國立臺灣科技大學
資訊工程所

碩 士 學 位 論 文

論文的中文題目

English title of the paper English title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the paperEnglish title of the

研 究 生：你的名字
學　　號：M12345678
指導教授：OOO

中華民國九九九年九月九日

中 文 摘 要

本文件依據國立臺灣科技大學論文格式建議撰寫，作爲撰寫學術文件時的起始範本。考量到現代多數使用者已習慣以 Markdown 撰寫筆記與報告，本模板設計的核心目標之一，是協助使用者從 Markdown 思維順利過渡至 LaTeX 的語法結構。

本文各章節將逐步介紹 Markdown 常見語法與其在 LaTeX 中的對應方式，並說明如何在排版過程中加入演算法（pseudo code）、表格（table）、圖片（figure）等常見結構，進一步引導讀者理解如交叉參照（`\ref`、`\nameref`）與文獻引用（`\cite`）等功能的使用方式與排版效果。

爲了協助讀者快速理解整體架構，模板亦於下方列出各主要組成檔案的功能說明：

- **environment.tex**：預先引入常用套件並設定整體排版風格。若需調整字型、行距或語言模式，可於此進行全域設定。請在 `main.tex` 中，於其他 `tex` 檔案引用之前先行載入。
- **`frontmatter.tex`**：管理論文前言部分，包括摘要、致謝、目錄、圖表清單等內容。各段落內容分別寫於 `front-matter/` 資料夾中，方便獨立撰寫與維護。
- **`main.tex`**：整體論文的編譯進入點，主要引入各章節（位於 `chapters/`）內容。在撰寫階段，若僅需編譯個別章節（例如 Chapter 2），建議將其他 `\include` 指令暫時註解，能有效縮短編譯時間。撰寫章節時亦可先略過 `frontmatter` 區段。

本模板不僅提供一份可立即套用的排版起點，也試圖在說明與範例中，協助使用者逐步熟悉 LaTeX 的語意標記邏輯，爲後續學術文件的撰寫奠定基礎。

Abstract

This document follows the thesis formatting guidelines recommended by National Taiwan University of Science and Technology. Recognizing that most users today are accustomed to Markdown-style writing, this template is designed to assist users in transitioning from Markdown to the structured syntax of LaTeX.

Each chapter introduces common Markdown expressions and their corresponding LaTeX syntax, including examples of pseudo code, tables, figures, and referencing techniques such as `\ref`, `\nameref`, and `\cite`. The aim is to provide both conceptual and practical understanding of academic typesetting.

To help users quickly understand the file structure, the following key components are highlighted:

- **`environment.tex`**: Loads essential packages and defines global formatting. Font styles, spacing, and language settings can be customized here. It must be included at the top of `main.tex` before loading any other content.
- **`frontmatter.tex`**: Manages the preliminary parts of the thesis, including the abstract, acknowledgments, table of contents, and figure/table lists. These sections are stored in the `front-matter/` directory for modular editing.
- **`main.tex`**: The entry point of the entire thesis compilation. It includes chapters from the `chapters/` directory. During early drafting stages, users may comment out unnecessary `\include` lines (e.g., unused chapters or front matter) to accelerate compilation.

This template aims not only to offer a functional starting point, but also to help users gain familiarity with the semantic and structural patterns of LaTeX, laying a solid foundation for producing high-quality academic documents.

致謝

對初學者來說，LaTeX 是一個學習曲線陡峭的工具。它不是開箱即用的軟體，而是需要花時間理解排版邏輯與語法規則，才能讓它穩定運作。

這份模板的整理，只是我在使用 LaTeX 的過程中，發現現有的工具有點不好用，就自己整理一下，把常用的排版邏輯與格式整理成一份比較順手的起點。若這份模板能幫你少踩幾個坑、節省一些時間，那眞是我的榮幸。

