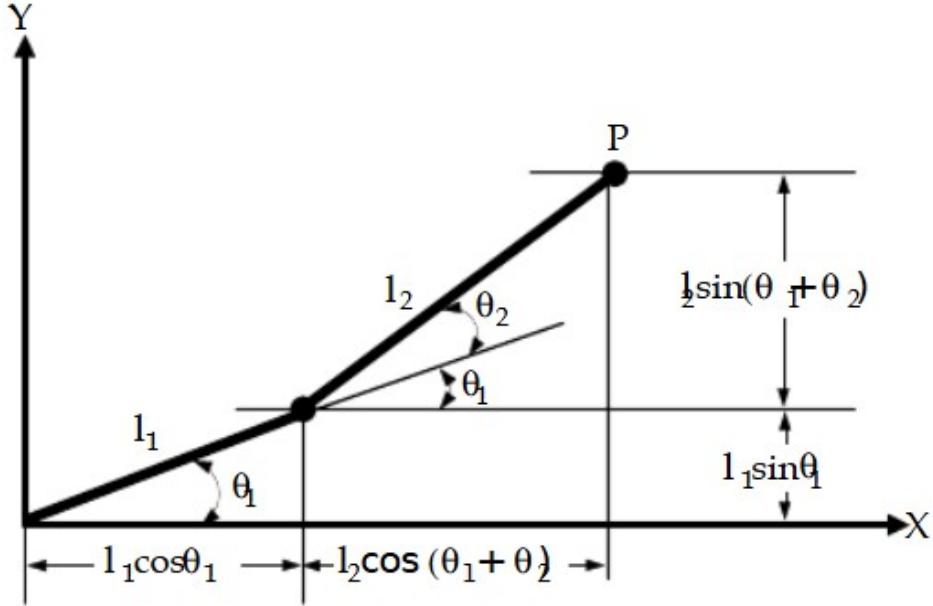


圖 2.2: 幾何求解逆向運動學示例圖

$$p_x^2 = l_1^2 \cos^2 \theta_1 + l_2^2 \cos^2 \theta_{12} + 2l_1 l_2 \cos \theta_1 \cos \theta_{12}$$

$$p_y^2 = l_1^2 \sin^2 \theta_1 + l_2^2 \sin^2 \theta_{12} + 2l_1 l_2 \sin \theta_1 \sin \theta_{12}$$

$$p_x^2 + p_y^2 = l_1^2 (\cos^2 \theta_1 + \sin^2 \theta_1) + l_2^2 (\cos^2 \theta_{12} + \sin^2 \theta_{12}) + 2l_1 l_2 (\cos \theta_1 \cos \theta_{12} + \sin \theta_1 \sin \theta_{12})$$

由於  $\cos^2 \theta_i + \sin^2 \theta_i = 1$ ，故可將上式化簡為：

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1 l_2 (\cos \theta_1 [\cos \theta_1 \cos \theta_2 - \sin \theta_1 \sin \theta_2] + \sin \theta_1 [\sin \theta_1 \cos \theta_2 + \cos \theta_1 \sin \theta_2])$$

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1 l_2 (\cos^2 \theta_1 \cos \theta_2 - \cos \theta_1 \sin \theta_1 \sin \theta_2 + \sin^2 \theta_1 \cos \theta_2 + \cos \theta_1 \sin \theta_1 \sin \theta_2)$$

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1 l_2 (\cos \theta_2 [\cos^2 \theta_1 + \sin^2 \theta_1])$$

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1 l_2 \cos \theta_2$$

由移項後得：

$$\begin{aligned}\cos \theta_2 &= \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2} \\ \sin \theta_2 &= \pm \sqrt{1 - \left( \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2} \right)^2} \\ \theta_2 &= A \tan 2 \left( \pm \sqrt{1 - \left( \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2} \right)^2}, \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1l_2} \right)\end{aligned}$$

繼續利用  $\theta_2$  求解  $\theta_1$ ：將  $p_x = l_1 \cos \theta_1 + l_2 \cos \theta_{12}$  等號兩邊同乘  $\cos \theta_1$   
將  $p_y = l_1 \sin \theta_1 + l_2 \sin \theta_{12}$  等號兩邊同乘  $\sin \theta_1$  得：

$$\begin{aligned}\cos \theta_1 p_x &= l_1 \cos^2 \theta_1 + l_2 \cos^2 \theta_1 \cos \theta_2 - l_2 \cos \theta_1 \sin \theta_1 \sin \theta_2 \\ \sin p_y &= l_1 \sin^2 \theta_1 + l_2 \sin^2 \theta_1 \cos \theta_2 + l_2 \sin \theta_1 \cos \theta_1 \sin \theta_2 \\ \cos \theta_1 p_x + \sin \theta_1 p_y &= l_1 (\cos^2 \theta_1 + \sin^2 \theta_1) + l_2 \theta_2 (\cos \theta_1 + \sin^2 \theta_1)\end{aligned}$$

可化簡為：

$$\cos \theta_1 p_x + \sin \theta_1 p_y = l_1 + l_2 \cos \theta_2$$

將  $p_x = l_1 \cos \theta_1 + l_2 \cos \theta_{12}$  等號兩邊同乘  $-\sin \theta_1$  將  $p_y = l_1 \sin \theta_1 + l_2 \sin \theta_{12}$  等號兩邊同乘  $\cos \theta_1$  得：

$$\begin{aligned}-\sin \theta_1 p_x &= -l_1 \sin \theta_1 \cos \theta_1 - l_2 \sin \theta_1 \cos \theta_1 \cos \theta_2 + l_1 \sin^2 \theta_1 \sin \theta_2 \\ \cos \theta_1 p_y &= l_1 \sin \theta_1 \cos \theta_1 + l_2 \cos \theta_1 \sin \theta_1 \cos \theta_2 + l_2 \cos^2 \theta_1 \sin \theta_2 \\ -\sin \theta_1 p_x + \cos \theta_1 p_y &= l_2 \sin \theta_2 (\cos^2 \theta_1 + \sin \theta_1)\end{aligned}$$

可化簡為：

$$-\sin \theta_1 p_x + \cos \theta_1 p_y = l_2 \sin \theta_2$$

最後，將  $\cos \theta_1 p_x + \sin \theta_1 p_y = l_1 + l_2 \cos \theta_2$  等號兩邊同乘  $P_x$ ，  
 $-\sin \theta_1 p_x + \cos \theta_1 p_y = l_2 \sin \theta_2$  等號兩邊同乘  $P_y$ ，得：

$$\begin{aligned}\cos \theta_1 p_x^2 + \sin \theta_1 p_x p_y &= p_x (l_1 + l_2 \cos \theta_2) \\ -\sin \theta_1 p_x p_y + \cos \theta_1 p_y^2 &= p_y l_2 \sin \theta_2 \\ \cos(p_x^2 + p_y^2) &= p_x (l_1 + l_2 \cos \theta_2) - p_y l_2 \sin \theta_2\end{aligned}$$

可求出  $\theta_1$  :

$$\begin{aligned}\cos \theta_1 &= \frac{p_x (l_1 + l_2 \cos \theta_2) + p_y l_2 \sin \theta_2}{p_x^2 + p_y^2} \\ \sin \theta_1 &= \pm \sqrt{1 - \left( \frac{p_x (l_1 + l_2 \cos \theta_2) + p_y l_2 \sin \theta_2}{p_x^2 + p_y^2} \right)^2} \\ \theta_1 &= A \tan 2 \left( \pm \sqrt{1 - \left( \frac{p_x (l_1 + l_2 \cos \theta_2) + p_y l_2 \sin \theta_2}{p_x^2 + p_y^2} \right)^2}, \frac{p_x (l_1 + l_2 \cos \theta_2) + p_y l_2 \sin \theta_2}{p_x^2 + p_y^2} \right)\end{aligned}$$

以上為使用幾何求解逆向運動學的簡易範例，可以看出使用此方法求解需要繁瑣的計算。

以下介紹使用策略梯度方法求解逆向運動學的計算方式：

### 2.2.3 3D 列印應用於機器人製作的相關文獻

引用 carbon design 的相關論文並進一步討論 (用於設計 3D 模型)

# 三、研究方法

## 3.1 硬體設計流程

### 3.1.1 模型設計軟體：**Autodesk Fusion 360**

Autodesk Fusion 360 是一款集合了電腦輔助設計 (Computer-Aided Design, CAD)、電腦輔助製造 (Computer-Aided Manufacturing, CAM)、電腦輔助工程 (Computer-Aided Engineering, CAE) 及印刷電路板設計 (Printed Circuit Board, PCB) 的多功能設計軟體。由於結合了眾多工具，所以在產品設計、工程設計、機械設計和製造等眾多領域都累積了一定規模的使用者，也成為許多設計師、工程師和製造專業人士的首選工具。

- 3D 建模：提供精準的參數化建模，方便使用者精準的設計每一個零件的細節。
- 裝配設計：模擬多個零件組合後的裝配設計，能確認組合後的零件是否能正常運作。
- 渲染與設計圖生成：內建強大的渲染引擎，可以生成高質量的圖像和客製化設計圖，以便更好的展示設計成果。
- 模型庫：官方與使用者社群建立的廣大模型庫，能找到市面上多數的電路板、馬達與零件模型，方便使用者將這些零件一起納入設計圖。

- 雲端協作：Autodesk 擁有自己的雲端平台，團隊成員可以實時共享和協作設計，方便遠端工作和版本控制。



圖 3.1: Autodesk Fusion 360 Logo

### 3.1.2 檔案輸出格式：STL（Stereolithography）

STL（Stereolithography）是一種常用的 3D 模型文件格式，特別是在 3D 列印領域中廣泛應用。STL 格式最早由 3D Systems 公司在 1987 年為其立體光刻（SLA）3D 列印技術開發，但隨後成為各類 3D 列印技術的標準文件格式。STL 文件描述了 3D 物體的表面幾何形狀，不包含任何顏色、材質或其他屬性。以下為使用 STL 格式用於 3D 列印的優勢：

- 簡單性：STL 格式結構簡單，易於生成和處理。
- 廣泛性：幾乎所有的 3D 建模和 3D 列印軟體都支援 STL 檔案格式。
- 精度彈性：可以通過調整三角形數量來控制模型的精度和細節，適應不同的列印需求。
- 輕量：STL 格式僅描述幾何形狀，不包含顏色、材質或其他屬性資訊，正好符合 3D 列印只需幾何形狀的需求。



圖 3.2: 3D Systems Logo

### 3.1.3 3D 列印機：Creality K1 MAX

本實驗選擇使用的 3D 列印機為 Creality K1 MAX，以下為該機型的主要特點：

- 大尺寸打印：使用 CoreXY 運動結構，高達 300x300x300mm 的列印面積，能夠滿足大型模型的列印需求。
- 解析度：具有極高的打印解析度，可以實現細膩的表面細節和精確的尺寸控制。
- 高速列印：配備了高效的運動系統和強大的驅動機構，最高能達到 600mm/s 的高速列印、 $20000\text{mm/s}^2$  的加速度與  $32\text{mm}^3/\text{s}$  的超大流量，能夠顯著縮短列印時間。
- 耗材多樣性：支援多種列印耗材，包括 PLA、ABS、TPU 等，為使用者提供了更多創作自由度。



圖 3.3: Creality K1 MAX

### 3.1.4 馬達與開發版介紹

以下為本論文使用到的零件簡介：

1. SG90，是一款微型伺服馬達，由於扭矩較小，所以本實驗將其用於機械臂後段(操作點)。其特點包括：

- 重量：小巧輕便，重量僅為 9 克。
- 操作電壓：4.8V 至 6.0V。
- 扭矩：在 4.8V 時可達 1.8kg/cm。
- 控制方式：使用 PWM 信號來控制轉角，通常範圍為 0° 至 180°。

2. MG90S，是一款金屬齒輪伺服馬達，由於扭矩適中，所以本實驗將其用於機械臂前中段。其特點包括：

- 重量：重量約為 14g，稍大於 SG90，但仍屬於微型伺服馬達。
- 操作電壓：4.8V 至 6.0V。
- 扭矩：在 4.8V 時可達 2.2kg/cm。
- 控制方式：使用 PWM 信號來控制轉角，通常範圍為 0° 至 180°。

