國立交通大學

資訊學院資訊科技（IT）產業研發

碩士專班

碩 士 論 文

六子棋遊戲之設計與研究在 Android 行動裝置

The Study and Development of Connect6 Game for Android Devices

研 究 生：鄭吉閎

指導教授：吳毅成　教授

中 華 民 國　101　年 9 月

六子棋遊戲之設計與研究在 Android 行動裝置

The Study and Development of Connect6 Game for Android Devices

研 究 生：鄭吉閎　　Student：Ji-Hong Zheng

指導教授：吳毅成　　Advisor：I-Chen Wu

國 立 交 通 大 學

資訊學院資訊科技（IT）產業研發

碩 士 論 文

A Thesis

Submitted to College of Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Industrial Technology R & D Master Program on

Computer Science and Engineering

September 2012

Hsinchu, Taiwan, Republic of China

中華民國 101 年 9 月

六子棋遊戲之設計與研究在 Android 行動裝置

研究生：鄭吉閎　　指導教授：吳毅成

國立交通大學 資訊學院產業研發碩士專班

摘要

六子棋是 2005 年由吳毅成教授所發明的一種棋類遊戲，近年來已經發展成為世界性的遊戲。此外，隨著硬體技術的進步與嵌入式系統的發展，智慧型手機與平板電腦的運算及功能愈來愈強大，而搭載 Android 系統的智慧型手機與平板電腦，更是在市場中佔有舉足輕重的地位，

為了推廣六子棋，我們在 Android 系統行動裝置上開發六子棋程式。這篇論文提出一個棋類遊戲框架並透過此框架實作六子棋程式。此框架讓遊戲開發者能夠加速開發棋類遊戲，同時遊戲開發者可以更專注於遊戲內容的設計。

此六子棋程式除了完成基本的對弈功能與友善的使用者介面之外，還支援由本實驗室所研發的六子棋人工智慧程式── NCTU6。此外，我們收集大量的詰棋盤面與開局，讓使用者能達到自我訓練的目的。我們相信使用者透過此六子棋程式能夠更了解六子棋。

The Study and Development of Connect6 Game for Android Devices

Student: Ji-Hong Zheng Advisor: I-Chen, Wu

Industrial Technology R & D Master Program of
Computer Science College
National Chiao Tung University

Abstract

Connect6, first introduced by Professor I-Chen Wu in 2005, has become a popular game that is played around the world. Meanwhile, smartphones and tablets have become more powerful with advances in hardware technology and the development of embedded systems. Android operating system devices are popular for the smartphone and tablet market.

In order to further popularize Connect6, we designed and developed a Connect6 application on Android devices. In this paper, a generic framework for board games was designed, and the Connect6 application is based on this framework. Developers only need to concentrate on game-related implementation, so they can implement board games more easily and efficiently.

In this Connect6 application, we provide the basic features of Connect6 and a friendly user interface. Additionally, the application supports NCTU6, a Connect6 AI program developed by our laboratory. We also collected a large number of puzzles and openings. Users are able to train themselves by solving puzzles. We believe that users can familiarize themselves with Connect6 through this application.

誌謝

這篇論文的完成，首先我要感謝我的指導教授吳毅成教授，教授日以繼夜地陪伴我做研究，也給了我諸多相關建議，甚至教導我許多研究方面的態度，使得本篇論文的研究得以有如此的結果。同時也必須感謝口試委員們，在我的研究方面給予諸多指導，我才能順利完成這篇論文。

感謝群想在美工上的支援，設計了六子棋應用程式的遊戲畫面，與計畫的討論與建議。

感謝尊博科技股份有限公司，在就學期間每月提供生活費，還有主管及同事的關心慰問，讓我非常期待未來投入職場的生活。

感謝王智功，透過您的專業知識協助分析六子棋詰棋譜的難度分級。

感謝我的父母，謝謝你們從小辛苦教養我，在我求學的路上給我支持，教導我人生中待人處事的規矩。

感謝盧玉琴主任，過去在高雄市樂群國小服務的日子裡，因您的鼓勵讓我重拾學習的樂趣與動力，並在這二年的碩士生涯中時時刻刻給予我關懷與信心。

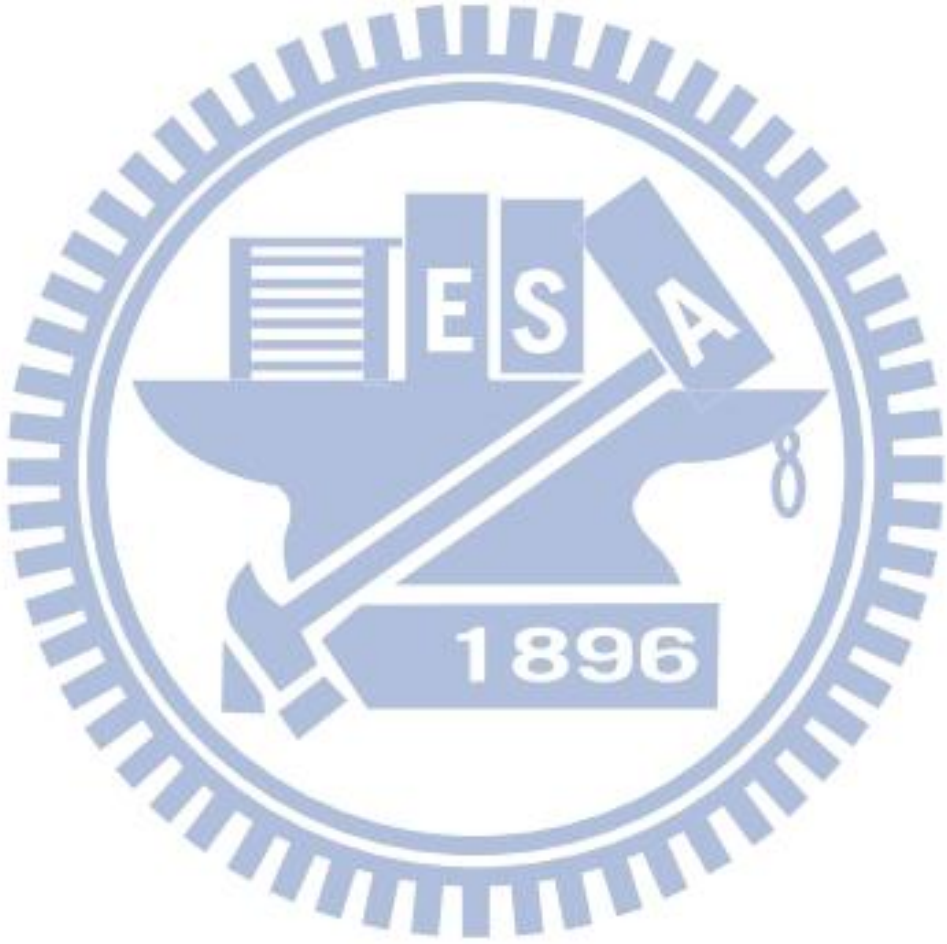
感謝實驗室的學長。林宏軒學長幫忙我檢查論文內容，也給了我很多程式上以及論文方面的建議。

感謝我的同學與學弟們。康皓華與張傑閔幫忙我了解六子棋 AI 與調整 AI 的強度，讓我能快速修改並移植到 Android 平台上。胡嘉芸一起討論論文與激勵彼此。廖挺富在程式上幫忙我一起除錯。魏經軒與劉浩雲在程式上給予我一些建議。

最後，僅以此篇論文獻給所有陪伴我的家人、老師、同學和朋友們，謝謝你們。

民國一百零一年九月　於　新竹交大工程三館 EC511 實驗室

目錄



圖表目錄

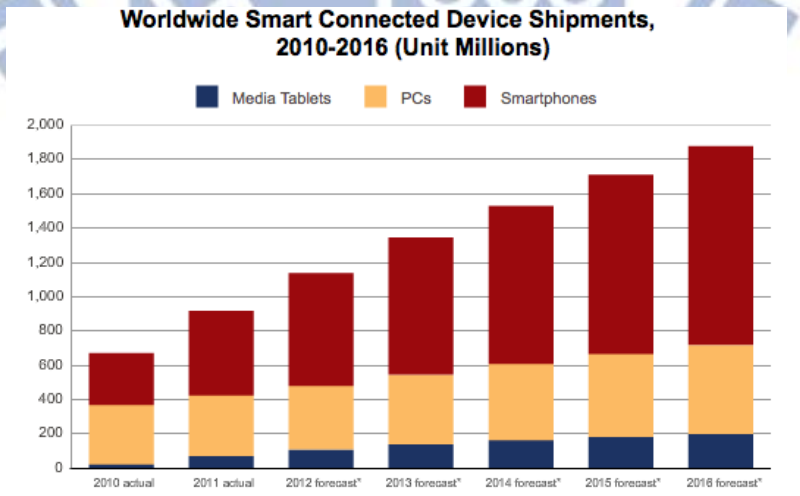
表格目錄

第一章、介紹

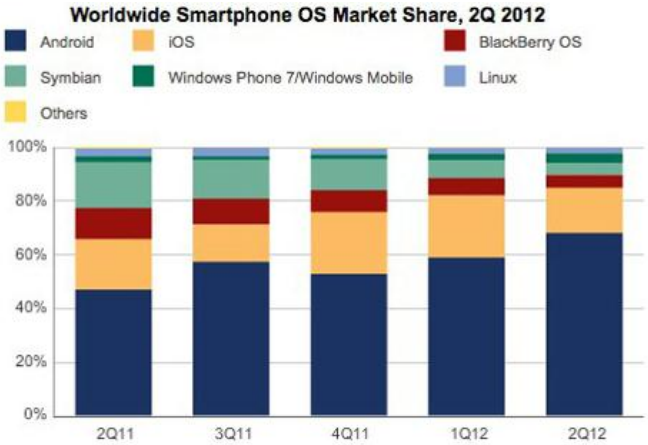
在一開始的章節中，會來介紹一些背景知識。第 1.1 節介紹近年來行動裝置的市場狀況，第 1.2 節介紹六子棋，第 1.3 節提出研究的動機和目的，節簡單整理此篇論文的貢獻，第 1.4 節介紹整篇論文的架構。

1.1 行動裝置的市場狀況

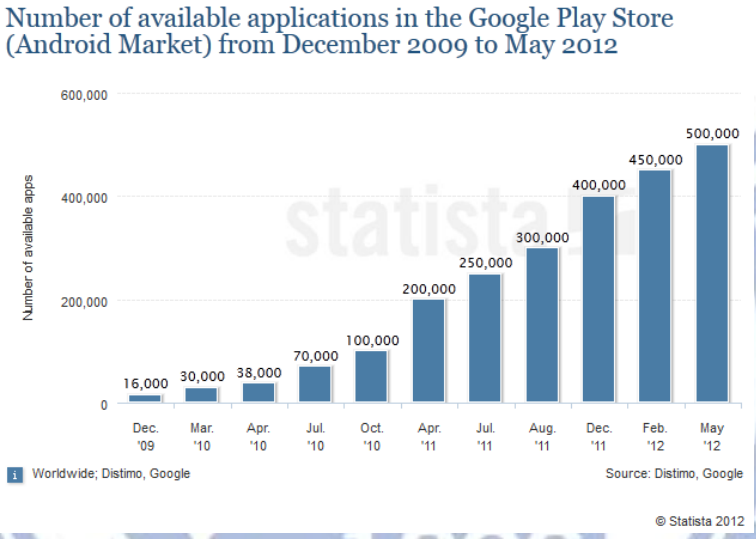
近年來隨著硬體製程技術的進步及嵌入式系統的發展，市面上陸續出現運算及功能強大的智慧型手機和平板電腦，這些裝置的作業系統就如同一台小電腦，使用者可以在上面安裝各種應用程式，為了便於攜帶，和傳統手機體積相仿並且在操作上力求簡單，為人類帶來很多便利。在此同時市場也反應出人們對行動裝置的需求愈來愈大，圖一[12]為全球個人電腦、智慧型手機與平板電腦的出貨量比，可以看出在 2011 年智慧型手機的出貨量已超越個人電腦的出貨量。