我希望這樣的基礎，可以讓後來使用這份模板的人，把更多心力花在內容本身——理清問題、設計實驗、打磨論述，產出眞正有價值的學術作品，而不是被格式困住、被技術分心。如果有一天你對這份模板熟悉了，也開始寫出自己的東西，那我期望當別人遇到困難的時候，可以給點幫助。這樣做不會花你太多時間，但對於剛開始的人來說，會有一個明確的指引。

目　　錄

圖　目　錄

表　目　錄

程式碼清單目錄

第一章　編譯設定與常用指令

編譯是一個常見的問題，這裡簡單描述一下：\input 行爲大概就是直接複製貼上那個 .tex 的內容，\include 比較像是鏈結進來，像是 C 的編譯器把 "a.o" 跟 "b.o" 合併起來的感覺。

一、 基本差異比較

- **\include**：用於章節級別的插入，會在插入前後自動加入換頁，適合用於引入整章內容（如第 1 章、第 2 章）。
- **\input**：用於片段內容的插入，如一段說明、一份表格或一段自訂命令，不會換頁。

二、 使用範例

假設我們有以下檔案結構：

```
main.tex
chapters/
   chapter1.tex
   chapter2.tex
   sections/def.tex
```

在 main.tex 中，我們這樣插入章節：

```
\include{chapters/chapter1}
\include{chapters/chapter2}
```

chapter1.tex 中，也可以插入子段落：\input{sections/def.tex}

三、 注意事項

- \include 不能巢狀使用。你不能在 chapter1.tex 裡再寫 \include。
- \include 支援 \includeonly 指令，可以只編譯特定章節，加快草稿時的速度。(但是要放在 \begin{document} Document 之前)
- 檔案名稱不需要加上 .tex 副檔名。
- 無論是 \include 或 \input，都必須寫在 \begin{document} 之後。

四、 建議使用方式

- 使用 \include 管理各章節（例如 Chapter 1, 2, 3）。
- 在章節中，使用 \input 引入表格、插圖說明、定義區塊、命令模板等。

五、 newcommand

最後解釋一下 \newcommand 的用途，通常用法是：

```
\newcommand{ <name> }[]{ <defintion> }
\newcommand{ <name> }{ <defintion> }
```

在 LaTeX 中，[] 跟 {} 的差別是 [] 是可選的參數，{} 則爲必要參數。

以本文件來說，main.tex 最上方的：

```
\ documentclass[a4paper,12pt,fontset=none]{report}
```

[a4paper,12pt,fontset=none] 是完全可以省略的。\newcommand 行爲大概是定義一個 Function，frontmatter.tex 最上方就先定義好中文摘要、Abstract、致謝顯示的標題，並且在 acknowledge.tex 有以下語法：

```
\chapter*{\enAbstractTitle}
\addcontentsline{toc}{chapter}{\enAbstractTitle}
\normalsize
```

其行爲就是：我想要宣告一個章節，並且這個章節不要編號，並且加入到目錄中 (toc = table of content, 就是指目錄)。然後使用 `\enAbstractTitle` 把前面定義好的章節標題引入進來。

第二章　Markdown 語法映射

在撰寫技術文件時，Markdown 與 LaTeX 是兩種常見的標記語言。Markdown 語法簡潔直觀，適用於快速記錄與網頁排版，而 LaTeX 則因其排版精細與學術用途而被廣泛使用於論文撰寫。下列將介紹 Markdown 中常見的語法元素，並說明其在 LaTeX 中的對應寫法。

一、 標題結構對應

Markdown 中以井字號（#）表示標題階層，從一個 # 到六個 # 分別對應第一至第六層標題。LaTeX 中則使用章節結構指令來表達對應層級，如下所示：

Markdown 標題語法	對應 **LaTeX** 指令
# Header 1	`\chapter{...}`
## Header 2	`\section{...}`
### Header 3	`\subsection{...}`
#### Header 4	`\subsubsection{...}`
##### Header 5	`\paragraph{...}`
###### Header 6	`\subparagraph{...}`