3. MG996R，是一款高扭矩伺服馬達，由於扭矩較大，所以本實驗將其用於機械臂前段。其特點包括：

- 尺寸：重量約為 55g，適用於中型或大型專案。
- 操作電壓：4.8V 至 7.2V。
- 扭矩：在 6.0V 時可達 9.4kg/cm。
- 控制方式：使用 PWM 信號來控制轉角，通常範圍為 0° 至 180°。

4. Raspberry Pi 4，是一款高性能的嵌入式裝置。其特點包括：

- 處理器：四核 ARM Cortex-A72，1.5GHz。
- 內存：有 2GB、4GB 和 8GB 可選，本實驗採用 8GB 內存。
- 輸入/輸出：支援 USB 3.0、HDMI、GPIO、以太網等連接方式，同時也支援 wifi 藍芽等無線連接方式
- 作業系統：本實驗採用 Raspberry Pi OS（基於 Debian）。

5. ESP32-S3-Devkit，是一款高性能、低功耗的微控制器，適合用於物聯網應用。其特點包括：

- 處理器：雙核 Xtensa LX7，最高 240MHz。
- 內存：512KB SRAM，支持外部 RAM 擴展。
- 輸入/輸出：支援 microUSB 連接方式，同時也支援 Wi-Fi 和藍芽等無線連接方式。
- 作業系統：本實驗採用 circuit python。

6. ESP32 Doit-Devkit，是一個常見的 ESP32 開發板，設計簡單且功能強大。其特點包括：

- 處理器：雙核 Xtensa LX6，最高 240MHz。
- 內存：520KB SRAM。
- 支援 microUSB 連接方式，同時也支援 Wi-Fi 和藍芽等無線連接方式。
- 作業系統：本實驗採用 circuit python。

7. L298N，直流馬達驅動模組，常用於控制雙馬達系統，本實驗將此裝置用於控制車輪。其特點包括：

- 電壓範圍：可操作電壓範圍為 5V 至 35V。

- 電流：每通道峰值電流可達 2A。
- 控制：使用 PWM 訊號控制馬達速度和方向。

8. PCA9685，是一款 16 通道 PWM 驅動模組，通常用於控制伺服馬達和 LED 燈，本實驗將此裝置用於控制機械臂馬達。其特點包括：

- PWM 頻率：可調頻率範圍為 24Hz 至 1526Hz。
- 接口：使用 I2C 介面進行訊號傳輸。
- 控制：使用 PWM 訊號控制馬達速度和方向，適合需要一次性控制大量馬達的專案。

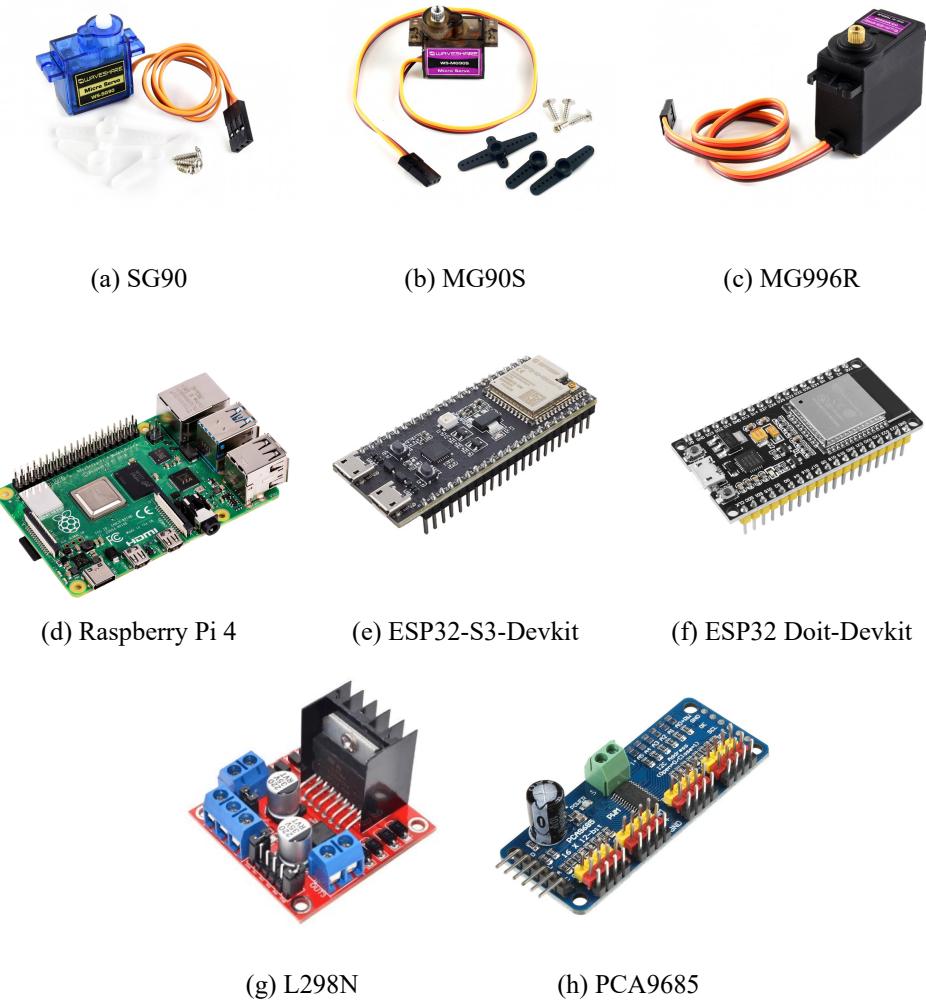


圖 3.4: 開發板與馬達縮圖

```

1
2     class ArmEnv_3D:
3         def __init__(self, lList, vision=True):
4             self.Link_length = lList
5             self.Link_offset=[0]*len(lList)
6             self.Twist_angle=[0]*len(lList)
7             self.Joint_angle = [0]*len(lList)
8             self.Target_point=[0, 0, 0]
9             self.Current_point=[0, 0, sum(lList)]
10
11        def Forward_kinematics(self, joint_angles):...
12
13        def Geometric_solution(self):...
14
15        def Gradient_policy(self,times=100, points=10, speed=5):...
16
17        def initPicture(self):...
18
19        def updatePicture(self):...
20

```

Listing 3.1: 運動模擬環境程式架構

## 3.2 運動學開發

### 3.2.1 運動模擬環境

本專案使用 Python 語言開發了一個簡易的機械臂模擬環境。這個模擬環境的基本原理，是將機械臂定義為一個擁有多個可調整參數的獨立物件，其中包括力臂長度、扭轉角、節點偏移、節點角度、目標頂點、和目前頂點等。這些參數可以由使用者根據需求進行調整，從而在虛擬空間中生成機械臂的模型。該模擬環境提供了一個直觀且靈活的介面，使得使用者能夠方便地設計和測試機械臂的運動行為。

### 3.2.2 順向運動學

在順向運動學部分，本專案採用了 Denavit-Hartenberg 方法作為主要的計算方式。這種方法能夠系統地描述和計算機械臂的運動行為。模擬環境中的相關函式可以直接被呼叫，利用即時的機械臂參數來計算操作點的空間座標。計算結果會即時顯示在模擬空間中。

### 3.2.3 逆向運動學

在逆向運動學方面，本專案結合了幾何求解 (Geometric solution approach) 方法和策略梯度 (Policy-Gradient solution approach) 方法來進行計算。幾何求解方法主要用於計算運動模式較為簡單的機械臂，而策略梯度方法則適用於運動較為複雜的機械臂。這些計算函式同樣可以在模擬環境中直接被呼叫。根據輸入的目標操作點位置，系統會計算出對應的扭轉角，並將結果顯示在模擬空間中。

## 3.3 大型語言模型開發

### 3.3.1 OpenAI 與 GPT 模型

本實驗使用的大型語言模型為 OpenAI 所推出的 GPT (Generative Pre-trained Transformer) 模型，GPT 模型是一種基於變換器 (Transformer) 架構的生成式預訓練模型，用於應對自然語言處理相關的任務。自推出以來，此模型已經經歷了多次迭代，從最初的 GPT-1 到最新的 GPT-4o，每一代都在參數量、性能與應用靈活度上取得了顯著的進步。這些模型通過在大規模文本數據上進行預訓練，能夠生成高品質文字、圖片等資訊。

```

1  {
2      "model": "model_type",
3      "prompt": " 介紹一下 OpenAI 和 GPT 模型。",
4      "max_tokens": 100,
5      "temperature": 0.7,
6      "top_p": 1.0,
7      "n": 1,
8      "stop": null
9  }

```

Listing 3.2: JSON 請求範例

### 3.3.2 模型調用流程

呼叫 OpenAI chat-GPT API 的流程，簡單可以分為以下幾個步驟：

1. 註冊與獲取密鑰：在 OpenAI 的官方網站上註冊一個帳號，並建立一個 API 密鑰。
2. 安裝必要的函式庫：本實驗使用 python 作為開發語言，所以需要在開發環境中安裝 python 版本的函式庫。
3. 在程式中設置密鑰：在程式碼中設定建立好的密鑰，以供官方伺服器驗證。
4. 發送請求與處理回應：接下來即可向伺服器發送請求，並處理文本回應。

在對 GPT 模型下達指令時，需要將預計上傳的文字資訊封裝成特定格式，以下為簡易範例：

以下為主要參數說明：

- model: 指定使用的 GPT 模型，例如 text-davinci-004、GPT-4o、GPT-3.5 等。

```

1 curl https://api.openai.com/v1/completions \
2   -H "Content-Type: application/json" \
3   -H "Authorization: Bearer API 密鑰" \
4   -d '{
5     "model": "model_type",
6     "prompt": "介紹一下 OpenAI 和 GPT 模型。",
7     "max_tokens": 100,
8     "temperature": 0.7,
9     "top_p": 1.0,
10    "n": 1,
11    "stop": null
12  }'
13

```

Listing 3.3: HTTP 請求範例

- prompt: 下達的指令。
- max\_tokens: 回傳文字的最大長度。
- temperature: 控制生成文字的隨機性。此值越高會生成更具創造性的文字，反之值越低則生成越保守的文字。
- top\_p: 取樣方法，決定生成文本的多樣性。
- n: 返回生成文章的數量。
- stop: 用於指定生成文字停止的條件，可設定為空值。

## 3.4 系統架構

### 3.4.1 系統架構與流程

本實驗的系統架構與流程大致如下：

1. GUI 上傳使用者輸入至 Server：使用者在 GUI 上輸入文字並上傳至伺服器。

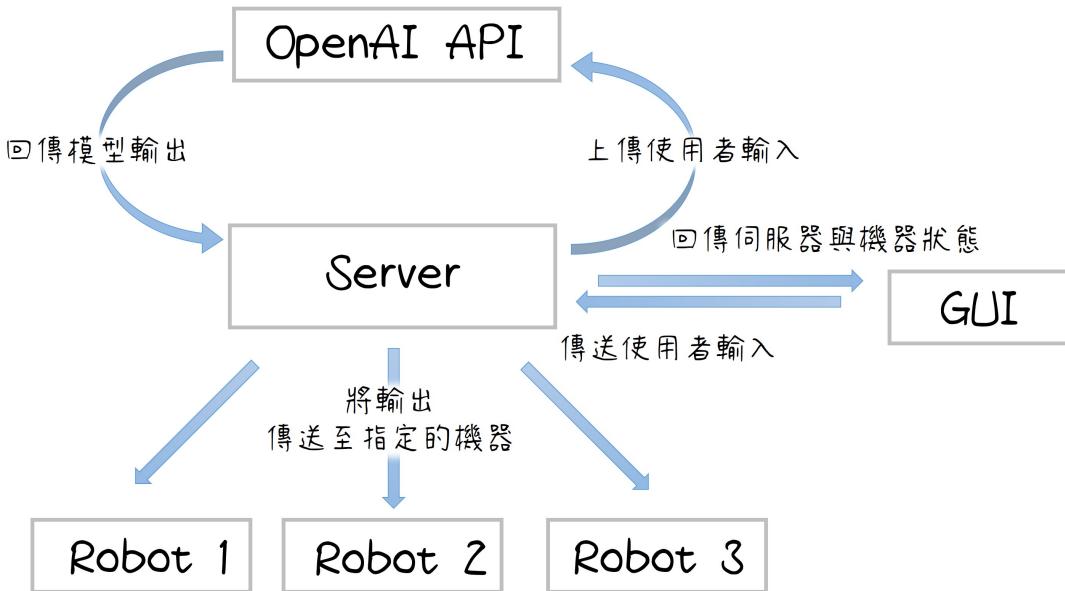


圖 3.5: 系統架構圖

2. Server 上傳使用者輸入至 OpenAI API：伺服器將這些使用者輸入發送至 OpenAI API。
3. OpenAI API 回傳模型輸出至 Server：OpenAI API 處理輸入並回傳模型生成的輸出至伺服器。
4. Server 將輸出傳送至指定的機器：伺服器將這些輸出數據傳送至指定的機器，進行控制。
5. Server 向 GUI 回傳伺服器與機器狀態：最後，伺服器將機器和自身的狀態訊息回傳至 GUI，讓使用者了解系統運行情況。