圖一. 2010-2016 全球智慧型裝置出貨量統計圖（來源：IDC[12]）



圖二. 全球智慧型手機作業系統市佔率統計圖（來源：IDC[11]）



圖三. Google Play Store 所提供應用程式數量統計圖（來源：Statista[17]）

搭載 Android 系統的智慧型手機在這龐大的行動裝置市場中，佔有舉足輕重的地位，圖二[11]為全球搭載不同作業系統的智慧型手機的市佔率統計圖，在 2012 年第二季，Android 手機的市佔率達到全部的 68%。

隨著 Android 市佔率上升，使用者對應用程式的需求也跟著上升，根據 Google Play[8]（前身是 Android Market，在 2012 年 3 月更名，為 Android 系統裝置的線上應用程式商店）統計，2012 年 5 月應用程式的下載量累積總量突破 150 億[2]，圖三[17]為 Google Play Store 提供應用程式的數量統計圖，圖中顯示 2012 年 5 月 Google Play 所提供應用程式數量到達 50 萬。

1.2 六子棋

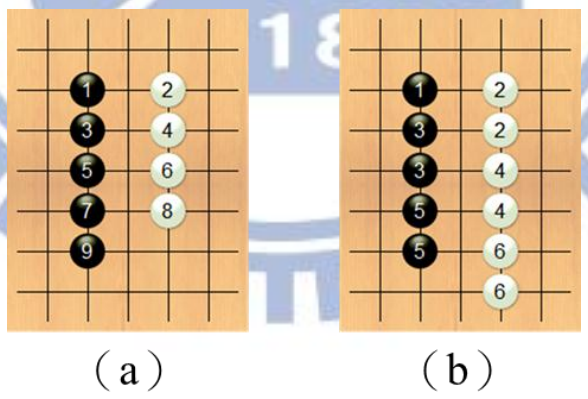
六子棋（Connect6）[22][25]是交通大學吳毅成教授在 2005 年 9 月發表於第十一屆電腦賽局發展學術研討會（ACG 11）的一個遊戲[13] [18]。六子棋是由五子棋改良而成，它有三個重要特性─規則簡單、遊戲公平、以及變化複雜。

- 規則簡單：

  六子棋遊戲使用十九路棋盤，先手持黑，下一顆子，接著雙方輪流各下兩子，先連成六子或更多者為勝。若下滿棋盤後仍未有玩家獲勝，則為和局。

- 遊戲公平：

  以五子棋而言，無論黑方或白方下子後，盤面中的黑子都不會少於白子（圖四（a））。以六子棋而言，每一手下完後，皆會較對方多一顆子（圖四（b）），因此具備潛在公平性的特質。



圖四.（a）五子棋棋局（b）六子棋棋局

- 變化複雜：

  六子棋的複雜度是介於日本將棋與象棋之間（表一）。由於六子棋一手可以下兩顆子，每顆子平均約有 300 個位置可以下，因此一手的變化

大約有 45,000 種。一局平均約 30 手到 40 手，可推出六子棋的複雜度約為 10^{140}~10^{188}。

表一. 遊戲複雜度

遊戲名稱	複雜度
Go （圍棋）	10^{360}
Shogi （將棋）	10^{226}
Connect6 （六子棋）	**$10^{140\sim188}$**
Chinese Chess （象棋）	10^{150}
Chess （西洋棋）	10^{123}
Go-Moku （五子棋）	10^{70}

1.3 動機與目的

六子棋發展至今獲得許多回響，除了 1.2 所述，還包括成立六子棋協會，舉辦六子棋公開賽等。我們希望能繼續研究及推廣六子棋，近年來行動裝置的便利與流行，Android 裝置與應用程式在市場中也佔有領先的地位（如 1.1 所述），因此，我們實作六子棋在 Android 上，並配合實驗室所發展的六子棋人工智慧程式，使得六子棋能有更多方面的擴展。

預期成功設計與實作一個 Android 系統的棋類遊戲框架，並透過此框架成功的實作出六子棋遊戲。

1.4 論文組織

本篇論文第二章是研究背景，在研究背景中，會介紹論文中所運用到相關的技術。第三章是設計與實作，會說明我們遊戲框架的設計與實作方法。第四章是實作結果，在該章節中會呈現我們實作六子棋的成果。第五章是結論以及未來展望。

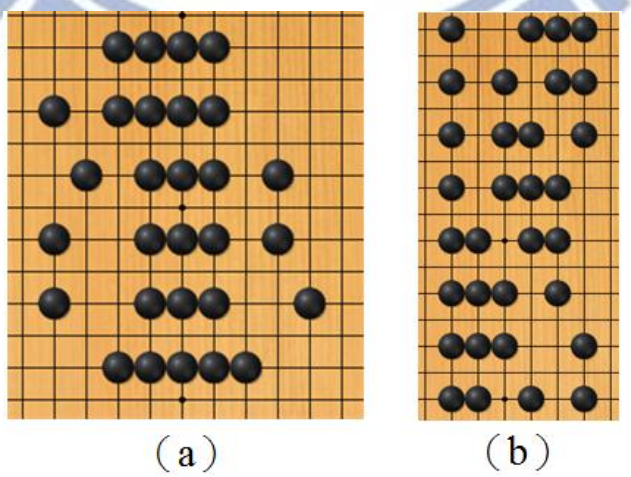
第二章、研究背景

在這個章節中將介紹本論文相關技術之背景知識，首先在 2.1 會介紹六子棋，第 2.2 節介紹 NCTU6，第 2.3 節介紹 Mobile6，第 2.4 節介紹 Android，第 2.5 節介紹 Android 系統架構，第 2.6 節介紹 Android-NDK，第 2.7 節介紹 SQLite。

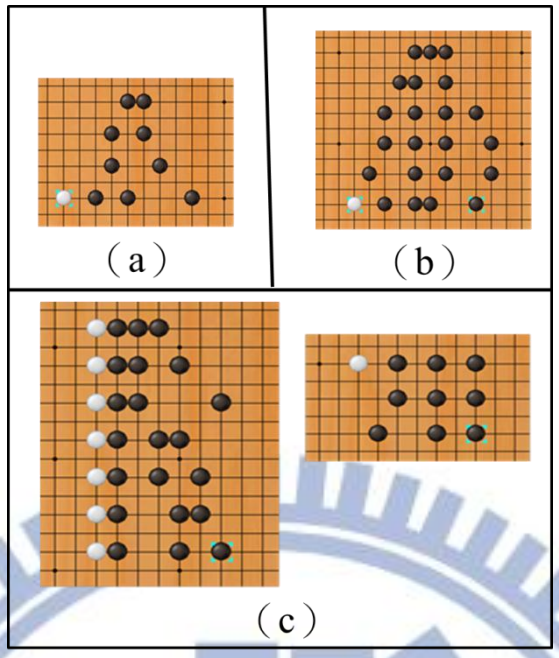
2.1 六子棋

迫著（Threats）的定義：若一方需要下 N 顆子來避免另一方連成 6 顆子以上，則稱該方有 N 個迫著。圖五.（a）雙迫著（Double Threats）（b）單迫著（Single Threat），對六子棋來說，贏的策略就是阻擋所有對方的迫著，並同時產生三個或以上的迫著。

迫著衍生出類似五子棋中的活三、活四、死三、死四的特性，而六子棋中，有活二、活三、死三，活四包含在雙迫著，死四包含在單迫著，如圖六所示；定義如下：若再下（4-T）顆子，就可以產生一個迫著，則為死 T 迫著；若僅再下（4-T）顆子，就可以產生兩個迫著，則為活 T 迫著。



圖五.（a）雙迫著（Double Threats）（b）單迫著（Single Threat）



圖六.（a）活二（b）活三（c）死三

2.2 NCTU6

NCTU6 是交通大學吳毅成教授實驗室所開發的六子棋 AI 程式，又稱為交大六號，從 2006 年開始至今參加過許多競賽[15][21][23][24][26]，獲得不少獎項，如表二。

表二. NCTU6 戰績表

時間	比賽	NCTU6 結果
2006	第十一屆國際奧林匹亞電腦賽局競賽六子棋組	冠軍
2008	第十三屆國際奧林匹亞電腦賽局競賽六子棋組	冠軍
2008	世界棋王周俊勳與電腦六子棋對抗賽	3 勝 0 負
2008	第一屆人腦對電腦六子棋大賽	11 勝 1 負
2009	第二屆人腦對電腦六子棋大賽	8 勝 0 負
2011	第三屆人腦對電腦六子棋大賽	5 勝 3 負

2.3 Mobile6

Mobile6 是由 NCTU6 改良而成，目的是為了將 NCTU6 移植至智慧型手機應用程式中，由於目前手機不比個人電腦，在 CPU 運算、記憶體容量，都有相當大的差距，因此，Mobile6 是我們限制 NCTU6 中演算法的搜尋深度、記憶體使用量與運算時間的版本，至今參加過 Little Golem 網站[16]上舉辦的 Bi Lao Da–13（比老大賽之十三）得到冠軍，與 TCGA2012 六子棋競賽得到亞軍。

2.4 Android

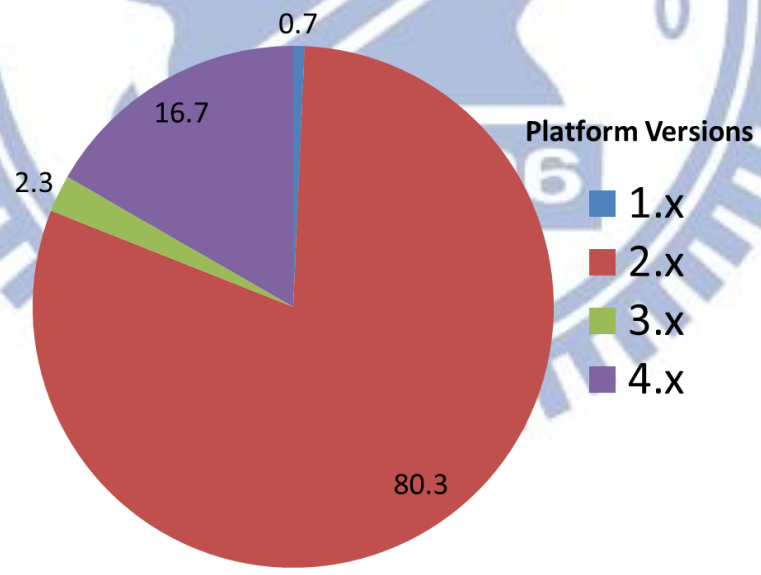
Android[6]是 Linux-based 作業系統，專為移動裝置設計，並以 JAVA 作為應用程式開發語言。

Android 科技公司在 2003 年 10 月由安迪・魯賓（Andy Rubin）成立，2005 年 Android 科技公司被 Google 收購。在 2007 年 Google 成立開放手持設備聯盟（Open Handset Alliance）當時成員包括 Broadcom, HTC, Intel, LG, Marvell 等公司。而後在 2008 年，ARM, Huawei, Sony 等公司也加入聯盟，同時對外展示第一支搭載 Android 1.0 平台的智慧型手機，HTC Dream（G1）。

Android 發展至今，發表了許多版本的平台（表三），根據 Google Play（原名為 Android Market，於 2012 年更名，是 Google 提供 Android 裝置開發的線上應用程式商店）統計，我們將各版本 Android 平台分佈區分成四類如圖七，我們發現 2.x 以上的版本平台佔目前所有平台的 99.3%。

表三. Android 各平台發表時間

發表時間	Android 平台版本
2009 年　4 月	1.5
2009 年　9 月	1.6
2009 年 10 月	2.0/2.0.1/2.1
2010 年　5 月	2.2/2.2.1
2010 年 12 月	2.3
2011 年　2 月	3.0.1/3.1/3.2
2011 年 10 月	4.0
2012 年　6 月	4.1