二、 清單結構

Markdown 支援有序與無序清單，分別使用數字與破折號（-）表示。在 LaTeX 中，對應爲 `enumerate` 與 `itemize` 環境：

有序清單 (Ordered List)

```
\begin{enumerate}
  \item 第一項
  \item 第二項
  \item 第三項
\end{enumerate}
```

1. 第一項
2. 第二項
3. 第三項

無序清單 (Unordered List)

```
\begin{itemize}
  \item 第一項
  \item 第二項
  \item 第三項
\end{itemize}
```

- 第一項
- 第二項
- 第三項

三、 引用區塊（**Blockquote**）

Markdown 使用 > 表示引用。在 LaTeX 中，可使用 quote 環境：

```
\begin{quote}
這是一段引用文字。
\end{quote}
```

> 這是一段引用文字。

四、 文字樣式

Markdown 提供簡易的文字樣式控制，以下爲其在 LaTeX 中的對應方式：

- 粗體：LaTeX 使用 `\textbf{Text}`
- 斜體：LaTeX 使用 `\textit{Text}`
- 底線：LaTeX 使用 `\uline{Text}`（需載入 `ulem` 套件）

五、 程式碼表示法（Code）

行內程式碼

- Markdown：`code`
- LaTeX：\texttt{code}

區塊程式碼　基本用法是 listings 套件輔助顯示：

```
\usepackage{listings}
...
\begin{lstlisting}[language=Python]
def hello():
    print("Hello, World!")
\end{lstlisting}
```

第三章　圖片與表格語法

在技術寫作與學術排版中，圖像與表格扮演著不可或缺的角色。Markdown 提供簡單直覺的語法以快速建立圖片與表格，而 LaTeX 則透過強大的浮動物件機制與格式控制，能夠呈現更高品質且可引用的版面效果。以下將介紹 Markdown 與 LaTeX 在圖片與表格處理上的語法差異與對應方法。

一、 排版邏輯

在 LaTeX 中，圖與表格通常被視爲「浮動體（float objects）」，也就是說它們不一定會出現在程式碼撰寫的確切位置，而是由排版引擎根據整體頁面美觀與空間安排自動決定擺放處。爲此，LaTeX 提供一組定位參數，來「建議」浮動體的擺放位置：

- `h`：放在此處（here）
- `t`：放在頁面頂部（top）
- `b`：放在頁面底部（bottom）
- `p`：放在單獨一頁（page of floats）

這些選項可組合使用，例如 `[htbp]` 表示「優先嘗試此處放置，否則嘗試頁首、頁尾或浮動頁」。

然而，若圖片或表格位置附近空間不足、頁面排版邏輯無法滿足條件，LaTeX 將自動「推遲」浮動體的呈現，導致圖表延後出現，有時甚至出現在相當遙遠的段落或頁面，造成閱讀混亂。

強制位置：`[H]` 的使用　若希望圖表 ** 完全出現在原地 **，不讓 LaTeX 自動移動，需使用 `float` 套件，並改用 `[H]` 選項（注意是大寫 H）：

```
\usepackage{float} % 加在 preamble

\begin{figure}[H]
```

```
  \centering
  \includegraphics[width=0.8\textwidth]{example.png}
  \caption{強制定位的圖片}
  \label{fig:fixed}
\end{figure}
```

此寫法會完全抑制浮動特性，強制圖片出現在當前位置，適合用於教學文章、逐步範例說明、或需圖文密切配合之場合。

使用 [H] 的注意事項　儘管 [H] 提供精準控制，但過度使用可能導致頁面佈局凌亂，因爲 LaTeX 排版引擎會喪失自動美化整體頁面的能力。因此，建議僅在下列情況使用：