## 四、 實驗設計與結果

### 4.1 實驗一：機械臂的基本控制

#### 4.1.1 機械結構設計圖

本實驗的硬體部分使用了 3D 列印技術，結合小型伺服馬達，設計了一個頂端為夾爪的小型機械臂，以下為此硬體的詳細設計圖紙：

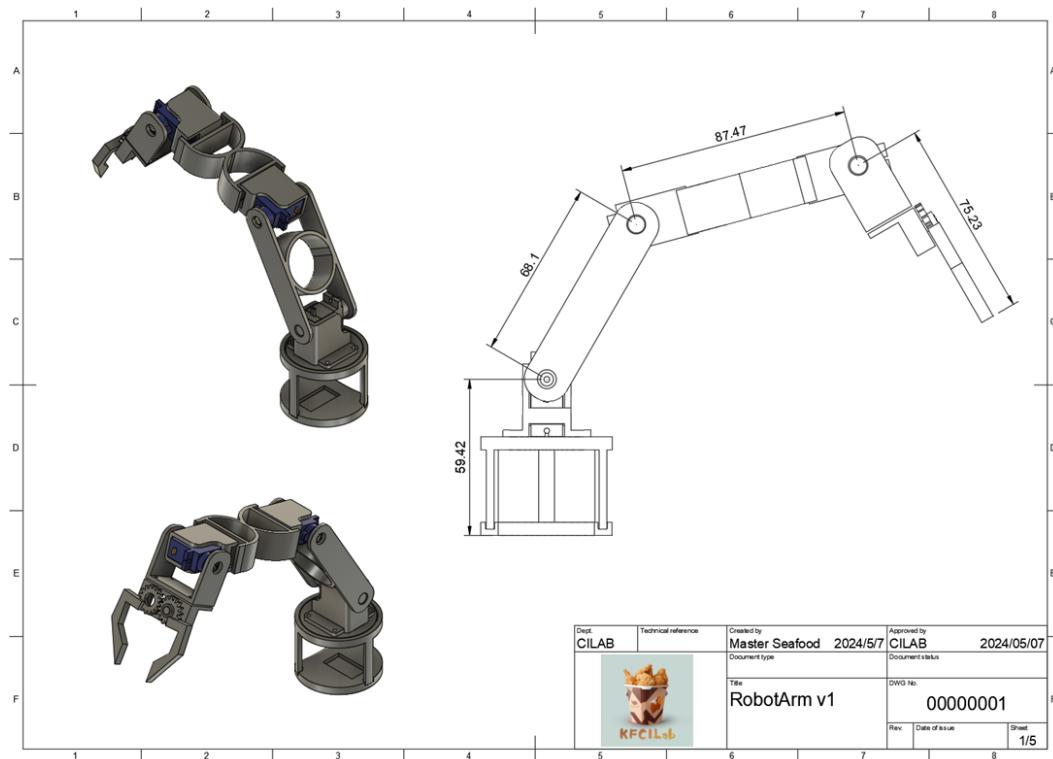


圖 4.1: 機械臂版本一設計圖紙第一頁 (單位:mm)

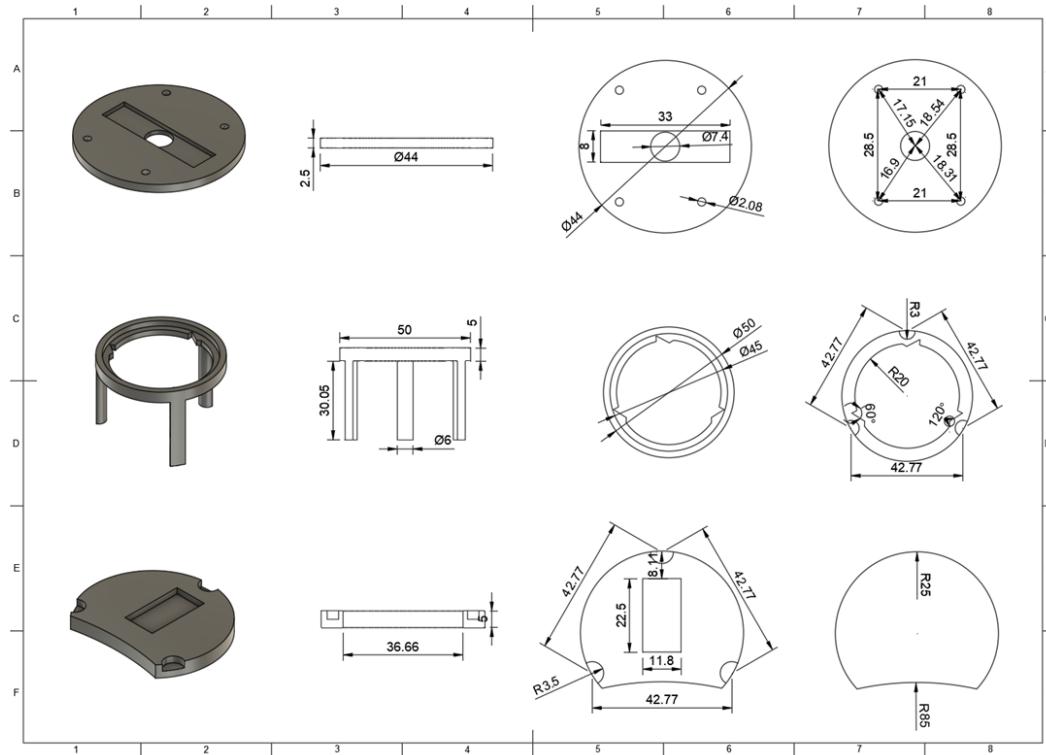


圖 4.2: 機械臂版本一設計圖紙第二頁 (單位:mm)

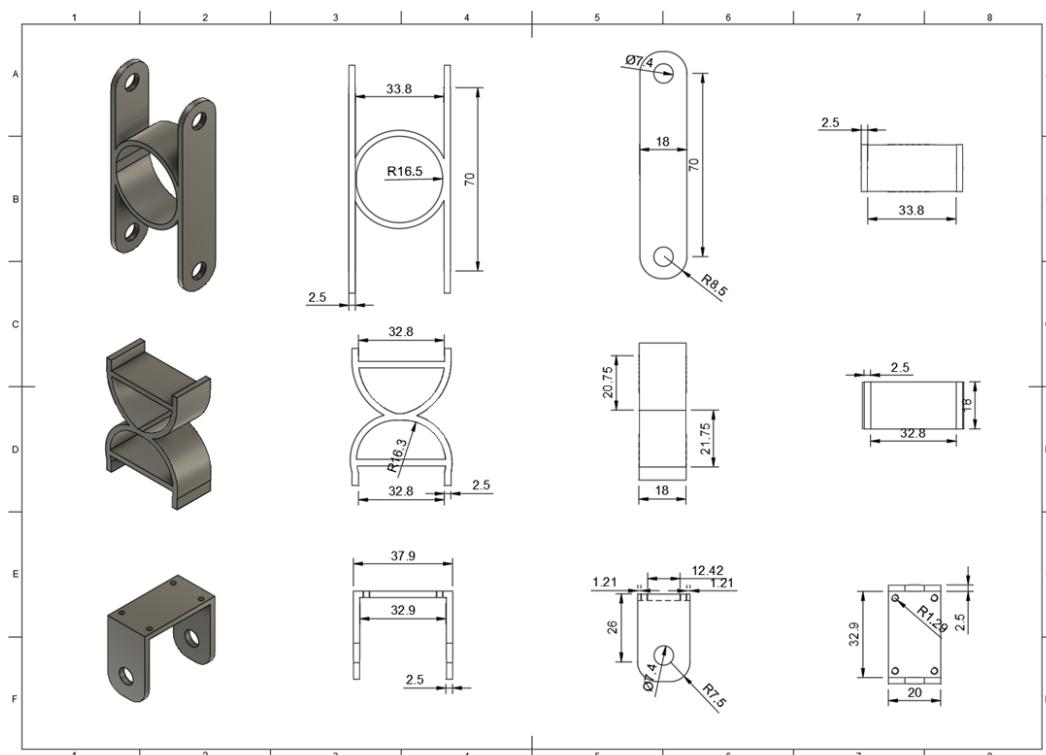


圖 4.3: 機械臂版本一設計圖紙第三頁 (單位:mm)

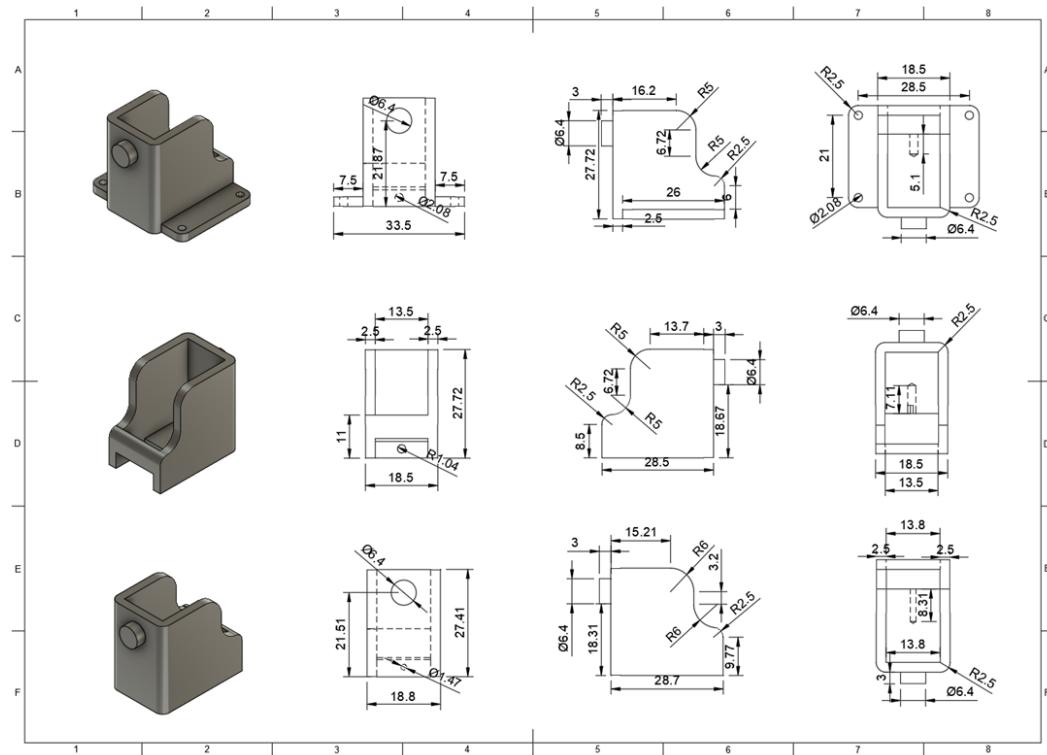


圖 4.4: 機械臂版本一設計圖紙第四頁 (單位:mm)