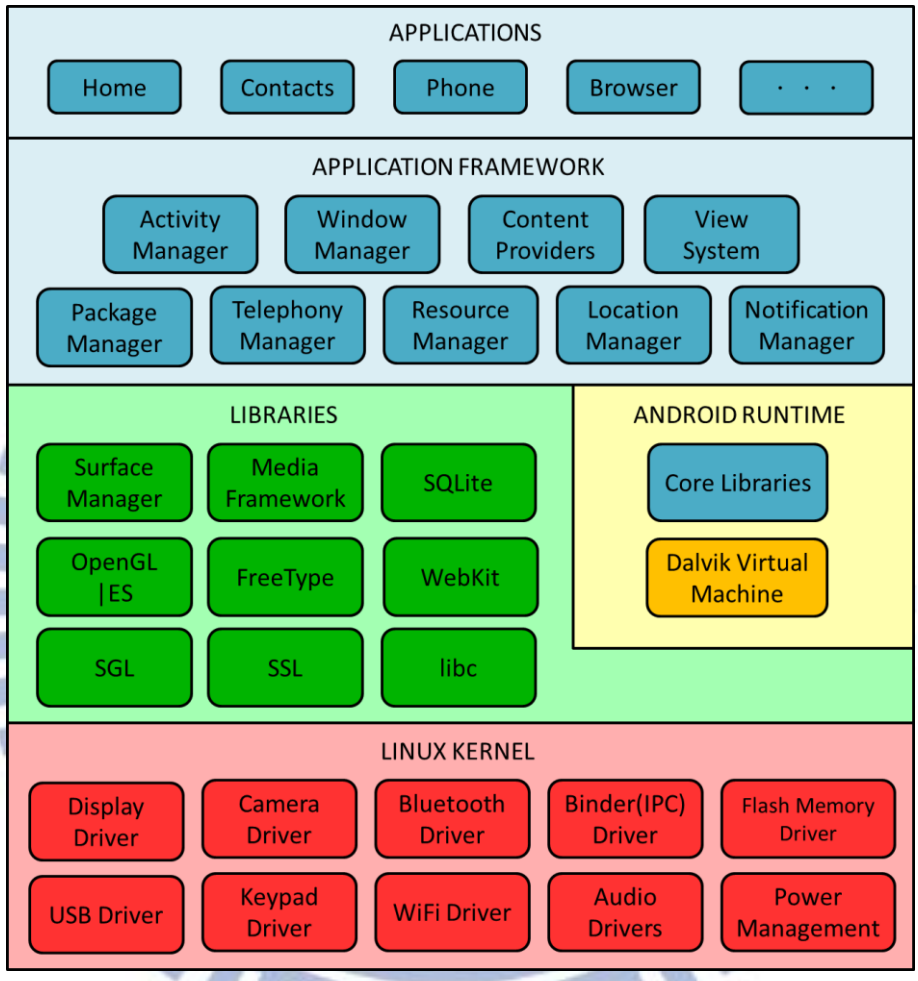


Platform Distribution
（Source[Android Developers.]）

圖七. Android 平台版本分佈統計

2.5 Android 系統架構

Android 系統架構分成 Applications、Application Framework、Libraries、Android Runtime 與 Linux Kernel，如圖八所示，並分述如下。



圖八. Android 系統架構

Applications

Android 附帶一套 JAVA 語言編寫的核心應用程式，包括電子郵件、簡訊程式、日曆、地圖、瀏覽器、聯繫人...等。讓使用者一開始就能使用一支手機應有的基本功能。

Application Framework

Android 提供開發人員一個開發平台,讓開發人員能夠完整使用與核心應用程式相同的 APIs,應用程式架構是為了要簡化元件重新利用而設計,應用程式可以發佈其功能並被其它應用程式所使用,使用者可以利用同樣的機制來置換元件。

Libraries

Android 包含一組 C/C++函式庫,這些函式庫被 Android 系統各個元件使用,而這些元件功能透過 Android Framework 給應用程式開發者加以使用。

Android Runtime

此部份包含 Core Libraries 與 Dalvik Virtual Machine,在 Core Libraries 中提供大多數 Java 所需要使用的函式。

Dalvik Virtual Machine 是一種暫存器型態的虛擬機器,且執行.dex 格式的檔案,此類型檔案在撰寫開發時就已經設想用最少的記憶體資源來執行。每一個 Android 應用程式都有屬於自己的程序。而且 Android 不是用一個 Dalvik Virtual Machine 來同時執行多個 Android 應用程式,而是每個 Android 應用程式都用一個屬於自己的 Dalvik Virtual Machine 環境來執行。

Linux Kernel

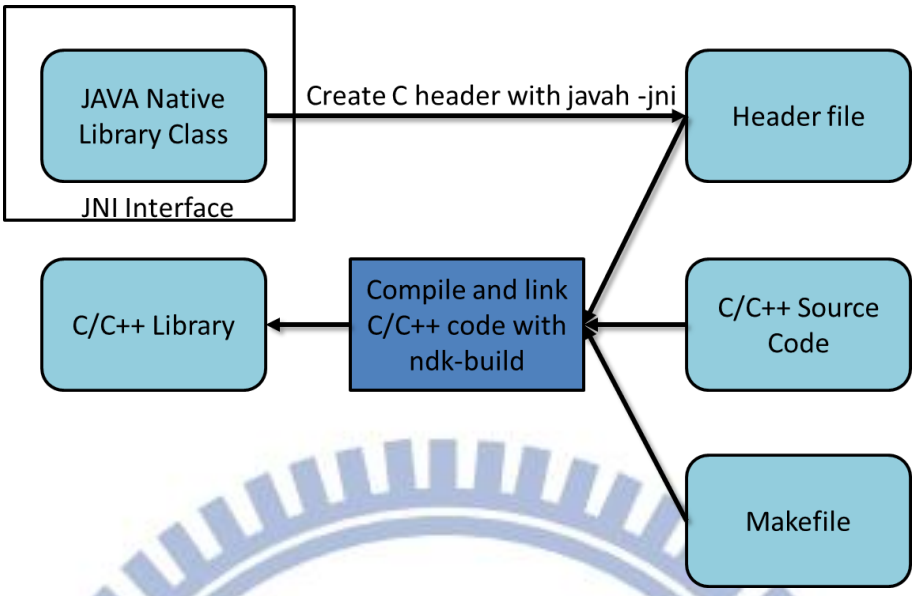
Android 的作業系統是 Linux。所提供的核心系統服務有 Security、Memory Management、Process Management、Network Stack 與 Driver Model。

2.6 Android-NDK

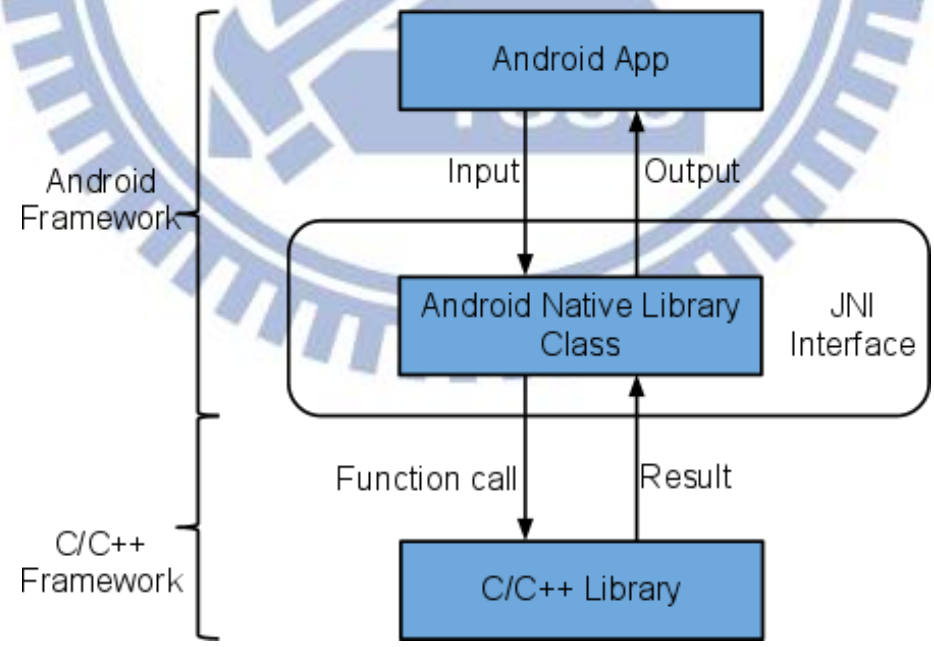
Android NDK （Native Development Kit）[7]是 Android 提供一套工具讓開發人員能夠在應用程式中嵌入 C/C++所開發的元件，由於 Android 應用程式是在 Dalvik Virtual Machine 上執行，透過 NDK 可讓應用程式使用 C/C++實作，優點是能夠重用過去已開發的程式與某些狀況下能夠加快應用程式的速度[1][3][14]。

Android NDK 讓開發人員能夠把 C/C++的程式碼編成 Dalvik Virtual Machine 能執行的 native library，並提供一套流程讓開發人員能夠將 native library 與應用程式的 package file 結合在一起，但 Android NDK 有開發上的限制，只能運行於 Android 1.5 以上版本的平台，這是因為開發嵌入式系統工具在 Android 1.5 以後的版本有變動，造成 Android 1.0 與 1.1 的平台不適用。

Android NDK 開發流程如圖九。首先，開發人員需將 JAVA Native Library Class 透過 javah 工具產生 C/C++的標頭檔（Header file），之後，開發人員需撰寫 Makefile 檔，告訴 NDK 工具，”需編譯哪些文件”、”目標函式庫名稱”等，再透過 build-ndk 工具將 C/C++程式碼編譯成 C/C++ Library（.so），完成後，應用程式中的屬於 JAVA 程式碼就可以與 C/C++程式碼溝通，如圖十所示。



圖九. Android NDK 開發流程



圖十. Android 與 C/C++ Library 溝通方式

2.7 SQLite

SQLite[9]是個實作 SQL 資料庫引擎的函式庫具有以下特性：

- Self-Contained:

  只需要非常少量外部函式庫或作業系統的支持。

- Serverless:

  不需要透過伺服器存取資料庫，SQLite 可以直接存取資料庫於本機儲存裝置。

- Zero-Configuration:

  SQLite 在使用前不需特別安裝。所以不需要啟動或停止伺服器程序。

- Transactional:

  SQLite 是一個符合 ACID[19]特性的關聯式資料庫。ACID 特性如下所示：

  - Atomicity:

    一個事務(transaction)的所有操作一定是全部完成或全部沒完成，若事務在執行過程中發生錯誤會回到事務執行前的狀態。

  - Consistency:

    在事務開始前和結束後，資料庫的完整性限制不會被破壞。

  - Isolation:

    兩個事務的執行互不干擾，事務時間不會相互影響。

  - Durability:

    在事務完成後，該事務對資料庫所造成的更改便持久保存在資料庫中，並且是完全的。

目前 SQLite 已應用在許多項目當中，例如：Mozilla Firefox, websites run PHP, Skype, Symbian smartphones, McAfee, iPhone, Android... 等。