- 教學文檔中，圖表需緊貼程式碼或說明。
- 小型圖片或表格不影響頁面排版。
- 實驗結果、逐步操作等需逐段插圖說明的場合。

綜合而言，[htbp] 提供彈性排版，而 [H] 則提供嚴格控制。依據文件性質與可接受的排版自由度選擇合適方式，將有助於產出既專業又可讀性良好的文檔。

二、 插入圖片

Markdown 的圖片語法採用驚嘆號與中括號及小括號結合，語法如下：

```
![替代文字](圖片網址或路徑)
```

在 LaTeX 中，插圖須透過 graphicx 套件輔助，建議使用 figure 浮動環境，範例如下：

```
\usepackage{graphicx} % 於環境加入
\begin{figure}[htbp]
  \centering
  \includegraphics[width=0.8\textwidth]{ntust.png}
```

```
  \caption{範例圖片說明}
  \label{fig:example}
\end{figure}
```

圖 **3-1** 範例圖片說明

上述語法說明如下：

- `figure` 爲浮動環境，控制圖像位置與標號。
- `width=0.8\textwidth` 控制寬度占比，可改用絕對尺寸。
- `caption` 提供圖說，會自動產生圖表編號。
- `label` 可搭配 `ref` 或 `autoref` 進行交叉引用。

建議圖片格式使用 PDF、EPS 等向量圖，若爲點陣圖則解析度應高於 300 DPI，以確保印刷品質。

三、 製作表格

Markdown 的表格語法簡潔，但僅支援基本資料排列與對齊：

```
| 名稱 | 數量 |
|------|------|
| 筆電 |  1   |
| 滑鼠 |  2   |
```

在 LaTeX 中，表格需透過 `tabular` 或 `table` 浮動環境撰寫。基本範例如下：

```
\begin{table}[htbp]
  \centering
  \begin{tabular}{|c|c|}
    \hline
    名稱 & 數量 \\
    \hline
    筆電 & 1 \\
    滑鼠 & 2 \\
    \hline
  \end{tabular}
  \caption{設備清單表}
  \label{tab:devices}
\end{table}
```

名稱	數量
筆電	1
滑鼠	2

表 **3-1** 設備清單表

欄位格式代碼說明如下：

- `l`：靠左對齊（left）
- `c`：置中對齊（center）
- `r`：靠右對齊（right）
- 加上 `|` 表示加上垂直分隔線

如需進一步美化表格，推薦載入 `booktabs` 套件，使用 `\toprule`、`\midrule`、`\bottomrule` 等專業排版指令取代 `\hline`。

延伸功能　若遇表格過寬或需要跨頁顯示，可結合以下套件進行擴展：

- `tabularx`：支援自動欄寬調整。
- `adjustbox`：可將表格縮放或旋轉。
- `longtable`：支援跨頁表格排版。

設備名稱	數量	備註
筆電	1	公司配發
滑鼠	2	自購
鍵盤	1	自購

表 **3-2** 美化過的表格

四、 圖片與表格尺寸控制對照表

爲提高排版彈性，LaTeX 提供多種尺寸單位與擴充指令供圖表配置使用。下表彙整常見語法與其用途：

語法	單位	說明與用途
`\textwidth`	寬度	整體文字區域的寬度，通常用於全文圖表寬度統一
`\linewidth`	寬度	當前區塊（如 minipage、list 環境）中的可用寬度
`width=...`	寬度	設定圖像或表格的最大寬度，單位可爲 `\textwidth`、`\linewidth`、cm 等
`height=...`	高度	設定圖像或表格的最大高度，常與 `keepaspectratio` 搭配使用以避免圖像變形
`keepaspectratio`	旗標	用於圖像縮放時保持寬高比例，避免變形
`\arraystretch`	倍數	用於拉高表格的行距，提升可讀性(預設值爲 1.0)
`adjustbox` 套件	縮放控制	可同時調整圖表的寬與高，並支援置中、旋轉等進階功能

表 **3-3** LaTeX 中圖表尺寸控制常用語法總覽

下面筆者提供一個簡單的 `\Image` 指令，分別使用 `label,imagePath,caption` 當作參數插入圖片。label 的命名，個人習慣使用 `fig:` 當圖片的前綴；`tab:` 當表格的前綴。`\label` 是一個很重要的語法，他可以建立一個可參考的位置。

```
\usepackage{graphicx}
```

```
\usepackage{caption}
\usepackage{float} % 若使用 [H]

% 定義 \Image 指令
\newcommand{\Image}[4][]{%
  \begin{figure}[H]
    \centering
    \includegraphics[#1]{#3}
    \caption{#4}
    \label{fig:#2}
  \end{figure}
}
```