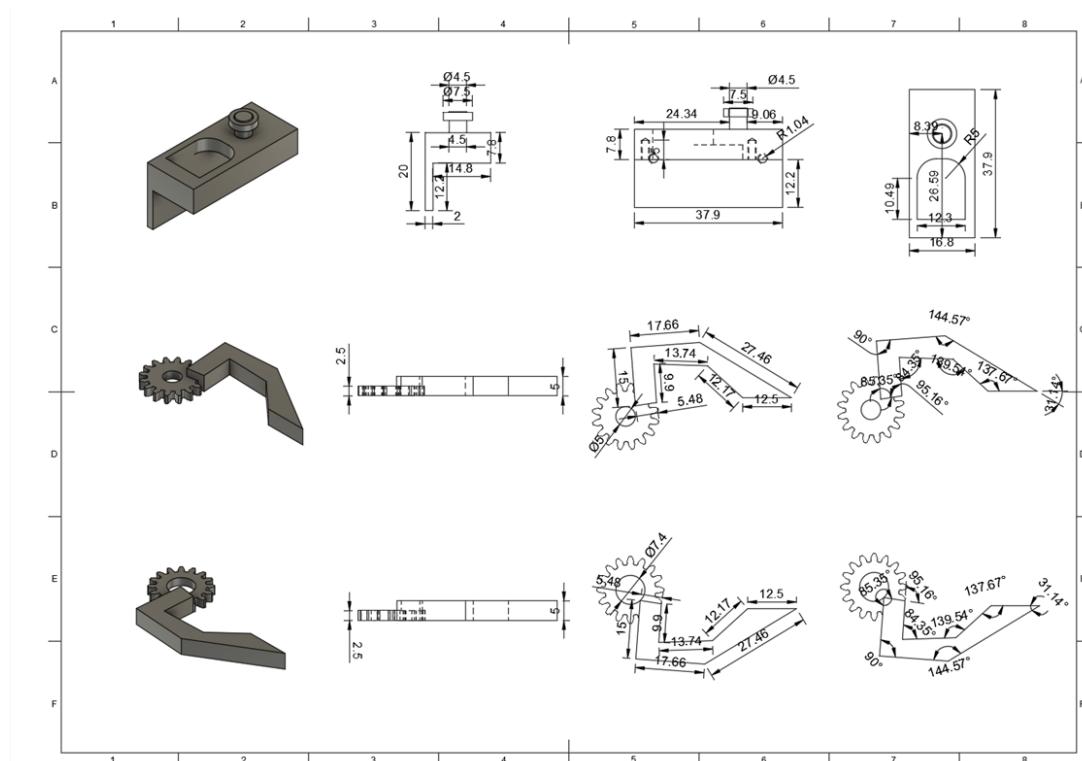


圖 4.5: 機械臂版本一設計圖紙第五頁 (單位:mm)

### 4.1.2 函數設計

在這個實驗中，我們的目標是讓機械臂能夠成功夾取特定物體，並放置於指定位置。這個過程涉及機械臂路徑的規劃和精確的抓取動作。為了實現這一目標，我們撰寫了一個簡易的路徑規劃程式，利用語言模型在填空方面的優勢，使其生成和編輯機械臂的運動指令。使用這樣的方式，嘗試讓大型語言模型自動生成適當的路徑規劃，並將其應用於機械臂的控制，以下為此實驗使用的程式碼：

```

1
2   from ArmEnv_3D_ori import ArmEnv_3D
3   import math
4
5   Arm = ArmEnv_3D([5.942, 6.81, 8.747, 7.523], False)
6   path = []
7   #Please planning path form here
8   #path+=Arm.moveto([0, 0, 25], 20) explain: Move to [0, 0, 25] in 20 steps
9   #path+=[[91, 0, 0, 0]] explain: Close claw
10  #path+=[[91, 0, 0, 0]] explain: Open claw
11
12  #reset
13  end=path[-2].copy()
14  while not sum(end)==0:
15      for i in range(len(end)):
16          if end[i]>0: end[i]-=1
17          elif end[i]<0: end[i]+=1
18      path.append(end.copy())
19

```

Listing 4.1: 實驗程式碼

### 4.1.3 下達指令的格式範例

```

1  {
2      role: "user",
3      content :
4      "\n
5         from ArmEnv_3D_ori import ArmEnv_3D\
6         import math\
7         \
8         Arm = ArmEnv_3D([5.942, 6.81, 8.747, 7.523], False)\
9         path = []\
10        #Please planning path form hear\
11        #path+=Arm.moveto([0, 0, 25], 20) explain: Move to [0, 0, 25] in 20 steps\
12        #path+=[[[-91, 0, 0, 0]]] explain: Close claw\
13        #path+=[[91, 0, 0, 0]] explain: Open claw\
14        \
15        #reset\
16        end=path[-2].copy()\
17        while not sum(end)==0:\\
18            for i in range(len(end)):\\
19                if end[i]>0: end[i]-=1\\
20                elif end[i]<0: end[i]+=1\\
21                path.append(end.copy())\\
22        "
23    },
24    {
25        role: "user",
26        content: "This is a sample program for controlling a robotic arm.\\
27            Please modify it to help me complete the following requirements:\\
28            the ball is at [10.5, 12, 2], and the top of the tower is at [9.5, 0, 14.5]. \\
29            Please help me catch the ball to the top of the tower. \\
30            During the process, please do not get too close to the side of the tower \\
31            (keep outside the radius 7).\\
32            return only code.\\
33        "
34    }

```

Listing 4.2: 實驗一：指令格式範例

```

1
2 {
3     role="assistant",
4     content=
5     "\n
6         from ArmEnv_3D_ori import ArmEnv_3D\
7         import math\
8
9         Arm = ArmEnv_3D([5.942, 6.81, 8.747, 7.523], False)\
10        path = []
11        path += Arm.moveto([0, 0, 25], 20)\n
12        path += Arm.moveto([10.5, 12, 2], 20)\n
13        path+=[-91, 0, 0, 0]]\n
14        path += Arm.moveto([10, 13, 15], 20)\n
15        path += Arm.moveto([9.5, 0, 14.5], 20)\n
16        path+=[[91, 0, 0, 0]]\n
17
18        #reset\n
19        end=path[-2].copy()\n
20        while not sum(end)==0:\n
21            for i in range(len(end)):\n
22                if end[i]>0: end[i]-=1\n
23                elif end[i]<0: end[i]+=1\n
24                path.append(end.copy())\n
25
26    }
27

```

Listing 4.3: 實驗一：回傳格式範例

#### 4.1.4 實驗結果

以下為本實驗的實驗過程縮圖總結：

表 4.1: 實驗結果：Please catch the ball to the top of the tower.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	7	3	0.7
GPT-3.5 Turbo	10	4	6	0.4

根據本實驗結果顯示，使用大型語言模型進行路徑規劃的效果相當不錯，並且隨著語言模型技術的發展，路徑規劃的準確率也有顯著的提升。然而，目前的方法需要將整個程式碼放入語言模型的輸入中以生成控制程式碼，這種方式過於繁瑣。此外，這種方法也需要提供大量的前置條件，使用起來相當不便。因此，我們認為有必要對這個實驗進行進一步的調整，以提高其實用性和效率。