第三章、設計

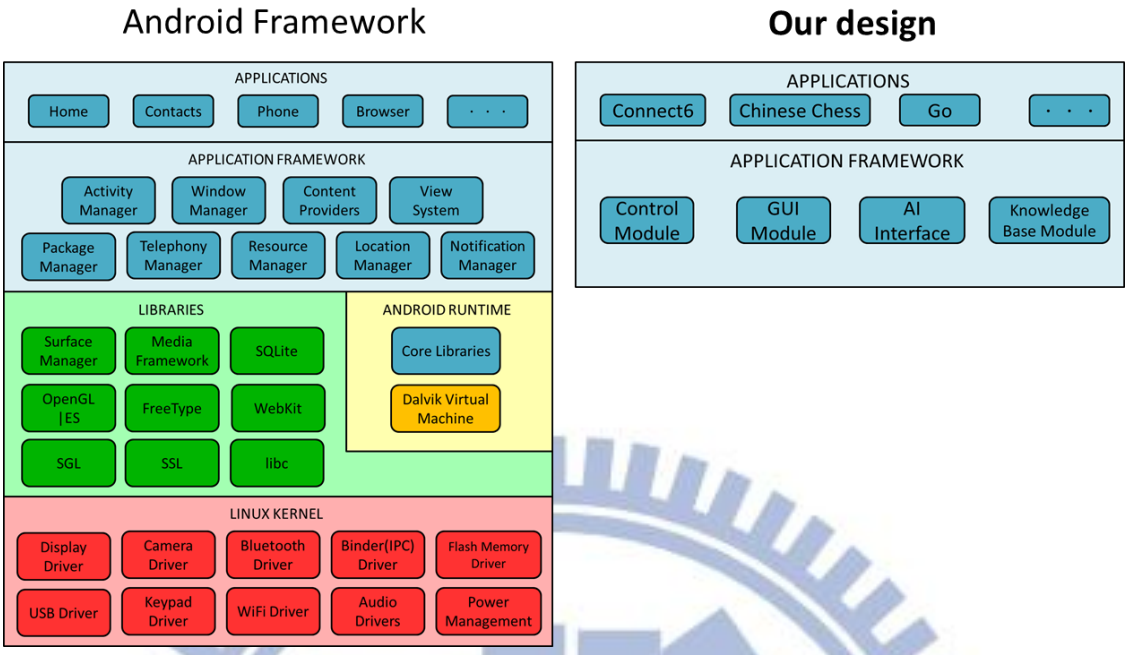
根據前述的動機與背景介紹，在這章節中會詳細說明框架架構，在第 3.1 節中說明整體框架架構，包含 Control Module, GUI Module, AI Interface, and Knowledge Base Module。在第 3.2 節中，會說明 Control Module 的設計。在第 3.3 節中，會說明 GUI Module 的設計。在第 3.4 節中，會說明 AI Interface 的設計。在第 3.5 節中，會說明 Knowledge Base Module 的設計。

3.1 整體架構

我們分析多數棋類遊戲的特性，與實作於 Android 裝置可能產生的需求，得到以下結論：

- 友善的使用者介面，包括可縮放的棋盤、悔棋、下子輔助線與迫著提示線等功能。
- 支援以 C/C++開發的人工智慧程式。
- 支援大量詰棋與開局。
- 程式的部建能夠符合 Android 各種不同的裝置與平台。

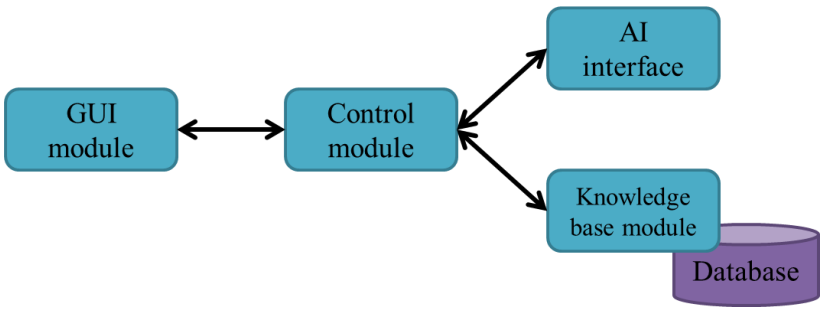
根據以上結論，設計了一個適用於 Android 系統的棋類遊戲框架，並透過此框架實作六子棋遊戲，以 Android 系統架構的角度來看，我們的設計屬於 Application Framework 的層級，如圖十一所示。



圖十一. Android 系統架構與棋類遊戲框架比較圖

整個框架分成四個模組，分別是 Control module, GUI module, AI interface 與 Knowledge base module，如圖十二所示，Control module 負責應用程式流程控制，GUI module 負責應用程式的輸入與輸出，AI interface 負責對 AI 程式的訪問，Knowledge base module 負責存取大量的詰棋與開局，在 3.2 節至 3.5 節會詳細說明各模組的設計。

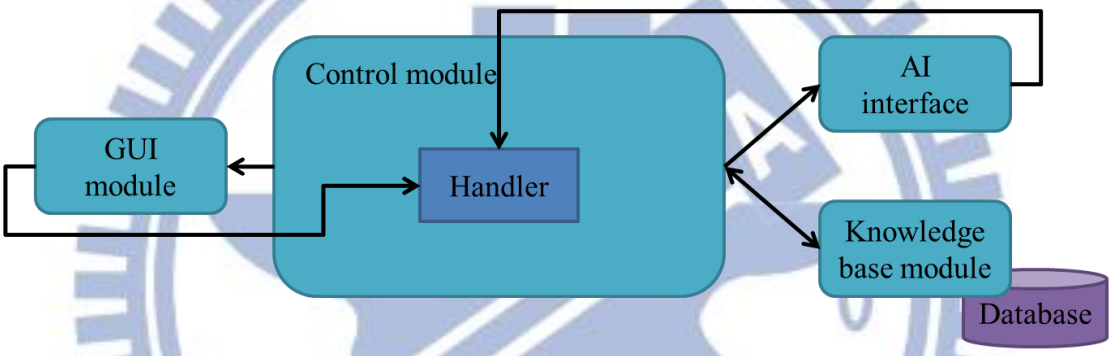
此外，由於 Android 平台有可向下相容但無法向上相容的特性，為了讓多數 Android 平台可以使用此框架所開發之應用程式，因此，以 Android 2.1 平台作為開發目標，動機如 2.4 節所述（Android 2.x 以上平台佔 99.3%）。



圖十二. 框架架構

3.2 Control Module 設計

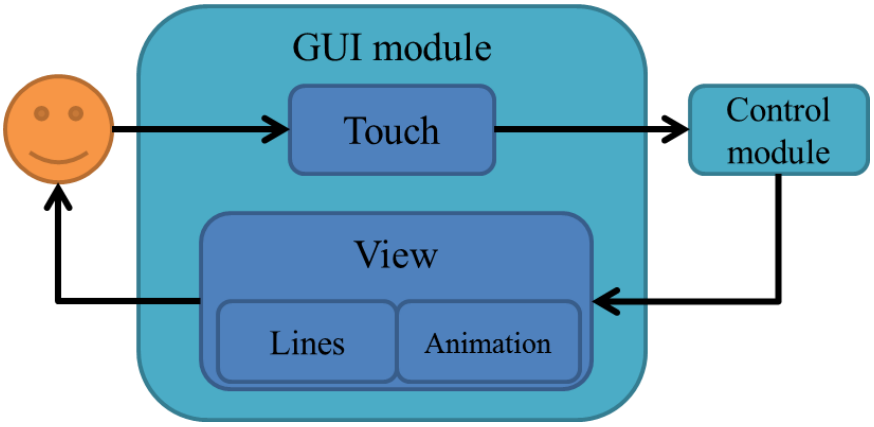
Control module 主要控制應用程式的流程，與傳遞模組之間的訊息（圖十三），由於各種棋類的規則與流程皆不同，因此，流程的控制交由遊戲開發者實作，在 Control module 的設計中主要開啟對各模組間的溝通，因此，在 Control module 中有一個 Handler 元件，負責接收各模組傳回來的訊息，根據不同的訊息結合開發者制定的流程，再交由 Control module 傳遞相對應的訊息給其它模組。



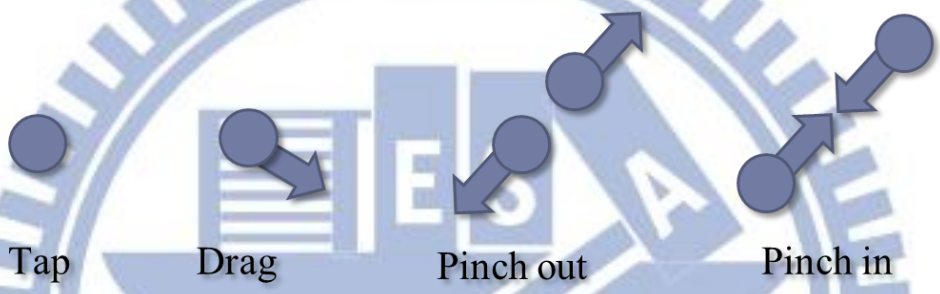
圖十三. Control module 架構圖

3.3 GUI Module 設計

GUI module 中有二個主要的元件，如圖十四所示，Touch 主要負責接受使用者輸入的手勢（Gesture）並將手勢轉成相對應的訊息傳送給 Control module；View 負責顯示遊戲的畫面，在 View 中還有二個元件，Animation 元件負責呈現遊戲中的動畫，例如：棋盤的放大、縮小、平移與棋子移動的過程（稱作飛子）、Lines 元件提供二個功能，其一是繪製提示線，其二是繪製輔助線，下面會進一步說明。