第四章　插入程式碼片段與演算法

在技術與學術文件中，清晰展示程式碼與演算法邏輯是極爲重要的排版需求。LaTeX 提供了多種高品質工具來實現此目標，其中以 minted 以及 algorithm 搭配 algpseudocode 最爲常用。本章將說明兩者的使用方式與適用情境。

一、 使用 minted 插入語法高亮程式碼

minted 套件利用 Pygments 提供跨語言的語法高亮，能準確保留縮排與語意格式，適合展示原始碼片段。

使用前置設定

使用 minted 時，請於前言區引入套件：

```
\usepackage{minted}
\usepackage{xcolor}
```

並在編譯時使用：pdflatex -shell-escape main.tex

範例語法

```
\begin{listing}[H]
  \begin{minted}[<樣式設定>]{<選擇語言>}
  <程式碼內容>
  \end{minted}
  \caption{<說明>}
  \label{<標籤>}
\end{listing}
```

以下爲一段 C++ 程式碼片段的範例：

```cpp
#include <bits/stdc++.h>
using namespace std;
int binarySearch(int arr[], int low, int high, int x)
{
    while (low <= high) {
        int mid = low + (high - low) / 2;

        // Check if x is present at mid
        if (arr[mid] == x)
            return mid;

        // If x greater, ignore left half
        if (arr[mid] < x)
            low = mid + 1;

        // If x is smaller, ignore right half
        else
            high = mid - 1;
    }

    // If we reach here, then element was not present
    return -1;
}
```

範例代碼 **4-1** 二分搜尋法

可選參數說明如下：

- style：選擇風格，這裡使用 vs 風格
- linenos：顯示行號
- frame=lines：上下加框線
- bgcolor=gray!1：背景顏色
- fontsize=：字體大小

二、 使用 algorithm 插入演算法

若需表達邏輯流程、條件分支與步驟運算，建議使用 algorithm 套件搭配 algpseudocode，以排版接近 pseudo-code 形式展示演算法。

前置設定

請在 preamble 加入以下指令：

```
\usepackage{algorithm}
\usepackage{algpseudocode}
```

基本範例

```
\begin{algorithm}
    \caption{<說明>}
    \label{<標籤>}
    \begin{algorithmic}[1]
    <算法描述>
    \end{algorithmic}
\end{algorithm}
```

以下爲一個遞迴階乘演算法的範例：

Algorithm 1 階乘演算法

```
function Factorial(n)
    if n = 0 then
        return 1
    else
        return n × Factorial(n − 1)
    end if
end function
```

語法元素說明

- Function, If, Return, Call 爲結構性語句

- \State 用於表示單行操作
- [1] 可爲演算法加上行號

三、 關於 algorithm 的 \label 放置位置

在使用 algorithm 環境搭配 caption 產生標題與標號時，標籤（\label）的放置位置會影響引用的正確性。這與一般 figure 或 table 環境略有不同。

\label 必須放在 \begin{algorithm} 之後，且 * 緊接著放在 \caption 之後才有效產生正確的標號引用 *。只有 \begin{algorithm} 在 begin 後直接設定 caption。figure, table, listing 都在最後的 \begin{...} 前在設定 caption 與 label 就好。

```
\begin{algorithm}
\caption{演算法名稱}
\label{alg:example} % ← 要放在 caption 之後
\begin{algorithmic}
  ...
\end{algorithmic}
\end{algorithm}
```

常見錯誤： 若將 \label 放在 \begin{algorithm} 之前，或放在 \begin{algorithmic} 之後，將導致 \ref{alg:example} 引用失效或標號錯誤。