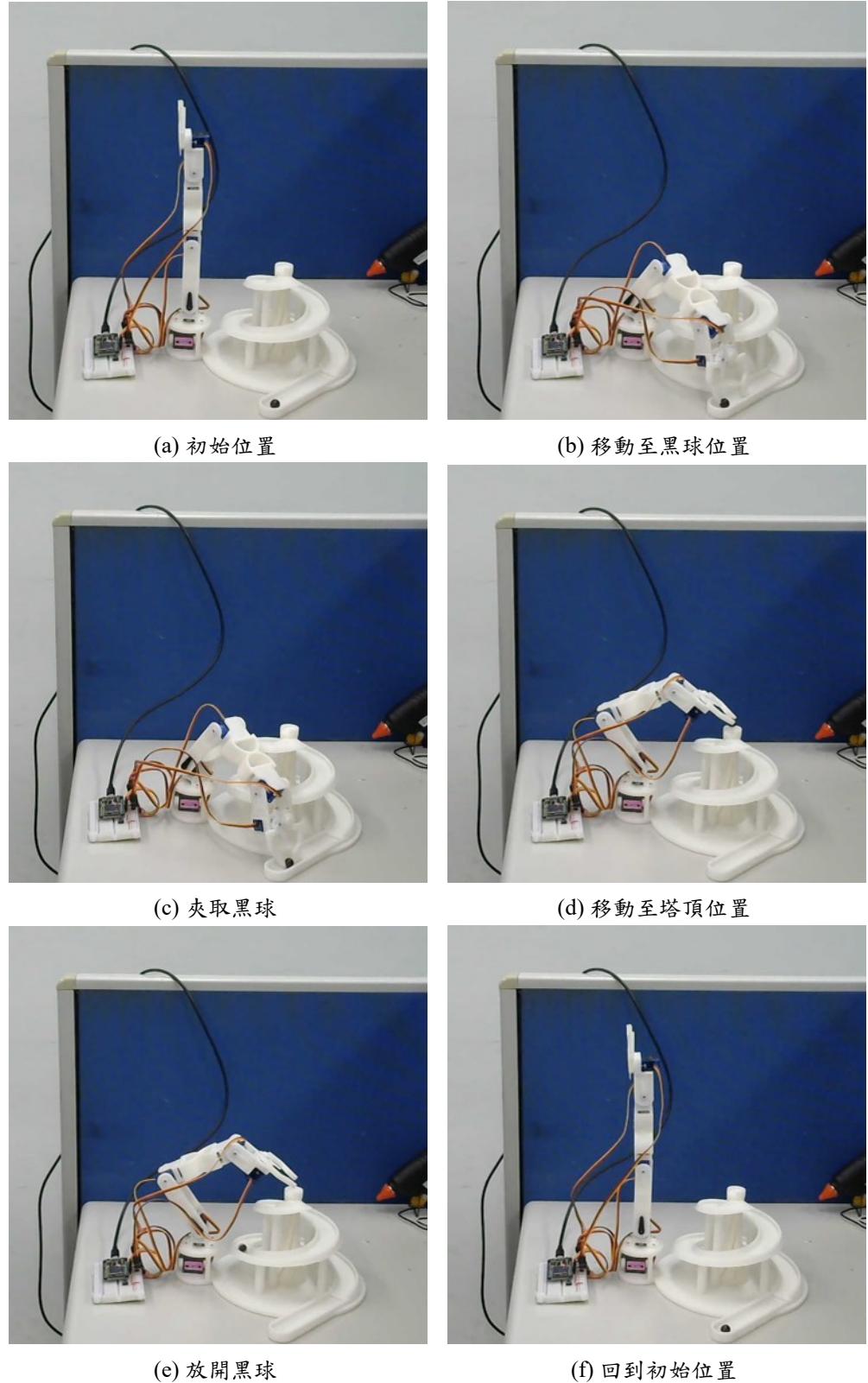


圖 4.6: 實驗一：實驗過程縮圖

## 4.2 實驗二：將機械臂用於畫圖

### 4.2.1 機械結構設計圖

本實驗的硬體部分使用了 3D 列印技術，結合小型伺服馬達，設計了一個畫筆的小型機械臂，以下為此硬體的詳細設計圖紙：

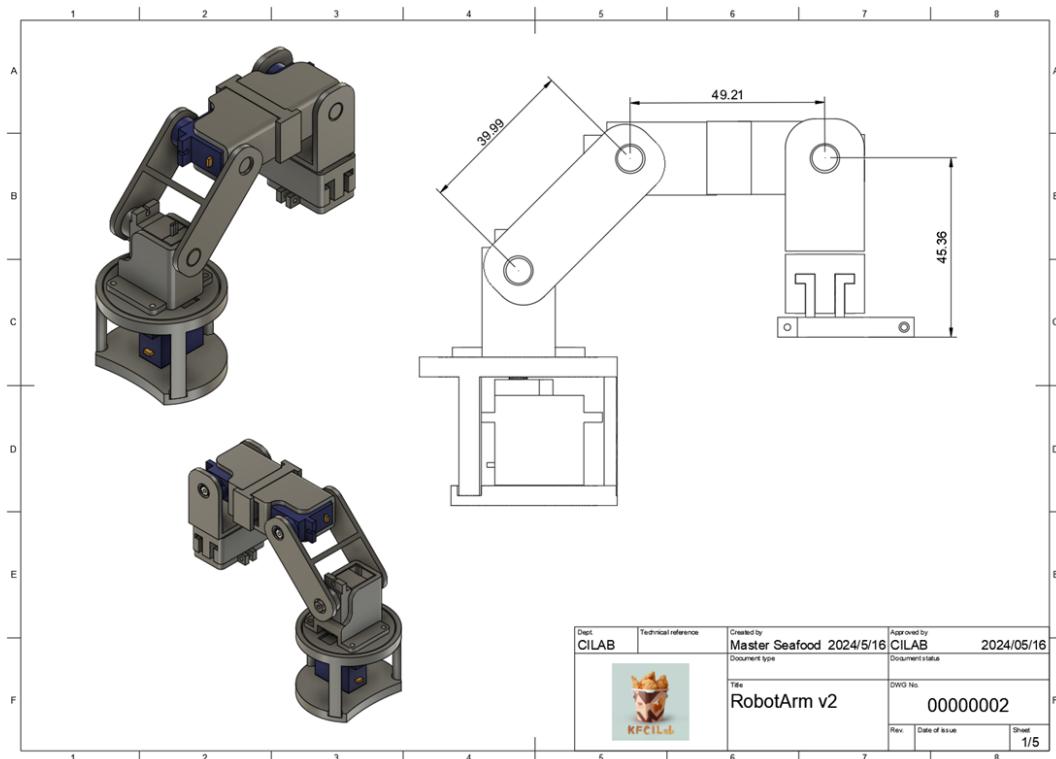


圖 4.7: 機械臂版本二設計圖紙第一頁 (單位:mm)

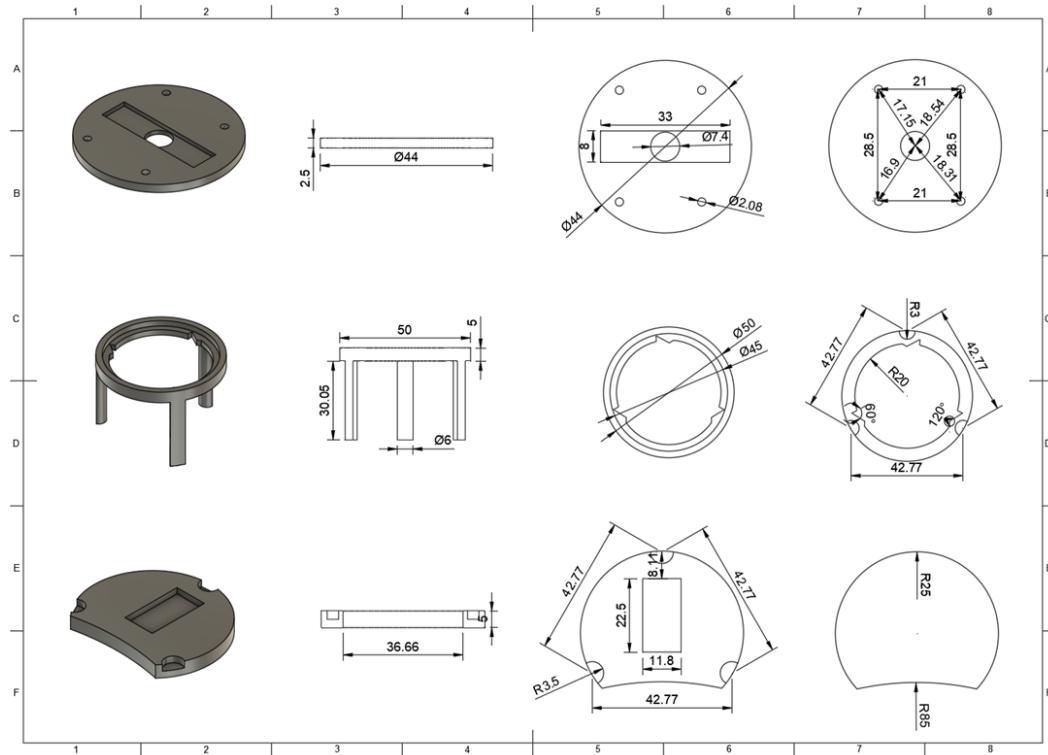


圖 4.8: 機械臂版本二設計圖紙第二頁 (單位:mm)

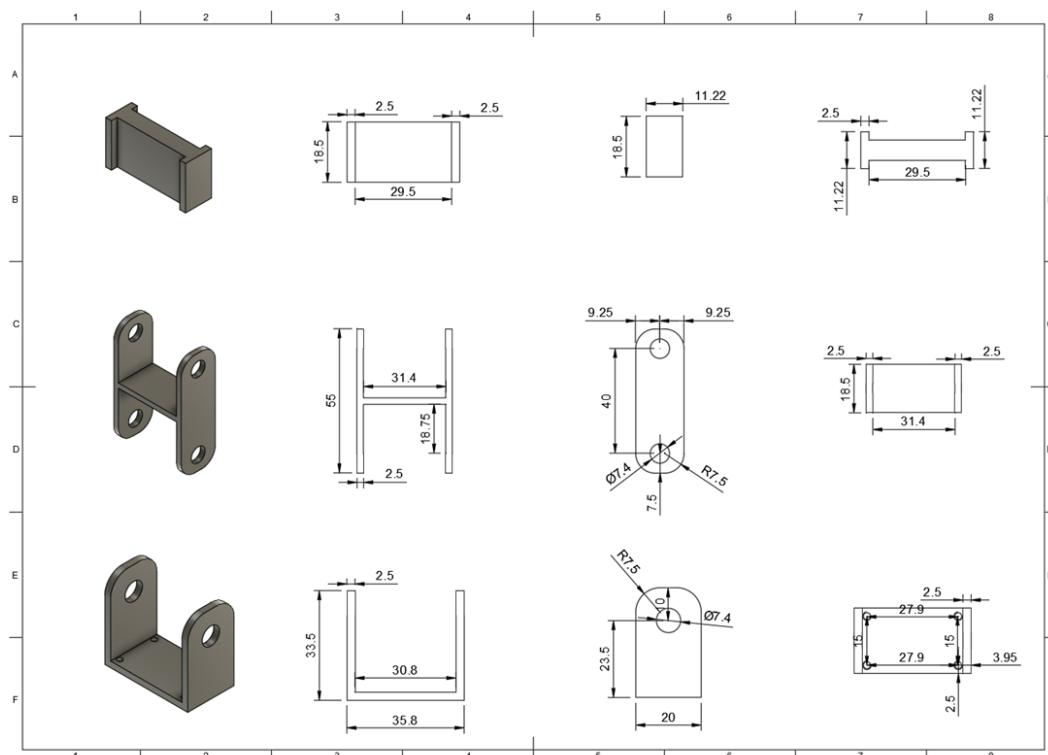


圖 4.9: 機械臂版本二設計圖紙第三頁 (單位:mm)

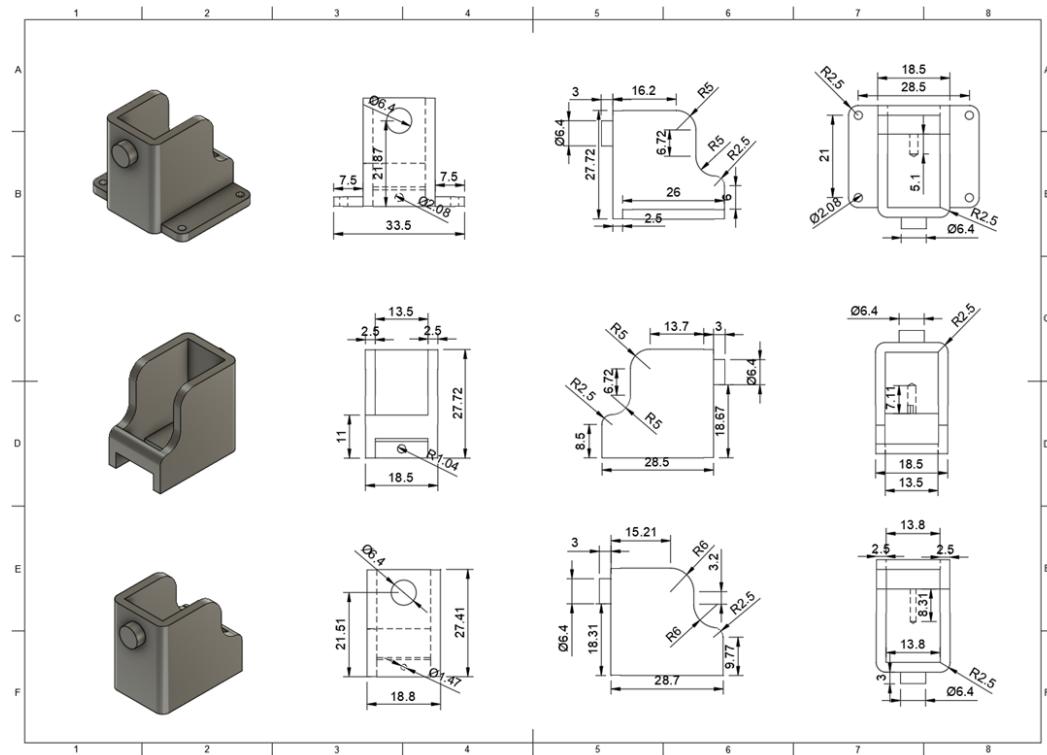


圖 4.10: 機械臂版本二設計圖紙第四頁 (單位:mm)

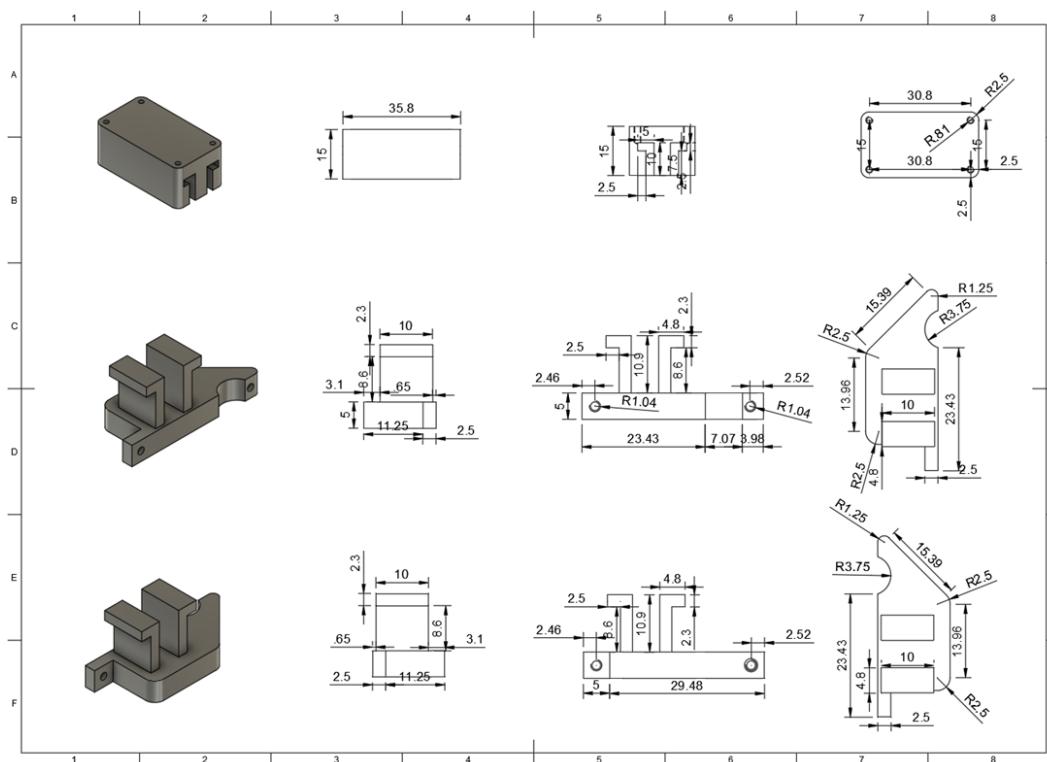


圖 4.11: 機械臂版本二設計圖紙第五頁 (單位:mm)