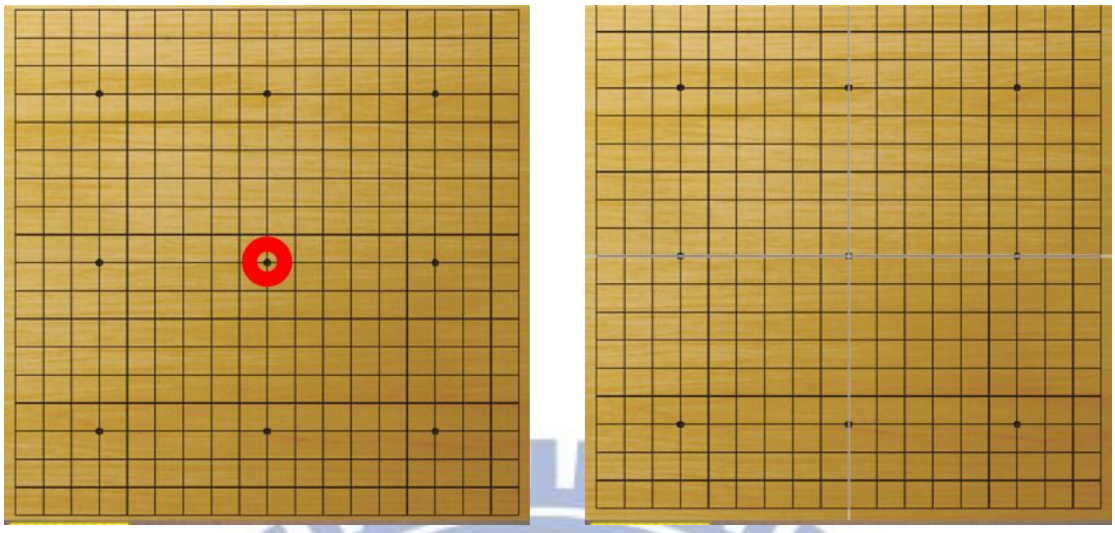


圖十四. GUI module 架構圖

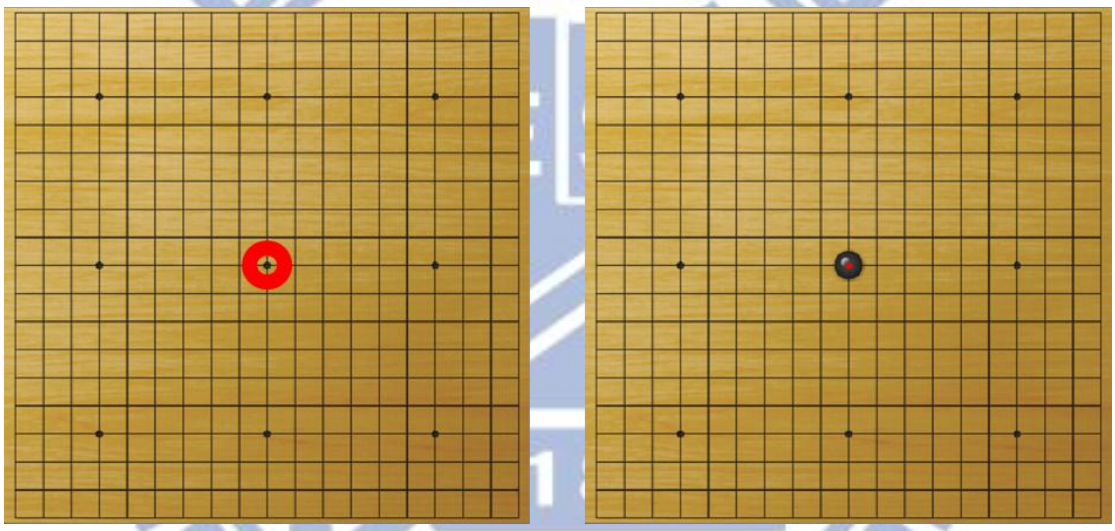


圖十五. 手勢圖

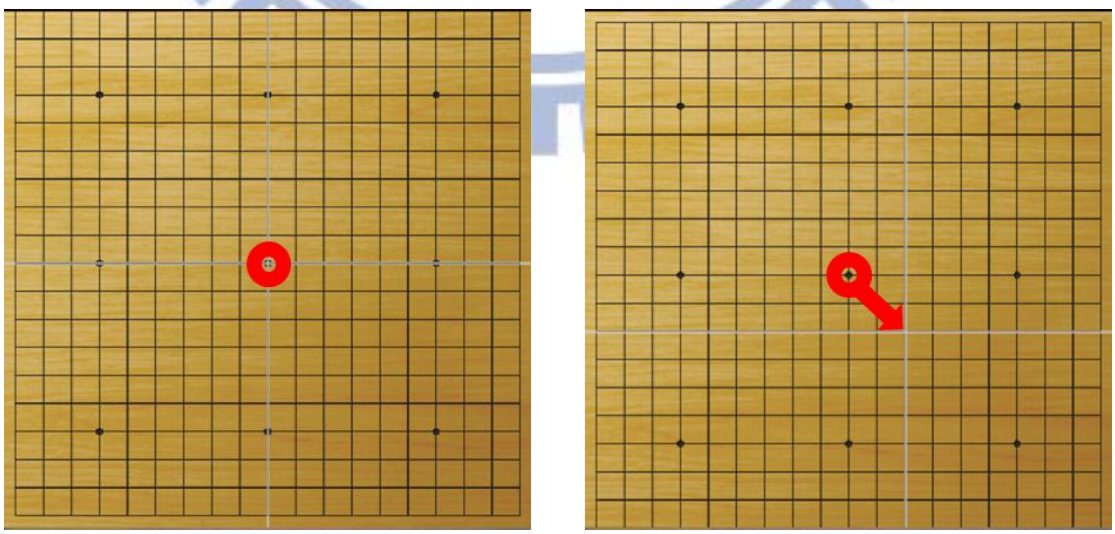
Touch 元件分析的手勢分成 Tap, Drag, Pinch in, Pinch out，如圖十五所示。以下手勢介紹皆以六子棋為例，Tap 手勢會根據按住的時間判斷成不同的行為，按住一小段時間會出現輔助線（圖十六），之後放開或按下馬上放開皆會判斷成下子的動作（圖十七），Drag 手勢當出現輔助線時作拖曳的動作時會移動輔助線（圖十八），棋盤放大時作拖曳的動作時會移動棋盤（圖十九），Pinch out 手勢會放大棋盤（圖二十），Pinch in 手勢會縮小棋盤（圖二十一）。



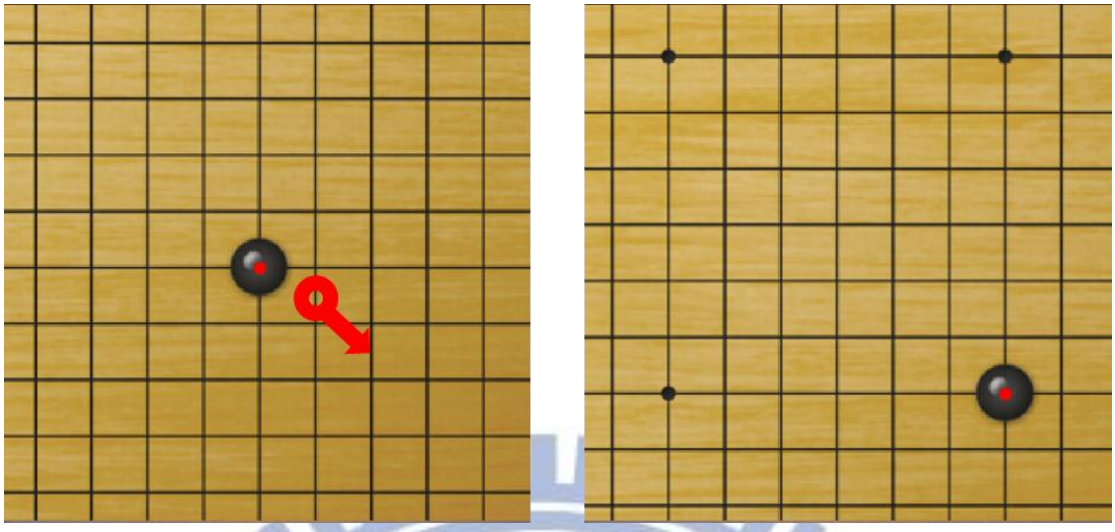
圖十六. Tap 手勢（按住一段時間）



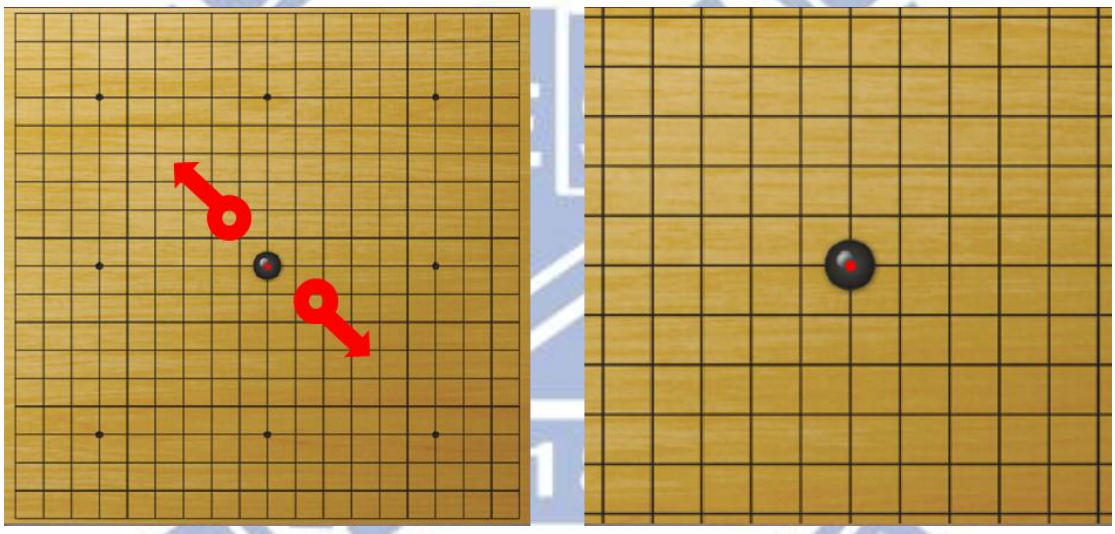
圖十七. Tap 手勢



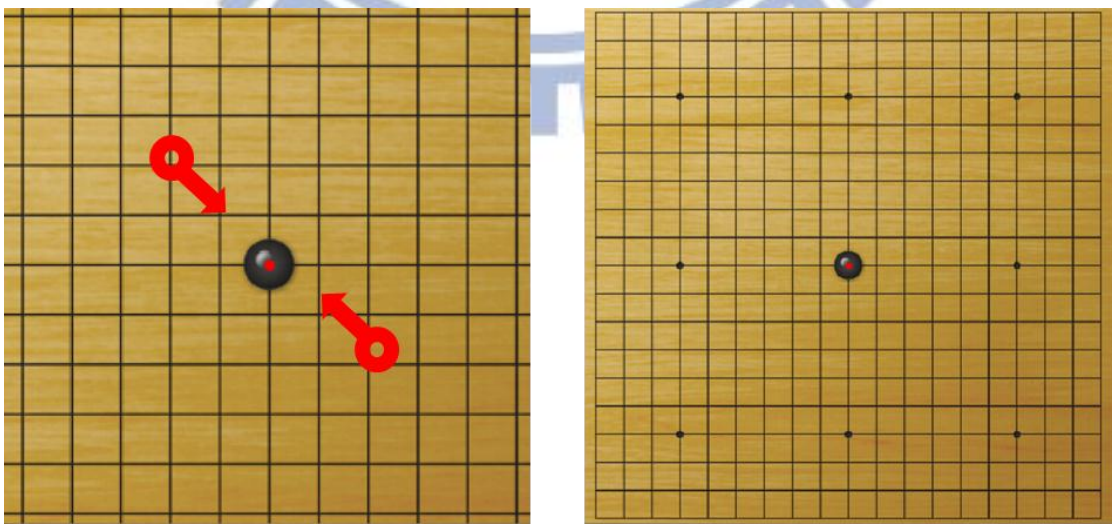
圖十八. Drag 手勢（拖曳輔助線）



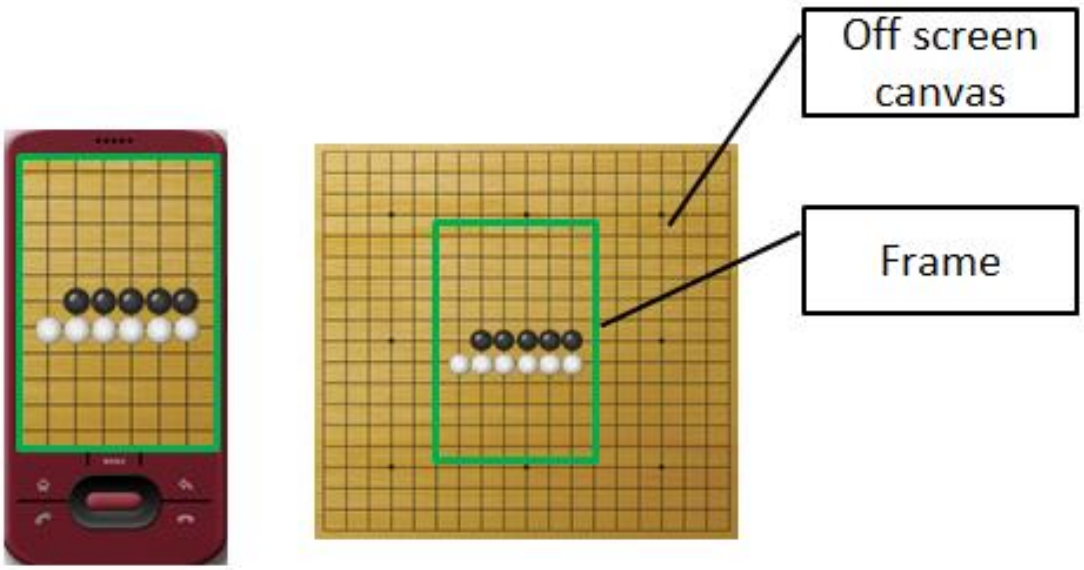
圖十九. Drag 手勢（拖曳棋盤）



圖二十. Pinch out 手勢

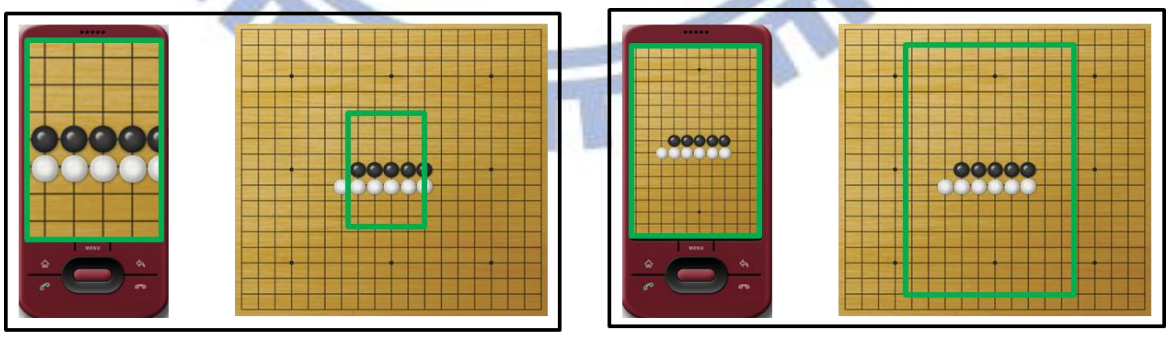


圖二十一. Pinch in 手勢



圖二十二. Double buffering（六子棋為例）

View 在呈現遊戲畫面時為了符合所有 Android 裝置的螢幕尺寸與繪製動畫時不會造成閃爍的現象，在實作上結合 Double buffering，除了顯示在螢幕上的畫布（Canvas）之外，我們還需要另一個非顯示在螢幕上的畫布（Off screen canvas）與一個框架（Frame），Frame 的長寬比與裝置的長寬比相同，如圖二十二所示，開發者將遊戲畫面畫在 Off screen canvas 上，再投影 Frame 內的畫面至裝置的 Canvas 上，如此，View 即可做到符合所有裝置螢幕的尺寸。