技術說明： 這是因爲 LaTeX 中的浮動體（如 figure、table、algorithm）之編號是在 \caption 處才產生的。若 \label 放在 caption 之前，當時該浮動體尚未取得編號，導致標號參照錯誤。

建議習慣： 爲確保一致性與可維護性，建議一律將 \label 緊接著放在 \caption 後，如下所示：

```
\begin{algorithm}
\caption{XXX 方法的流程}
\label{alg:xxx}
\begin{algorithmic}[1]
```

```
  \State ...
\end{algorithmic}
\end{algorithm}
```

如此能保證使用 \ref{alg:xxx} 或 \autoref{alg:xxx} 引用時正確顯示對應編號。

第五章　交叉引用與文獻引用機制

LaTeX 提供強大且嚴謹的交叉引用機制，可讓使用者在長篇文件中準確引用圖表、章節、演算法與外部文獻，維持內容一致性並提升可讀性。本章將介紹四種常見引用指令：\ref、\autoref、\nameref 與 \cite，並說明其使用情境與差異。除了\cite以外，這些都要跟 \label 語法進行搭配。

一、 標號引用：\ref

\ref 是最基本的標號引用指令，僅輸出對應編號本身，不包含任何類型前綴。使用者需手動加上詞彙（如「圖」、「表」、「演算法」）：

```
如圖~\ref{fig:architecture} 所示 ...
```

輸出結果範例：

> 如圖 3 所示...

此語法適用於細緻排版控制，但需使用者自行維持用詞一致。

二、 自動標號類型：\autoref

\autoref 來自 hyperref 套件，會根據標籤所屬環境自動加上類型前綴（如 Figure、Table、Algorithm 等），語法簡潔且一致性高：

```
\autoref{alg:factorial}
```

輸出結果：

> Algorithm 1

這對於長篇文件尤爲實用，能大幅減少人工錯誤，但前綴會依語言預設值(英文)，若需中文可使用 cleveref 套件替代。

三、 引用名稱：\nameref

\nameref（需載入 nameref 套件，或搭配 hyperref）用來輸出標籤對應的標題文字，非常適合自然語言式敘述：

詳見「\nameref{sec:evaluation} 」一節。

輸出結果：

> 詳見「效能評估」一節。

此語法對於強調章節標題內容而非編號的敘述情境特別有用，可與 \ref 或 \autoref 搭配使用。

四、 文獻引用：\cite

\cite 用於插入文獻來源標號，通常配合 BibTeX 或 BibLaTeX 資料庫管理文獻。引用格式與排序會由所選格式決定，例如：

如文獻~\cite{webrtc-overview} 所述 ...

- 如 RFC3350 [1] 所述...
- 參考 Gstreamer [2] 網站...
- 根據 SRT [3] 協定...
- 此篇論文 [4] 指出...

這樣就可以引用參考資料了。在本模板中，已經使用 \addbibresource{references.bib} 把 references.bib 加入參考來源了，只需要把對應的引用格式丟入該檔案，並確保 main.tex 的最下方有使用 \printbibliography[heading=bib] 渲染參考文獻即可

參 考 文 獻

[1] Henning Schulzrinne et al. *RTP: A Transport Protocol for Real-Time Applications*. RFC 3550. July 2003. doi: 10.17487/RFC3550. url: https://www.rfc-editor.org/info/rfc3550.

[2] GStreamer Project. *GStreamer GitLab Repository*. Accessed on GitLab. 2025. url: https://gitlab.freedesktop.org/gstreamer (visited on 04/16/2025).

[3] Haivision. *SRT Protocol Specification 1.4*. Technical Specification. Haivision, 2018. url: https://www.srtalliance.org/specifications (visited on 04/16/2025).

[4] D. T. Hoang et al. "A dynamic edge caching framework for mobile 5G networks". In: *IEEE Wireless Communications* 25.5 (2018). https://ieeexplore.ieee.org/document/8443597, pp. 95–103. doi: 10.1109/MWC.2018.1700360.

附錄 A：XXX

附錄 B：XXX