### 4.2.2 函數設計

從實驗一的基礎上，我們改進了機械臂的路徑規劃方法，將其應用於讓機械臂進行畫畫。此次實驗中，我們簡化了路徑規劃，使大型語言模型在平面座標內進行規劃，這樣可以降低複雜度，並節省資源。通過這種簡化的路徑規劃方法，我們能夠更有效地控制機械臂，讓其能夠在紙上或畫布上繪製出預定的圖案。本實驗將此機器的動作指令簡化為以下函式：

- draw(from, to): 由變數 from 提供的座標，畫至變數 to 提供的座標。

### 4.2.3 下達指令的格式範例

```

1  {
2    role: "user",
3    content :
4      "The following functions are available:\\
5      \
6      draw(from, to): Let the arm draw a straight line from from to to,\\
7      such as draw([0, 0], [5, 5]).\
8      the limit is[0, 5]\\
9      \
10     Please help me use the above functions to control the robot arm,\\
11     and do not output other text other than the above functions.\\
12     (Use "," to separate each step)"
13   },
14   {
15     role: "user",
16     content: "Task: Draw a square."
17   }

```

Listing 4.4: 實驗二：指令格式範例

```

1  {
2    role="assistant",
3    content="draw([1, 1], [1, 4]),\
4      draw([1, 4], [4, 4]),\
5      draw([4, 4], [4, 1]),\
6      draw([4, 1], [1, 1])"
7  }
8
9

```

Listing 4.5: 實驗二：回傳格式範例

#### 4.2.4 實驗結果

以下為本實驗的實驗過程縮圖總結：

表 4.2: 實驗結果：Draw a square.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	10	0	1
GPT-3.5 Turbo	10	10	0	1

表 4.3: 實驗結果：Draw a triangle.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	10	0	1
GPT-3.5 Turbo	10	10	0	1

表 4.4: 實驗結果：Draw a star.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	9	1	0.9
GPT-3.5 Turbo	10	5	5	0.5

表 4.5: 實驗結果：Draw a house.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	7	3	0.7
GPT-3.5 Turbo	10	0	10	0

在這次實驗中，我們針對上一個實驗中存在的問題進行了改進。首先，我們將複雜的三維空間路徑規劃簡化為平面上的路徑規劃。使的整個控制系統不在需要提供冗長的程式碼與相關條件條件，只需提供特定的圖形，就能驅動機械臂進行繪圖的控制模式，從實驗結果可以看出，這種改進的控制方式不僅大大減少了輸入和回傳的資料量，同時也保持了相當不錯的準確率。這表明此簡化方法在實現高效控制的同時，並未犧牲其性能。因此，我們決定將實驗繼續沿著這個方向進行調整和優化。

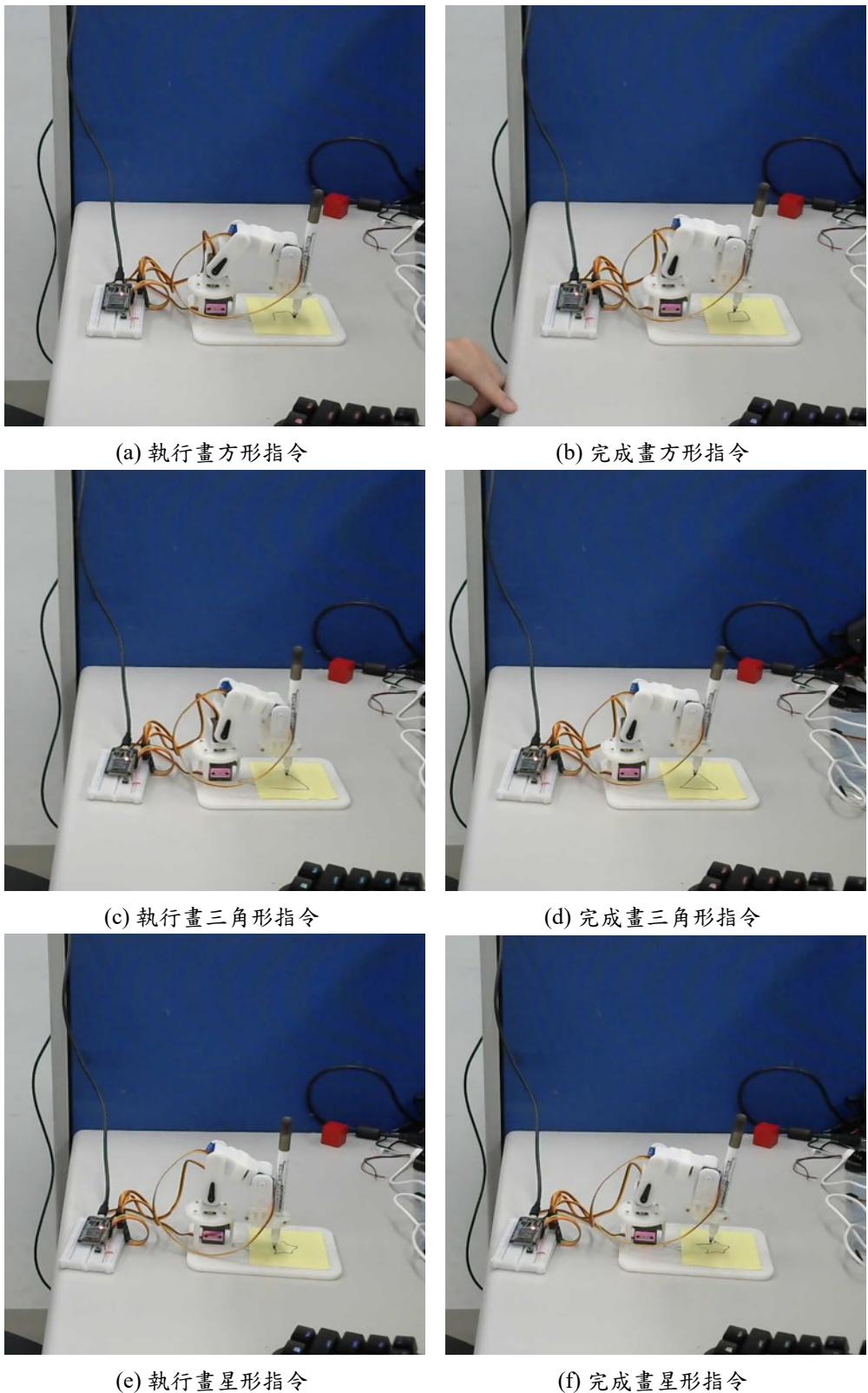


圖 4.12: 實驗二：實驗過程縮圖

## 4.3 實驗三：機械臂在自動運輸車上的應用

### 4.3.1 機械結構設計圖

本實驗將機械臂與自動運輸車、鏡頭等硬體結合，設計了一台裝載機械臂的無人搬運車，以下為此硬體的詳細設計圖紙：

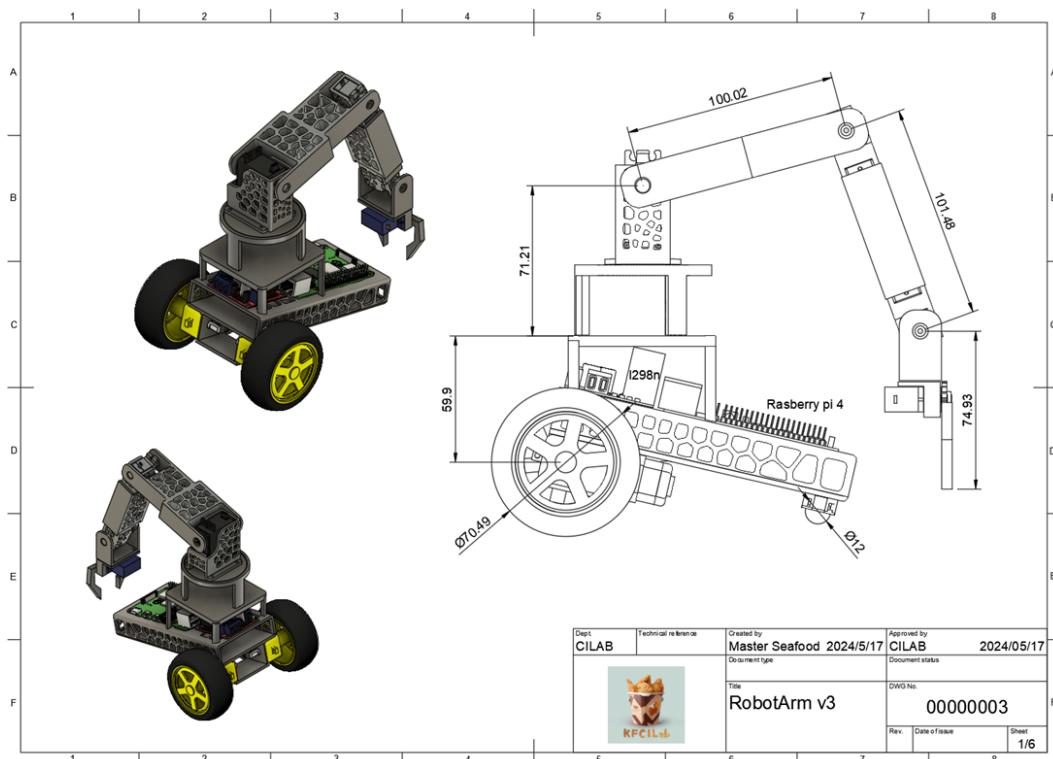


圖 4.13: 機械臂版本三設計圖紙第一頁 (單位:mm)

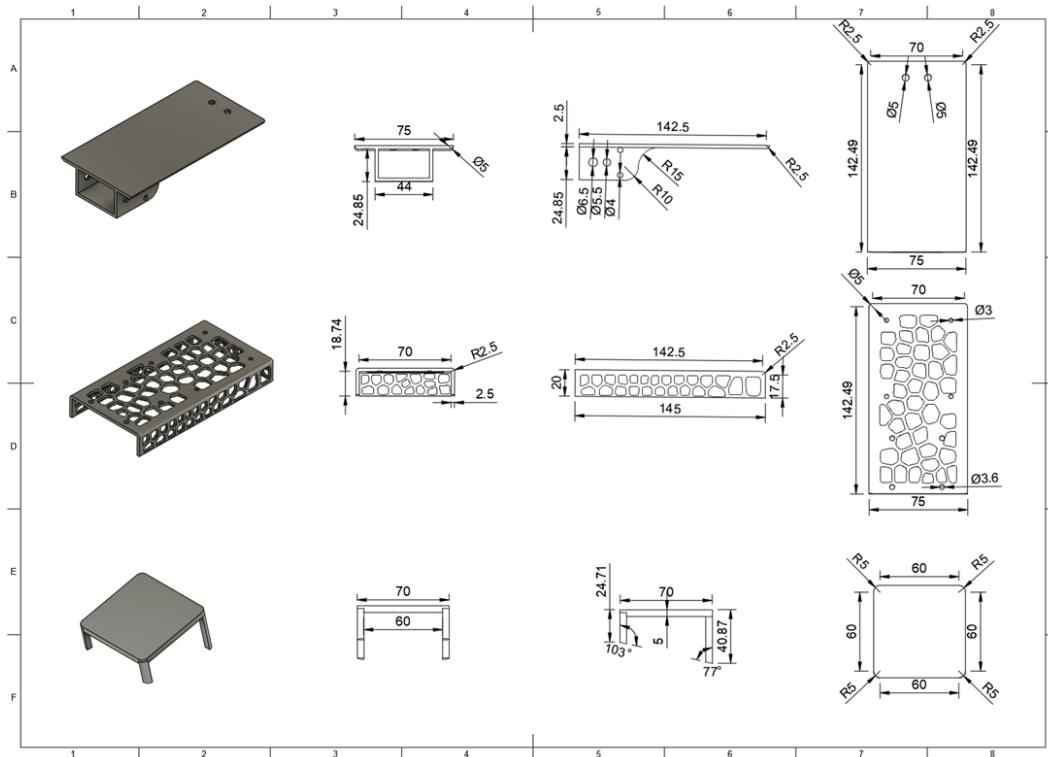


圖 4.14: 機械臂版本三設計圖紙第二頁 (單位:mm)