（a）　　（b）

圖二十三.（a）放大棋盤（b）縮小棋盤

表四. 優化前畫面更新率

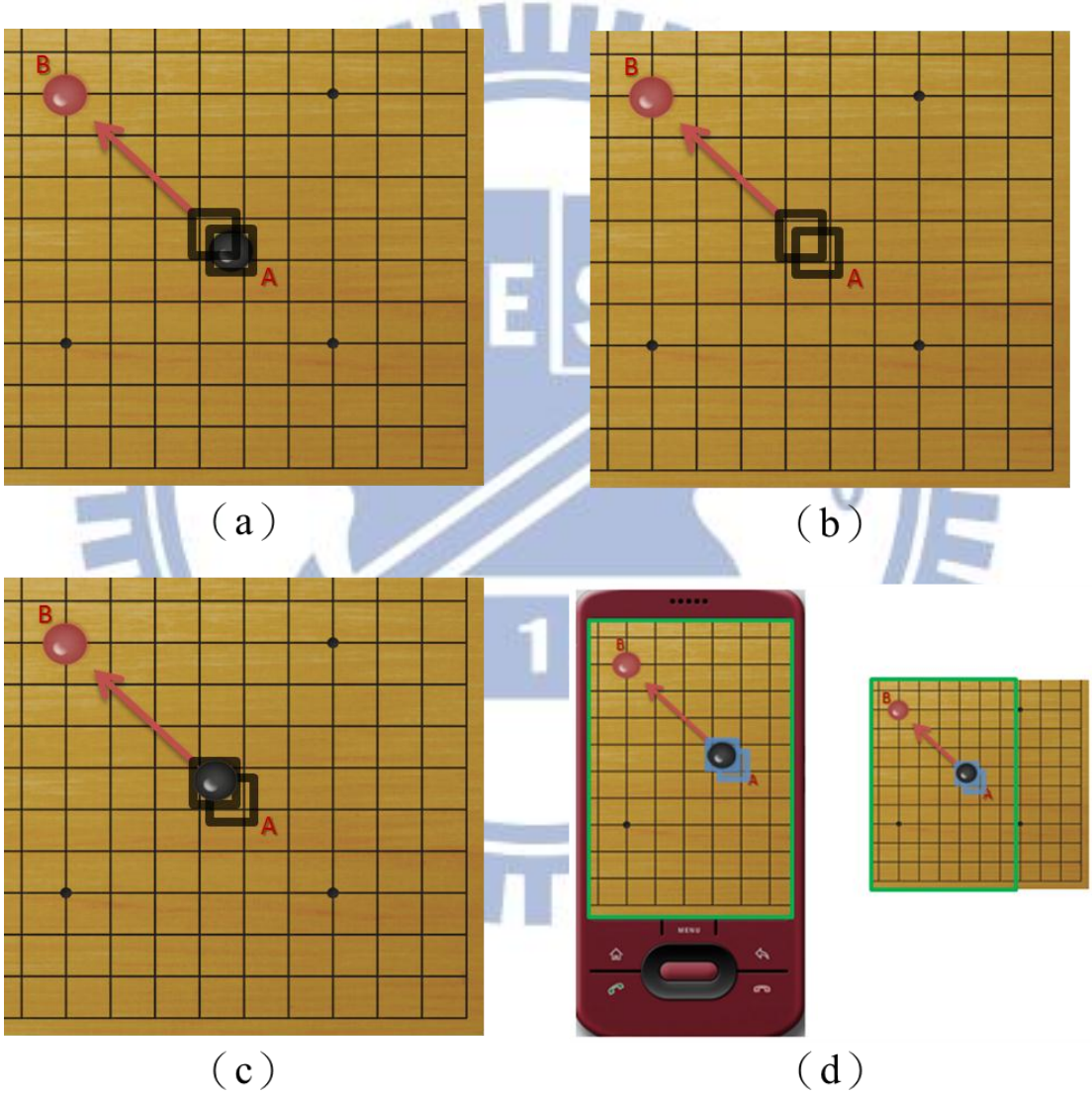
Android 裝置	Android 平台	每秒畫面更新次數	
		優化前	
		硬體加速	
		有	無
ASUS Transformer TF101	4.0	20	16
HTC Flyer	3.2.1	15	12

在上一段已說明 View 中結合了 Double buffering 實作，此方法讓 Animation 呈現放大、縮小、平移簡單許多，需要放大棋盤時只需縮小 Frame，相對需要縮小棋盤時只需放大 Frame，如圖二十三所示，而平移時只需移動 Frame，最後再重新投影 Frame 內的畫面至裝置的 Canvas 上即可。

人類的視覺可接受動畫的每秒影格數需達 24FPS 以上[20]，因此，飛子動畫在移動過程畫面需能每秒更新至少 24 次，在 Android 平台 3.0 以上對於動畫的支援較佳，是因為 Android 提供了 Property animation，它可以幫助開發者自動計算物件的某個屬性的值，隨著時間線性或非線性遞增或遞減，此外 Android 平台 3.0 以上有支援硬體加速，但 3.0 以下二者皆無。因此，我們在有硬體加速與無硬體加速的情況下都做了實驗，測試裝置實際每秒可更新畫面多少次，如表三所示。從實驗結果可看出，無論是否有硬體加速，每秒更新畫面的速度皆無法達到人眼需求，為了符合大多數平台可呈現平順的動畫，我們優化繪製動畫的方式，在不開啟硬體加速的情況下，動畫也能流暢的顯示。

優化繪製動畫的方式，如圖二十四，(a) 假設黑子要從 A 位置移動到 B 位置，我們利用一個執行緒計算每一次棋子移動的位置並記錄，每一次更新位置後即要求 View 更新畫面，在 View 中繪製畫面時，(b) 首先會在

Off screen canvas 畫上次棋子的位置的背景,(c)再畫這次棋子的位置,(d)更新至裝置的Canvas時只更新Frame內二次棋子的位置即圖中藍色區域，不用整個 Frame 的內容都更新。經過優化後，我們重新做了一次與上一段所提相同的實驗，發現畫面的更新速度大幅提升並且不需硬體加速即可達到人眼的需求，如表五。



圖二十四. Off screen canvas 繪製流程

表五. 優化前後畫面更新率比較

Android 裝置	Android 平台	每秒畫面更新次數		
		優化前		優化後
		硬體加速		
		有	無	無
ASUS Transformer TF101	4.0	20	16	36
HTC Flyer	3.2.1	15	12	33

3.4 AI Interface 設計

棋類的 AI 程式一般需要複雜的演算法去計算出結果，計算的過程可能相當耗時，因此，不可使用負責 UI 介面的執行緒（UI thread）去執行 AI 的計算，否則會造成應用程式畫面停住，嚴重 Android 系統會要求關閉應用程式，Android 稱這種情況為 ANR（Application not respond.），Android 默認的 ANR 時間最長為 5 秒。

如圖二十五所示，我們設計當 Control module 跟 AI interface 要求 AI 去想下一步棋時，會將目前的遊戲狀況當作參數傳給 AI interface，在 AI interface 中會新增一個執行緒去執行 AI 程式的計算，此時，由於 UI thread 並沒有被堵塞，因此，使用者手勢的操作在 AI 計算時依然可以被 UI module 中的 Touch 元件接受（說明見 3.3）。

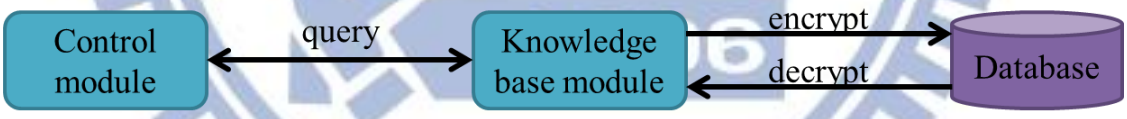
AI 程式方面，程式開發者可以自由選擇用 C/C++或 Java 開發，若開發者想透過 C/C++開發只需照著 2.6 節中所提 Android-NDK 的開發流程將 C/C++程式編譯成函式庫即可。

Connect6 應用程式的 AI 程式 Mobile6 如 2.3 所介紹，目前提供五種強度，由於 NCTU6 原本是在電腦終端上執行，電腦終端的資源（包含 CPU 時脈、記憶體容量）大於手機資源，因此，在移植 NCTU6 時，需注意運算時間與記憶體用量。

圖二十五. AI interface 架構圖

3.5 Knowledge Base Module 設計

棋類遊戲通常需要大量的開局與詰棋，因此，Knowledge base module 的設計結合 SQLite database，透過 Knowledge base module 能夠創建 database，並可對 database 作 create, insert, select, delete・・・等 query 行為，並且在 Knowledge base module 中提供加密與解密的功能，如圖二十六所示。



圖二十六. Knowledge base module 架構圖

Connect6 應用程式設計了詰棋闖關的機制，目前分為八個大關卡，每一個大關卡中有十二個小遊戲，這些詰棋題目是從 Little Golem 網站[16]上收集，從 6,539 題詰棋譜，請專家從中區別難度，挑選了 528 題作為 Connect6 應用程式的詰棋庫。

第四章、實作與結果

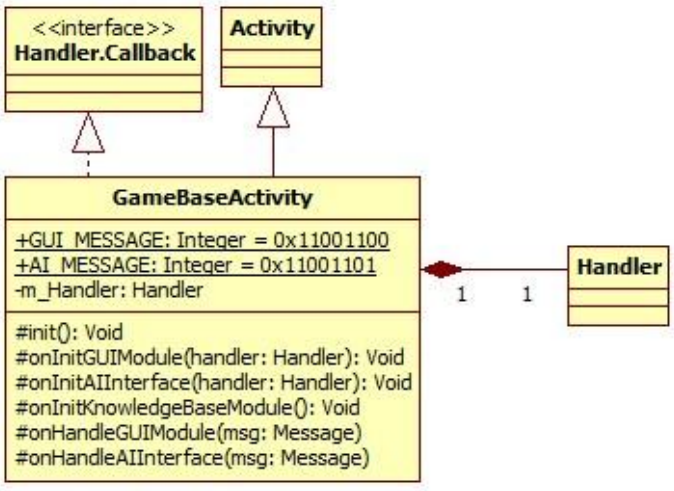
本章節中會說明實作，4.1 將會說明框架的實作，與如何使用框架快速開發棋類遊戲，4.2 將會提出 Mobile6 與 NCTU6 其效能與強度的比較，與我們的實作成果。

4.1 實作

在 4.1.1 到 4.1.4，會詳細說明開發者使用此框架需繼承的類別，需覆寫的函式與其相對應的功能。

4.1.1 Control Module

在 Control Module 中，主要控制各個模組初始化的流程，與開啟與各模組溝通的方法，開發者主要需繼承 GameBaseActivity 類別，開發者可視各自遊戲的需求，選擇覆寫相對應的函式，詳細的 UML 類別圖請見圖二十七。