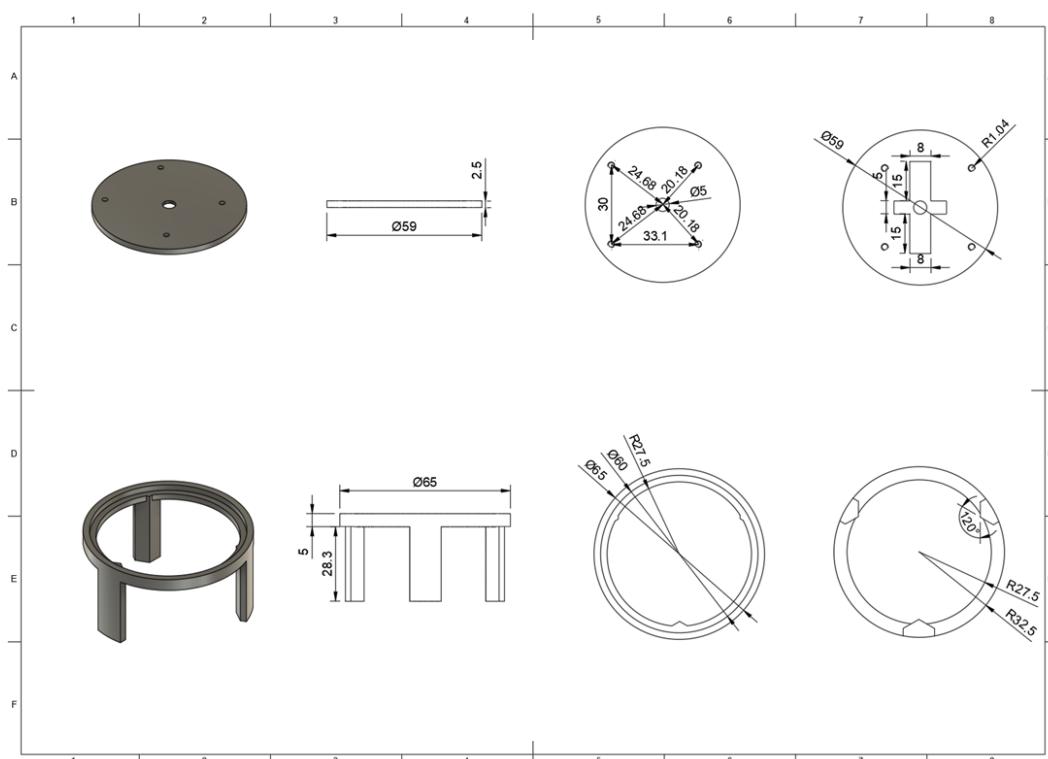


圖 4.15: 機械臂版本三設計圖紙第三頁 (單位:mm)

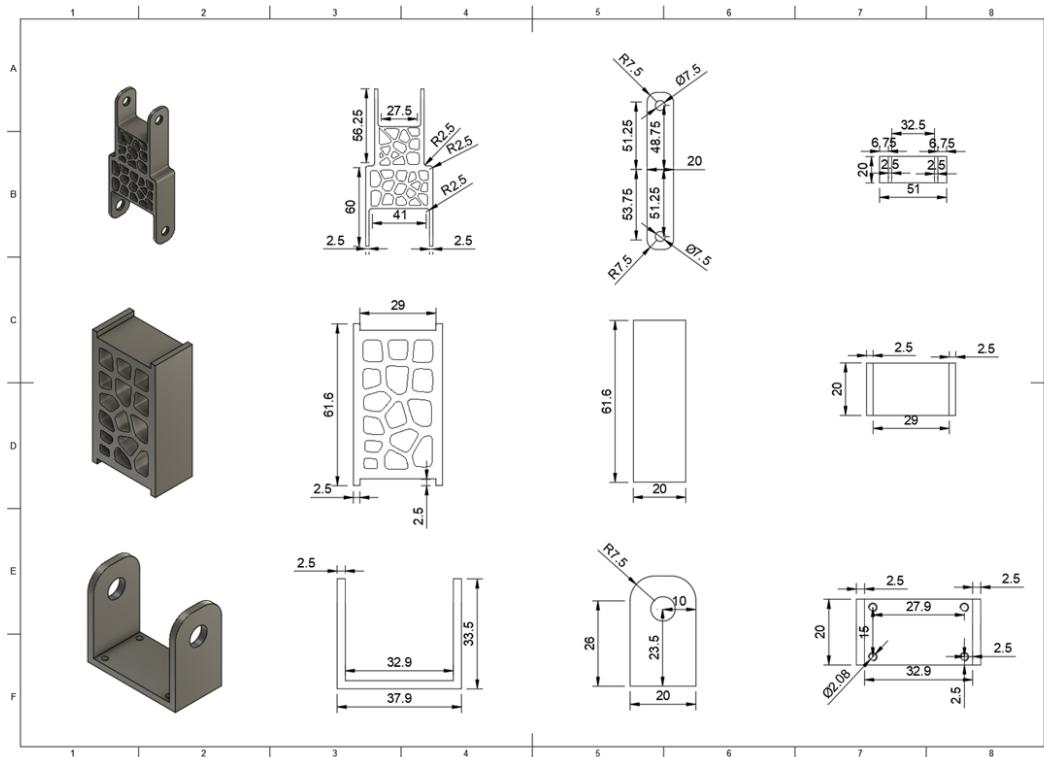


圖 4.16: 機械臂版本三設計圖紙第四頁 (單位:mm)

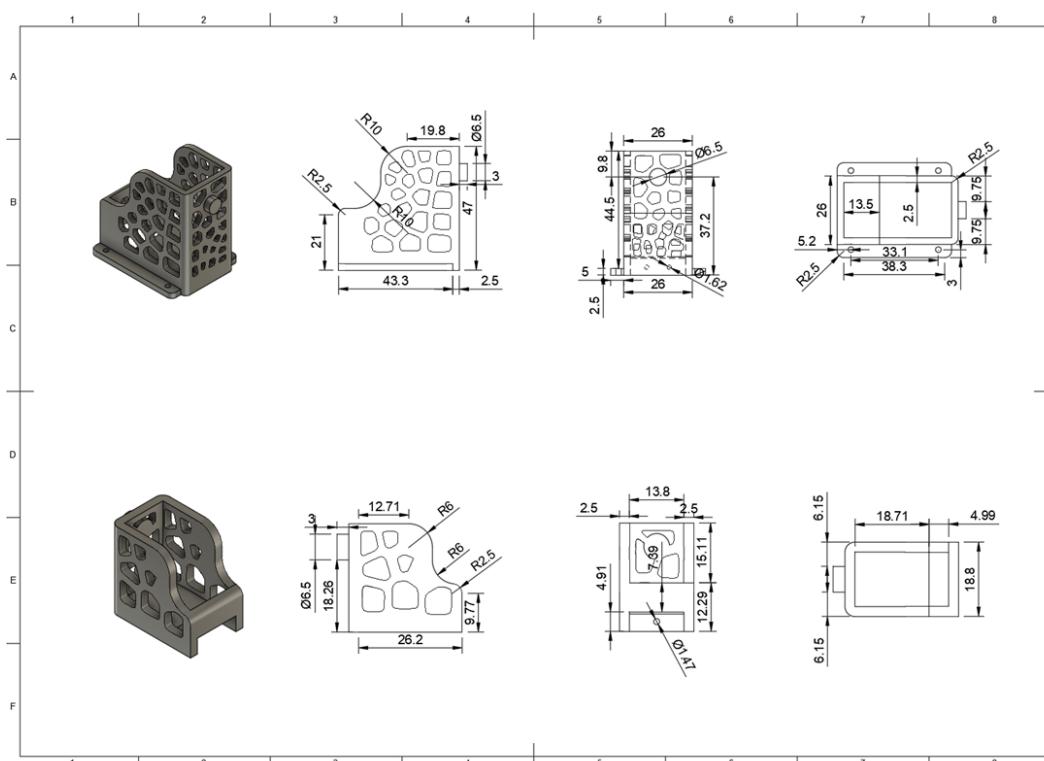


圖 4.17: 機械臂版本三設計圖紙第五頁 (單位:mm)

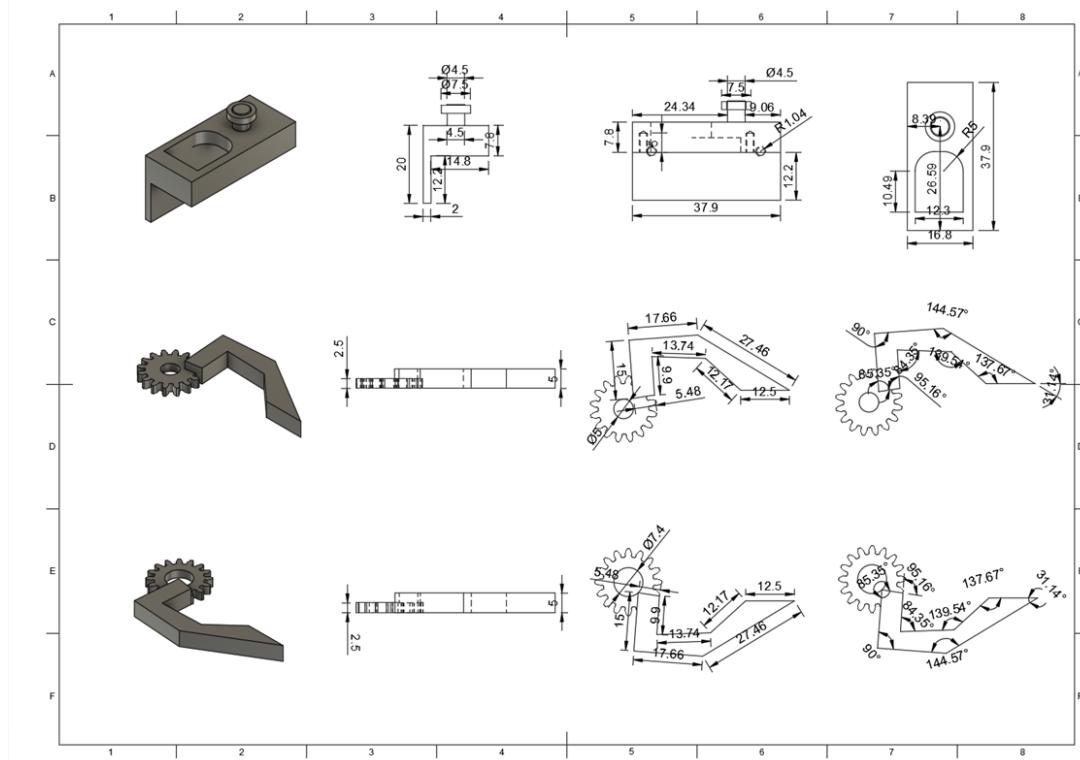


圖 4.18: 機械臂版本三設計圖紙第六頁 (單位:mm)

### 4.3.2 函數設計

在前兩個實驗的基礎上，我們進一步改進了控制流程，將機械臂與自動運輸車整合在一起。這次，我們將控制函式進行細分，並模組化，使得大型語言模型能夠通過簡單的指令來控制機械臂和車輛的行動。這種模組化的設計不僅降低了指令的複雜度，還減少了資源利用。最終，我們能夠通過簡單的模組化指令來控制整個系統，使得機械臂和車輛能夠協同工作，完成更複雜的任務。

- `find(color)`: 配合相機定位，驅動輪子尋找對應顏色的方塊。
- `aim(color)`: 配合相機定位，驅動機械臂瞄準對應顏色的方塊。
- `grab()`: 驅動機械臂，抓取正下方的物品。
- `reset()`: 驅動機械臂，回到初始位置

### 4.3.3 下達指令的格式範例

以下為指令格式：

```

1  {
2    role: "user",
3    content :
4      "The following functions are available:\\
5      \
6        find(color): Let the robot look for a block of a specific color.\\
7        color option: red, blue.\\
8        aim(color): Let the robot aim at a block of a specific color.\\
9        color option: red, blue\\
10       grab: Make the robot arm grab the block and put it down.\\
11       reset: Return the robotic arm to its initial position\\
12       (needs to be executed before each aiming).\\
13       \
14       Please help me use the above functions to control the robot arm,\\
15       and do not output other text other than the above functions.\\
16       (Use "," to separate each step)"
17   },
18   {
19     role: "user",
20     content: "Task: Please grab the red block, and then grab the blue block."
21   }

```

Listing 4.6: 實驗三：指令格式範例

```

1  {
2    role="assistant",
3    content="find(red), aim(red), grab(), reset(), find(blue), aim(blue), grab()"
4  }
5
6

```

Listing 4.7: 實驗三：回傳格式範例

#### 4.3.4 實驗結果

以下為本實驗的實驗過程縮圖總結：

表 4.6: 實驗結果：Please grab the red block, and then grab the blue block.