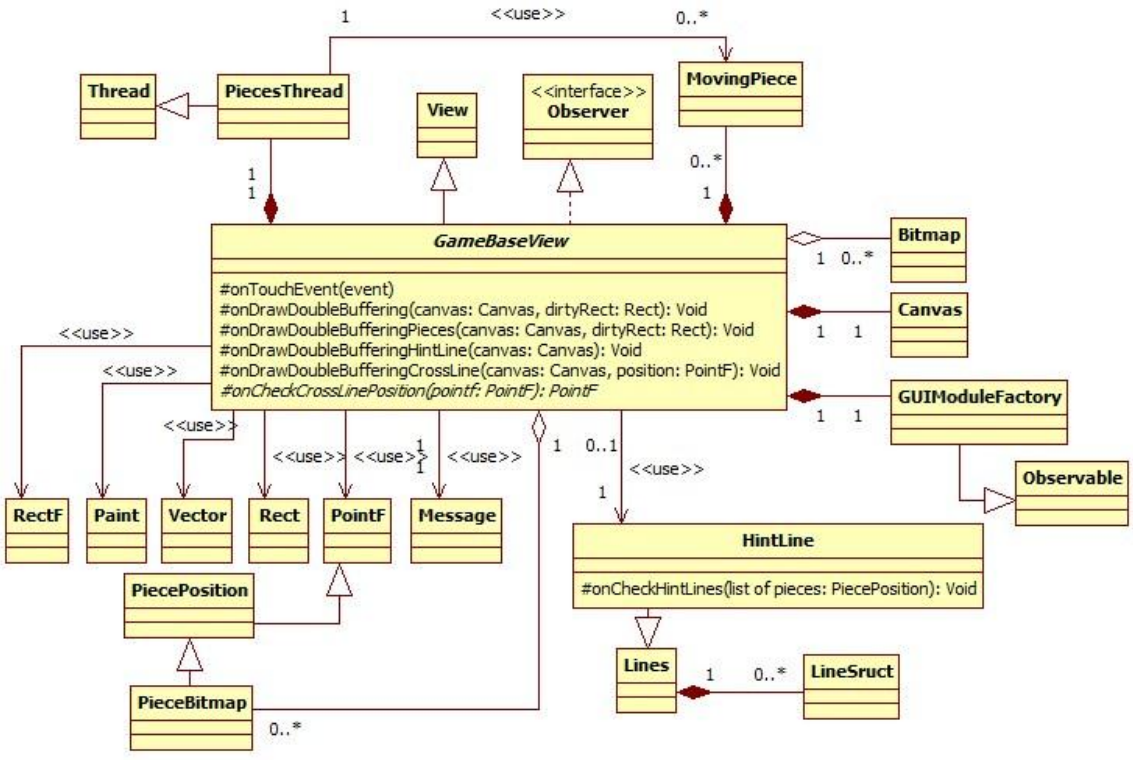


圖二十七. Control Module 類別圖

- `init`：初始化除了模組以外的物件或成員變數。
- `onInitGUIModule`：初始化 GUI Module。
- `onInitAIInterface`：初始化 AI Interface。
- `onInitKnowledgeBaseModule`：初始化 Knowledge Base Module。
- `onHandleGUIModule`：處理由 GUI Module 傳回的訊息。
- `onHandleAIInterface`：處理由 AI Interface 傳回的訊息。

4.1.2 GUI Module

在 GUI Module 中，提供開發者加速設計主要遊戲畫面，開發者需要繼承 GameBaseView，若遊戲需要提示線的功能，則開發者需要繼承 HintLine 類別，詳細的 UML 類別圖請見圖二十八。



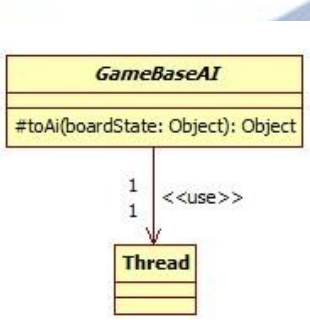
圖二十八. GUI Module 類別圖

- `onTouchEvent`：開發者可自行設計所需的手勢，預設的手勢可參考 3.3 所述。

- onDrawDoubleBuffering：繪製遊戲畫面的流程預設為棋盤、棋子、提示線、輔助線。覆寫此方法，開發者可自行設計繪製遊戲畫面的流程。
- onDrawDoubleBufferingPieces：覆寫此方法可供開發者在棋子上方增加圖層。
- onDrawDoubleBufferingHintLine：若開發者需要提示線的功能，需覆寫此方法，繪製遊戲的提示線。
- onDrawDoubleBufferingCrossLine：若開發者需要輔助線的功能，需覆寫此方法，繪製遊戲的輔助線。
- onCheckCrossLinePosition：開發者需設計根據玩家觸碰螢幕需顯示輔助線正確的位置。
- onCheckHintLines：需覆寫此函式，設計提示線的繪製方法。

4.1.3 AI Interface

在 AI Interface 中，開發人員需繼承 GameBaseAI 類別，詳細的 UML 類別圖請見圖二十九。

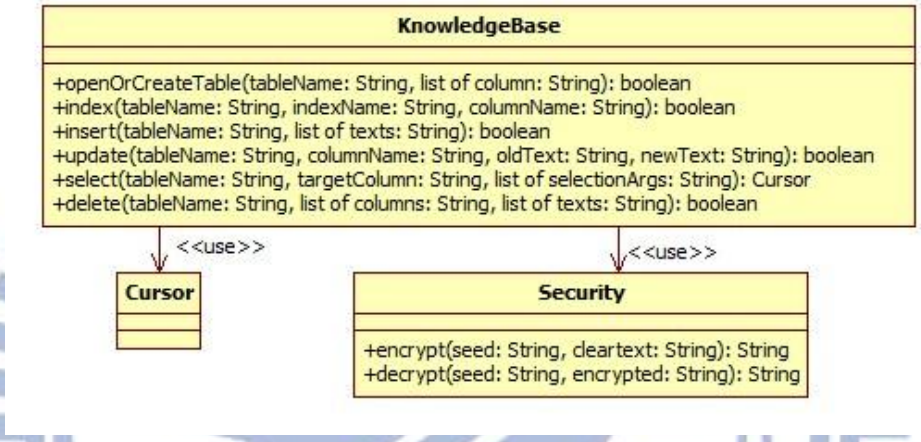


圖二十九. AI Interface 類別圖

- toAi：開發者只需覆寫此函式，設計遊戲的人工智慧，若開發者想用 C/C++設計人工智慧，則覆寫此函式呼叫 native 函式，做法可參考 2.6 所述。

4.1.4 Knowledge Base Module

在 Knowledge Base Module 中，我們提供 KnowledgeBase 類別，讓開發者使用，能夠快速建立開局庫或詰棋庫，並提供 Security 類別，讓開發者可對資料庫資料作加解密，詳細的 UML 類別圖請見圖三十。



圖三十. Knowledge Base Module 類別圖

- openOrCreateTable：建立表單，同時給予欄位名稱。
- index：對任意表單之欄位建立索引，同時需給予索引名稱。
- insert：對任意表單插入一筆資料，資料順序需依照欄位給予。
- update：對任意表單更新欄位的一筆資料。
- select：對任意表單取值，可給予條件取值。
- delete：對任意表單刪除一筆資料。
- encrypt：程式開發者需自行保有加密的種子，未來取值時解密用。
- decrypt：程式開發者解密時需有加密時所用的種子。

4.2 實作結果

在此章節會說明我們開發環境、適用平台與測試用機器，並呈現透過框架實作六子棋的成果。

- 開發環境：Eclipse IDE for Java Developers
- 適用平台：Android 2.1 or above.
- 測試用機器：請參考表六

表六. 實驗機器與規格表

Machine	ASUS Transformer TF101	HTC Flyer
CPU	NVIDIA® Tegra™ 2 1GHz	Qualcomm MSM8255, 1.5GHz
Memory	1 GB	1 GB
Android platform version	4.0.3	3.2.1

4.2.1 實驗

我們將 Mobile6 分成五個強度，從弱至強分別為 Rank1 至 Rank5，其中 Rank1 與 Rank2 並未作搜尋而是使用 Pattern Table[10]並做了二個實驗。第一個實驗，讓五個不同強度的 Mobile6 與 NCTU6 對戰，勝率如下，請參照表七。第二個實驗，取 127 題尚無必勝步之六子棋詰棋讓五個不同強度的 Mobile6 計算，目的是為了測試，在如表七所示勝率之下，Mobile6 的計算時間是否會超時，而造成玩家不好的觀感，結果如表八所示。結果顯示所有強度的 Moblie6 都在 1 分鐘內會作出回應。

表七. Mobile6 對 NCTU6 勝率表

Mobile6 Win rate	Mobile6 Rank1	Mobile6 Rank2	Mobile6 Rank3	Mobile6 Rank4	Mobile6 Rank5
NCTU6	0.56%	2.27%	8.24%	22.16%	28.98%

表八. Mobile6 運算時間表

	Mobile6 Rank1	Mobile6 Rank2	Mobile6 Rank3	Mobile6 Rank4	Mobile6 Rank5
Average Time	0	0	2.27	2.97	19.34
Worst Case Time	0	0	4.35	12.84	48.36

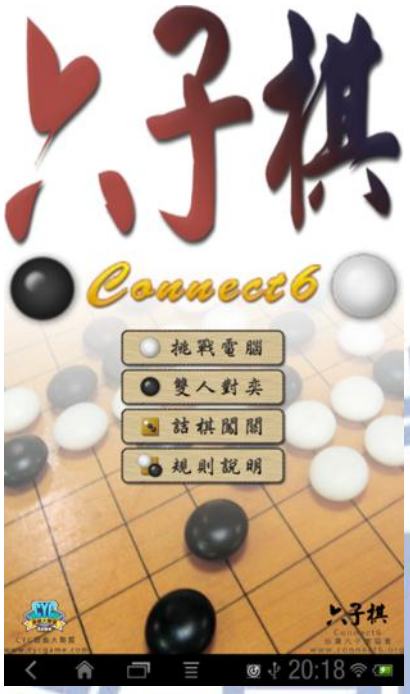
4.2.2 成果

我們成功透過框架實作六子棋在 Android 行動裝置，表九列出我六子棋應用程式的資訊，並與電腦版六子棋作比較。

表九. 電腦與行動裝置六子棋程式之比較表

	PC	Andorid Moblie			
		Without AI		With AI	
		Native	Dalvik	Native	Dalvik
Application size	21.1 MB	5.35 MB		14.88 MB	
Runtime memory size	581.116 MB	5.84 MB	3.2 MB	18.11MB	3.3MB

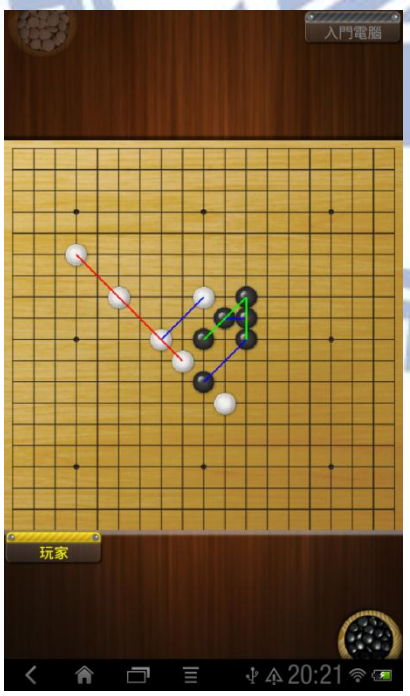
我們在 ASUS Transformer TF101 與 HTC Flyer 上都有安裝六子棋應用程式，在此列出 HTC Flyer 的程式畫面（ASUS Transformer TF101 畫面相同），如圖三十一。


六子棋
Connect6
挑戰電腦
雙人對奕
詰棋闖關
規則說明
六子棋
20:18

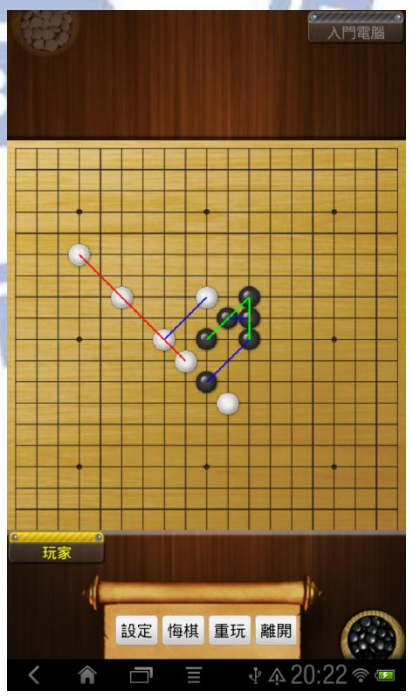
（a）


玩家一名稱
玩家一
黑色
白色
電腦人
入門
初級
中級
高級
畫線功能
開啟
關閉
音樂功能
開啟
關閉
開始
取消
玩家二名稱
玩家二

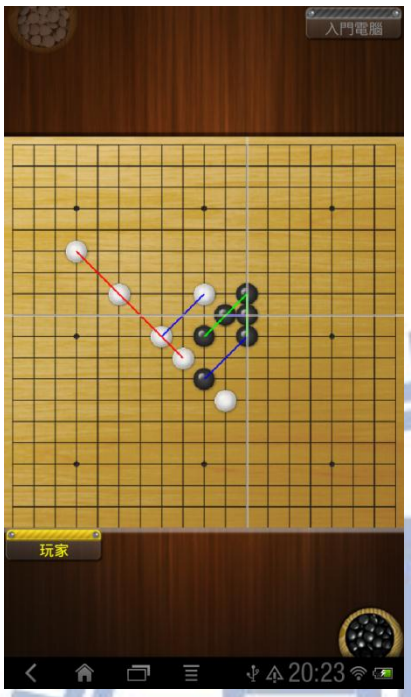
（b）


入門電腦
玩家
20:21

（c）


入門電腦
玩家
設定
悔棋
重玩
離開
20:22

（d）


入門電腦
玩家
20:23

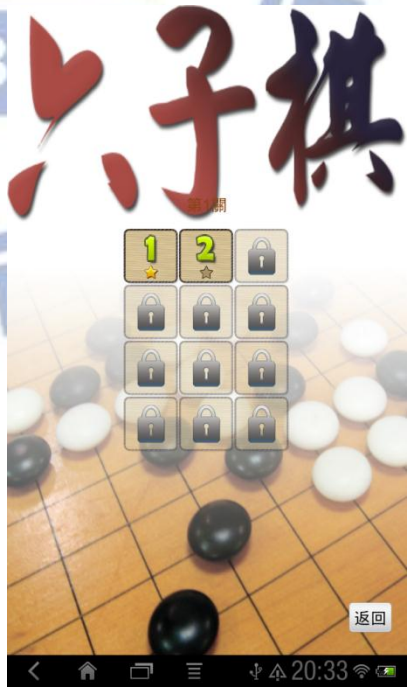
（e）


入門電腦
敗
勝
玩家
設定
儲存
新局
離開
20:26

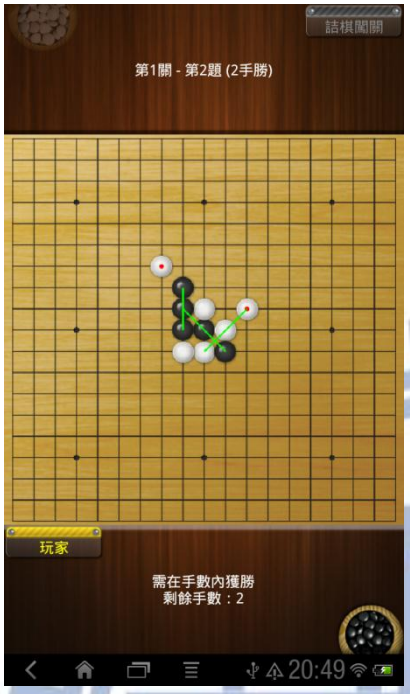
（f）


第一關
第二關
第三關
第四關
第五關
第六關
第七關
第八關
返回
20:27

（g）

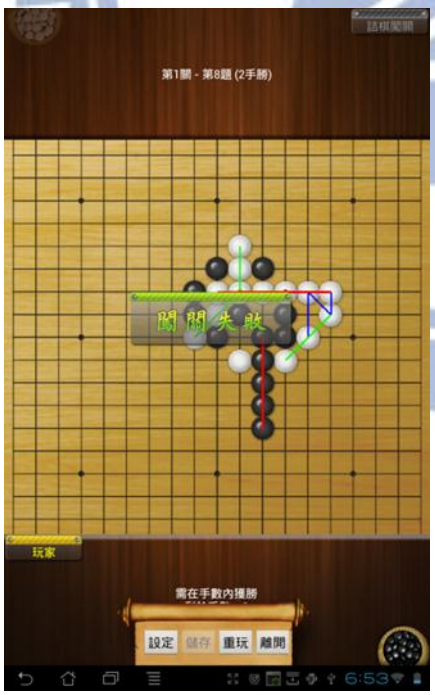

返回
20:33

（h）

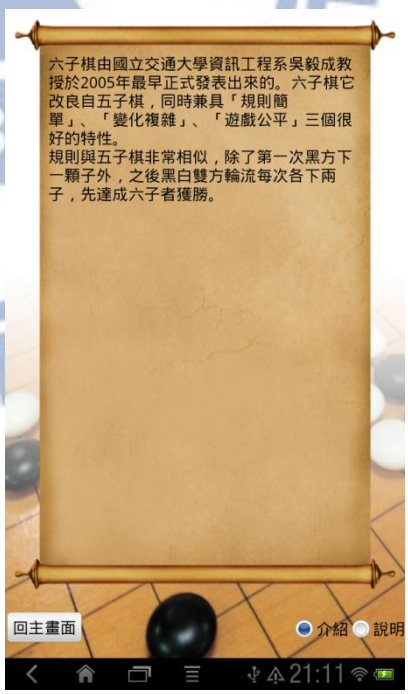


（i）

（j）



（k）



（l）

圖三十一. 六子棋程式畫面

在圖三十一中，(a) 是程式主畫面，按下挑戰電腦或雙人對弈皆會彈出 (b) 上方為單人模式設定視窗，下方為雙人模式設定視窗，讓玩家輸入名稱與選擇 AI 等級、畫線功能、音樂功能，按下確定會進入 (c) 遊戲畫面，在此畫面按下 Android 裝置的 Menu 鍵會跳出 (d) 選單，上面有設定、悔棋、重玩、離開等功能鍵。在棋盤上按住不放會出現 (e) 白色的十字輔助線，玩家贏或輸會出現 (f) 勝敗畫面。在 (a) 程式主畫面中按詰棋闖關會進入 (g) 詰棋關卡選單，按下關卡後，選單會換成 (h) 詰棋遊戲選單，再按下數字按鈕後會進入 (i) 詰棋闖關遊戲畫面，上方會提示玩家目前在第幾關第幾題與幾手內需獲勝，下方則會提示玩家還剩下幾手可下，玩家在手數內獲勝會出現 (j) 闖關成功，而超過手數未取得勝利或電腦 AI 取得勝利皆會出現 (k) 闖關失敗。在 (a) 程式主畫面中按下規則說明，會進入 (l) 規則說明畫面，在這裡會介紹六子棋與說明遊戲操作方法，按回主頁面按鈕可離開。

第五章、結論與未來展望

本篇論文中，我們建立了一個以 Android 為實作環境的遊戲框架，主要的目的是簡化遊戲設計的複雜度來縮短遊戲開發人員的開發時間，且開發人員能夠專注在遊戲內容上，透過此框架也能輕易地開發其他棋類遊戲，並且提供其他遊戲平台作為遊戲設計的參考。並且我們用此框架開發出六子棋遊戲，擁有以下功能：

- 友善的使用者介面，包括飛子、棋盤放大、縮小、平移、輔助線、提示線。
- 支援以 C++開發的六子棋 AI 程式（Mobile6），強度分成五種強度，從弱至強依序為 Rank1、Rank2、Rank3、Rank4、Rank5，分別與 NCTU6 對戰，得勝率分別為 0.56%、2.27%、8.24%、22.16%、28.98%，並且時間能在一分鐘內給予回應下一步棋的走法。
- 能夠存取大量的詰棋庫與開局庫並且能夠對資料庫加密與解密。
- 優化動畫繪製方式讓六子棋可平順的在 Android 平台 2.1 以上的裝置上執行。

未來系統框架中可加入網路模組，提供線上排名、遊戲中購買與網路對戰模式等機制，結合 Facebook[4]、Android C2DM 服務[5]與設計遊戲伺服端，達到通知配對與認證的功能。

參考文獻

[1] Bruski, Przemyslaw, The Java (not really) Faster than C++ Benchmark., available at http://bruscy.republika.pl/pages/przemek/java_not_really_faster_than_cpp.html

[2] Business Quotient Journal, Google 應用商店 Google Play 下載量破 150 億, available at http://www.bqjournal.com/?p=8259

[3] Cherrystone Softare Labs, Algorithmic Performance Comparison Between C,C++,Java and C# Programming Languages, available at http://www.cherrystonesoftware.com/doc/AlgorithmicPerformance.pdf

[4] Facebook, Facebook Homepage, available at http://www.facebook.com/

[5] Google Inc, Android Cloud to Device Messaging Framework(C2DM), available at http://code.google.com/intl/zh-TW/android/c2dm/

[6] Google Inc, Android developer guides, available at http://developer.android.com/index.html

[7] Google Inc, Android native development kit(NDK), available at http://developer.android.com/tools/sdk/ndk/overview.html

[8] Google Inc, Google Play, available at https://play.google.com/store

[9] Hipp, D. Richard., SQLite Homepage, available at http://www.sqlite.org/

[10] Hu, Chia-Yun., The Study and Development of Connect6 Game for iOS Devices, 2012

[11] IDC. Android and iOS Surge to New Smartphone OS Record in Second Quarter, available at http://www.idc.com/getdoc.jsp?containerId=prUS23638712

[12] IDC. Nearly 1 Billion Smart Connected Devices Shipped in 2011 with Shipments Expected to Double by 2016, available at http://www.idc.com/getdoc.jsp?containerId=prUS23398412

[13] International Computer Games Association(ICGA), ICGA Homepage, available at http://ticc.uvt.nl/icga/

[14] Jelovic, Dejan., Why Java will always be slower than C++, available at http://www.jelovic.com/articles/why_java_is_slow.htm

[15] Lin, H.-H., Sun, D.-J., Wu, I.-C. and Yen, S.-J., The 2010 TAAI Computer-Game Tournaments, ICGA Journal, Vol. 34(1), March 2011.

[16] Little Golem. Little Golem Homepage, available at http://www.littlegolem.net/jsp/

[17] Statista. Number of available applications in the Google Play Store from December 2009 to May 2012, available at http://www.statista.com/statistics/74368/number-of-available-applications-in-the-google-play-store/

[18] Taiwan Connect6 Association, Connect6 Homepage, available at http://www.connect6.org/

[19] Wikipedia, ACID., available at http://zh.wikipedia.org/wiki/ACID

[20] Wikipedia. Frame rate., available at http://en.wikipedia.org/wiki/Frame_rate

[21] Wu, I.-C. and Lin, P.-H., Relevance-Zone-Oriented Proof Search for Connect6, the IEEE Transactions on Computational Intelligence and AI in Games (SCI), Vol. 2(3), pp. 191-207, September 2010.

[22] Wu, I.-C., and Huang, D.-Y., A New Family of k-in-a-row Games. The 11th Advances in Computer Games Conference (ACG'11), pp. 180-194, Taipei, Taiwan, 2005.

[23] Wu, I.-C., and Lin, P.-H., NCTU6-Lite Wins Connect6 Tournament, ICGA Journal, Vol. 31(4), 2008.

[24] Wu, I.-C., and Yen, S.-J., NCTU6 Wins Connect6 Tournament, ICGA Journal, Vol. 29(3), pp. 157-158, September 2006.

[25] Wu, I.-C., Huang, D.-Y., and Chang, H.-C., Connect6. ICGA Journal, Vol. 28(4), pp. 234-242, 2006.

[26] Wu, I.-C., Lin, H.-H., Lin, P.-H., Sun, D.-J., Chan, Y.-C., and Chen, B.-T, Job-Level Proof Number Search for Connect6. In the International Conference on Computers and Games (CG 2010), Kanazawa, Japan, 2010.