大型語言模型	總次數	成功次數	失敗次數	成功率
GPT-4o	10	9	1	0.9
GPT-3.5 Turbo	10	7	3	0.7

在本次實驗中，我們增加了車輪、鏡頭等更多的硬體連動設備。由於每個任務都需要協調各個硬體的運作才能順利完成，我們在撰寫控制程式時，將各個重要指令整理成更加簡化且直觀的獨立函式，這樣可以方便大型語言模型在生成控制指令時直接呼叫這些函式。實驗結果顯示，這種方法取得了非常好的成效，顯著提高了控制指令的生成效率和準確性。然而，由於連動的硬體較多，在呼叫控制函式時難免會發生一些機械性的失誤。例如，鏡頭失真可能導致機械臂瞄準失誤，而車輪的移動則可能導致機械臂在夾取物體時出現掉落的情況。針對這些問題，我們希望在未來進一步優化硬體間的協調機制，以提升系統的穩定性和精確度。

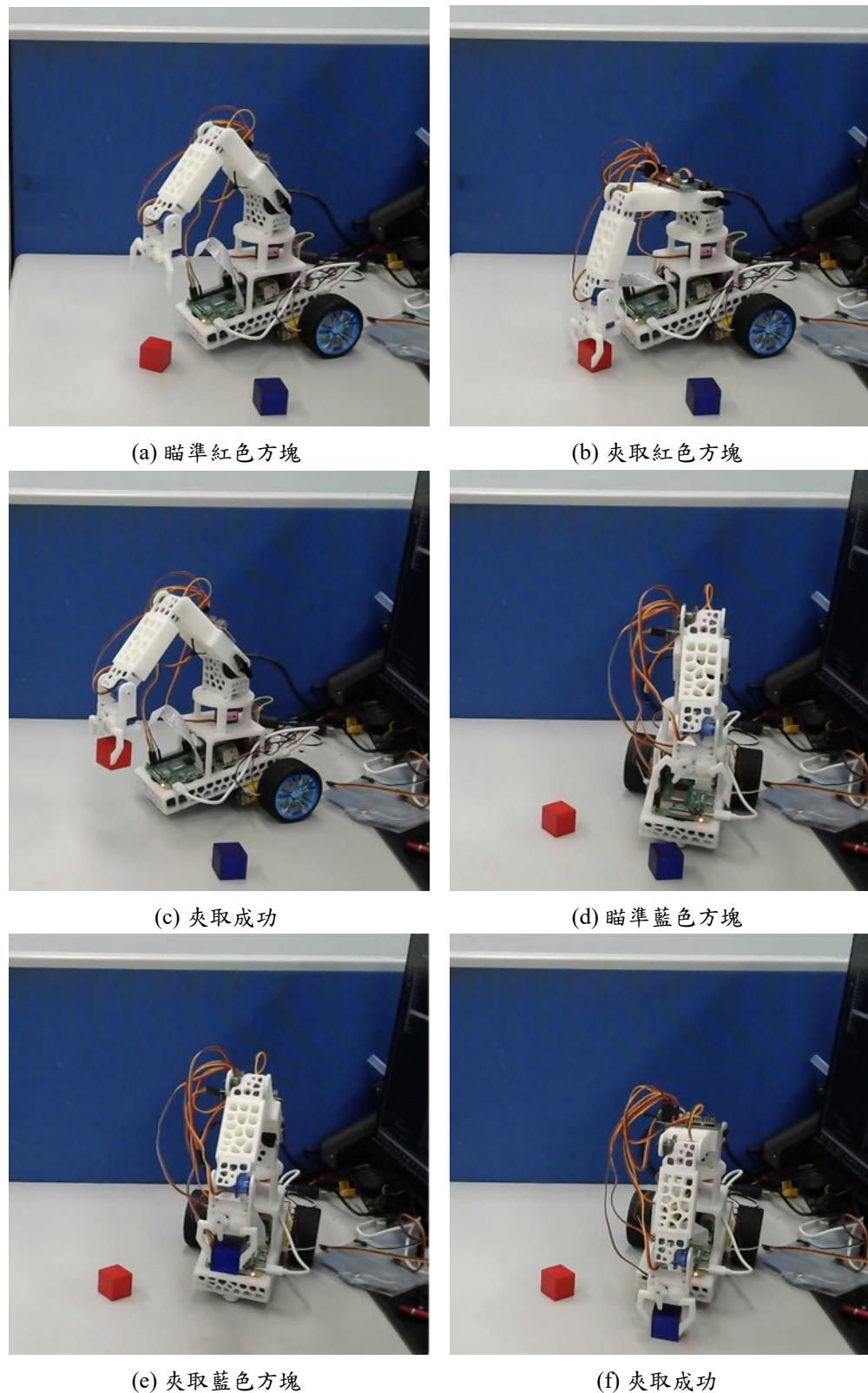


圖 4.19: 實驗三：實驗過程縮圖

## 五、總結

### 5.1 未來展望

# 參考文獻

- [1] D. Reed, D. Gannon, and J. Dongarra, *Reinventing high performance computing: Challenges and opportunities*, 2022.
- [2] W. X. Zhao, K. Zhou, J. Li, *et al.*, *A survey of large language models*, 2023.
- [3] Y. Liu, T. Han, S. Ma, *et al.*, “Summary of chatgpt-related research and perspective towards the future of large language models,” *Meta-Radiology*, vol. 1, no. 2, p. 100 017, Sep. 2023.
- [4] H. Touvron, L. Martin, K. Stone, *et al.*, *Llama 2: Open foundation and fine-tuned chat models*, 2023.
- [5] G. Team, R. Anil, S. Borgeaud, *et al.*, *Gemini: A family of highly capable multimodal models*, 2024.
- [6] Y. Feng, S. Vanam, M. Cherukupally, W. Zheng, M. Qiu, and H. Chen, “Investigating code generation performance of chatgpt with crowdsourcing social data,” in *2023 IEEE 47th Annual Computers, Software, and Applications Conference (COMPSAC)*, 2023, pp. 876–885.
- [7] K. Chowdhary and K. Chowdhary, “Natural language processing,” *Fundamentals of artificial intelligence*, pp. 603–649, 2020.
- [8] Y. Kim, Y. Jernite, D. Sontag, and A. Rush, “Character-aware neural language models,” in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, 2016.
- [9] T. Wolf, L. Debut, V. Sanh, *et al.*, “Transformers: State-of-the-art natural language processing,” in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, Q. Liu and D. Schlangen, Eds., Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45.
- [10] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019.
- [11] C. Cui, Y. Ma, X. Cao, *et al.*, *A survey on multimodal large language models for autonomous driving*, 2023.
- [12] R. Cai, Z. Song, D. Guan, *et al.*, *Benchlmm: Benchmarking cross-style visual capability of large multimodal models*, 2023.

- [13] J. Zhang and D. Tao, “Empowering things with intelligence: A survey of the progress, challenges, and opportunities in artificial intelligence of things,” *IEEE Internet of Things Journal*, vol. 8, no. 10, pp. 7789–7817, 2021.
- [14] I. Ahmed, G. Jeon, and F. Piccialli, “From artificial intelligence to explainable artificial intelligence in industry 4.0: A survey on what, how, and where,” *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5031–5042, 2022.
- [15] R. Cupek, J. C.-W. Lin, and J. H. Syu, “Automated guided vehicles challenges for artificial intelligence,” in *2022 IEEE International Conference on Big Data (Big Data)*, 2022, pp. 6281–6289.
- [16] T. Le Nguyen and T. T. H. Do, “Artificial intelligence in healthcare: A new technology benefit for both patients and doctors,” in *2019 Portland International Conference on Management of Engineering and Technology (PICMET)*, 2019, pp. 1–15.
- [17] M. Sarcar, K. M. Rao, and K. L. Narayan, *Computer aided design and manufacturing*. PHI Learning Pvt. Ltd., 2008.
- [18] C. Elanchezhian and G. S. Sundar, *Computer aided manufacturing*. Firewall Media, 2007.
- [19] G. Thyer, *Computer numerical control of machine tools*. Elsevier, 2014.
- [20] D. Pham and R. Gault, “A comparison of rapid prototyping technologies,” *International Journal of Machine Tools and Manufacture*, vol. 38, no. 10, pp. 1257–1287, 1998.
- [21] K. V. Wong and A. Hernandez, “A review of additive manufacturing,” *International scholarly research notices*, vol. 2012, 2012.
- [22] N. Shahrubudin, T. C. Lee, and R. Ramlan, “An overview on 3d printing technology: Technological, materials, and applications,” *Procedia Manufacturing*, vol. 35, pp. 1286–1296, 2019.
- [23] I. Hager, A. Golonka, and R. Putanowicz, “3d printing of buildings and building components as the future of sustainable construction?” *Procedia Engineering*, vol. 151, pp. 292–299, 2016.
- [24] S. K. Moon, Y. E. Tan, J. Hwang, and Y.-J. Yoon, “Application of 3d printing technology for designing light-weight unmanned aerial vehicle wing structures,” *International Journal of Precision Engineering and Manufacturing-Green Technology*, vol. 1, pp. 223–228, 2014.
- [25] M. Chinthavali, “3d printing technology for automotive applications,” in *2016 International Symposium on 3D Power Electronics Integration and Manufacturing (3D-PEIM)*, IEEE, 2016, pp. 1–13.
- [26] T. G. Papaioannou, D. Manolesou, E. Dimakakos, G. Tsoucalas, M. Vavuranakis, and D. Tousoulis, “3d bioprinting methods and techniques: Applications on artificial blood vessel fabrication,” *Acta Cardiologica Sinica*, vol. 35, no. 3, p. 284, 2019.

- [27] Z. Liu, M. Zhang, B. Bhandari, and Y. Wang, “3d printing: Printing precision and application in food sector,” *Trends in Food Science & Technology*, vol. 69, pp. 83–94, 2017.
- [28] P. Vaithilingam, T. Zhang, and E. L. Glassman, “Expectation vs. experience: Evaluating the usability of code generation tools powered by large language models,” in *Chi conference on human factors in computing systems extended abstracts*, 2022, pp. 1–7.
- [29] S. Vemprala, R. Bonatti, A. Bucker, and A. Kapoor, *Chatgpt for robotics: Design principles and model abilities*, 2023.
- [30] J. Liang, W. Huang, F. Xia, *et al.*, “Code as policies: Language model programs for embodied control,” in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 9493–9500.
- [31] S. Kucuk and Z. Bingul, *Robot kinematics: Forward and inverse kinematics*. INTECH Open Access Publisher London, UK, 2006.
- [32] E. Daucé and A. Dutech, “Inverse kinematics on-line learning: A kernel-based policy-gradient approach,” 2010.