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

在 iOS 行動裝置上之六子棋遊戲設計與研究

The Study and Development of Connect6 Game for iOS Devices

研 究 生：胡嘉芸

指導教授：吳毅成　教授

中 華 民 國　101　年 9 月

在 iOS 行動裝置上之六子棋遊戲設計與研究
The Study and Development of Connect6 Game for iOS Devices

研 究 生：胡嘉芸　　Student：Chia-Yun Hu

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）產業研發碩士專班

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

在 iOS 行動裝置上之六子棋遊戲設計與研究

研究生：胡嘉芸　　指導教授：吳毅成

國立交通大學 資訊學院資訊科技（IT）產業研發碩士專班

摘要

iOS 是 Apple 公司發展的行動裝置作業系統，目前以 iOS 為主的行動裝置十分普遍，其應用程式下載量亦相當龐大。

六子棋是一個新的棋類遊戲，近年來已成為國際性電腦遊戲競賽的項目之一。本實驗室已發表許多關於六子棋的研究，以及開發六子棋程式—NCTU6，在國際比賽中獲得許多佳績。為了使六子棋有多樣性的發展，我們決定在 iOS 行動裝置上開發六子棋遊戲程式。

本篇論文描述一個在 iOS 行動裝置上開發六子棋遊戲的設計與實作。此程式除了完成基本的六子棋雙人對弈功能，還支援由本實驗室發展的人工智慧程式讓使用者可以與程式對弈以增長棋力。同時，本程式提供詰棋闖關遊戲，讓使用者藉由解詰棋題目的遊戲過程達到自我訓練的目的。另外，本論文的設計經驗可以提供之後不同棋類遊戲的 iOS 應用程式開發作為參考。

The Study and Development of Connect6 Game for iOS Devices

Student: Chia-Yun Hu Advisor: I-Chen, Wu

Industrial Technology R & D Master Program of

Computer Science College

National Chiao Tung University

Abstract

iOS is a mobile operating system developed and distributed by Apple Inc.. Mobile devices based on iOS are popular, An evidence is that the times of application downloading is the highest among all mobile devices.

Connect6 is a new board game introduced by Professor I-Chen Wu in 2005. There has several researches of Connect6 were done before in our laboratory. In this thesis, we develop a Connect6 game on iOS mobile devices.

The program includes the basic game with two-players and the one-player game where users play against the Connect6 AI program supported by our laboratory. Besides, we also support puzzle games for users' self-training. With our design and experience, other board games can be developed for iOS devices easily in the future.

誌謝

本篇論文得以完成，首先要感謝我的指導教授–吳毅成老師。老師對研究總是保有熱忱，常在學校待到很晚，開會時都會很認真討論研究的目標，寫論文時也提供許多實用的建議，感恩老師兩年來的教導。還有十分感謝口試委員朱正忠教授、顏士淨教授與汪益賢博士的指導與講評。

再來，要感謝對我來說很重要的家人，父母使我在求學階段沒有經濟壓力，也包容我較少時間陪伴與聊天。弟弟們雖然比較不擅長表達關懷，但偶爾的關懷也很溫暖。

感謝即將進入的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，這些都使我感覺尊博是個溫馨和善的公司。

感謝群想網路科技股份有限公司提供遊戲流程編排與美工設計，感謝林修全學長對於程式實作的建議，感謝美工林信昌先生設計圖片。

CGI 實驗室裡的大家也是我強力的後盾，感謝實驗室助理柯月卿小姐處理實驗室各項事務，感謝林宏軒學長協助縮減棋型表(見表 6)的實作；感謝林秉宏學長、康皓華、張傑閔幫忙調整六子棋 AI 程式(見表 2)；感謝同在 mobile 組的鄭吉閎、魏經軒互相交流討論；感謝孫德中學長、曾汶傑學長提供的技術支援和遊戲的意見；感謝魏廷翰協助校正修改投影片；感謝張元耀準備口試餐點；感謝吳東穎維護實驗室的環境及協助測試程式；感謝所有學長姐、同學及學弟讓實驗室有和樂的氣氛；感謝曾給我鼓勵的所有朋友。

最後，僅以本篇論文獻給家人、老師、同學、朋友，感謝給予我心靈上支持的所有人、事、物。

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

在本章節中，將會介紹現今行動裝置趨勢及研究動機，另外會說明本篇論文的大綱。第 1.1 節介紹行動裝置發展趨勢，第 1.2 節討論 iOS 應用程式的普及程度，第 1.3 節提出研究的動機和目的，第 1.4 節介紹本論文的架構。

1.1 行動裝置發展趨勢

近幾年來，智慧型手機和平板電腦等智慧型行動裝置大量出現在市面上，使用智慧型行動裝置成為一種趨勢。

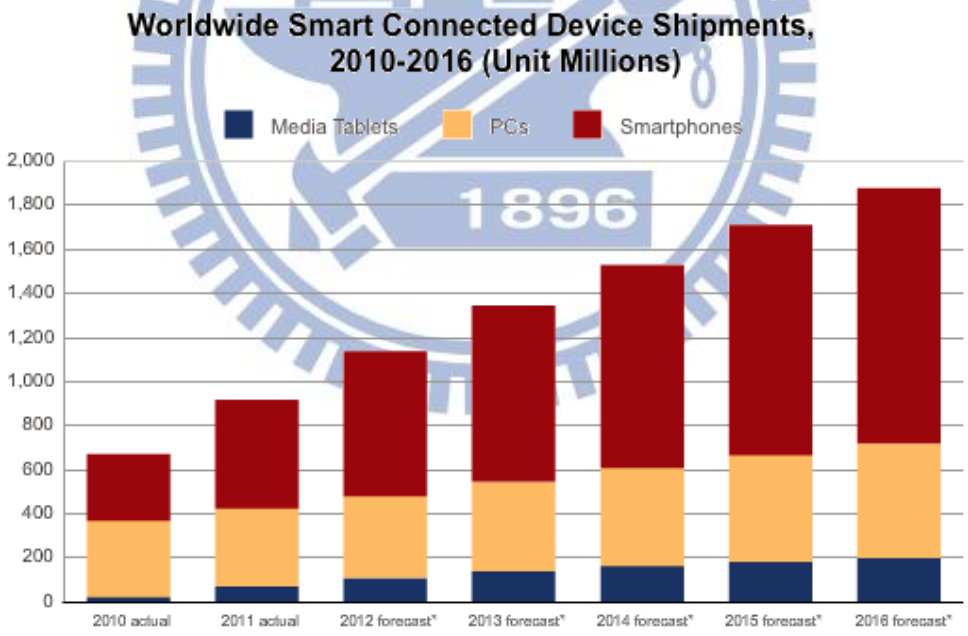


圖 1. 世界智慧型裝置數量數據(節錄自[4])

根據國際數據資訊公司(IDC)統計分析[4]，圖 1 顯示 2010 年到 2011 年平板電腦、智慧型手機及個人電腦的數量統計，而 2012 年到 2016 年的

數量是來自專家預測。如圖所示，在 2011 年間智慧型手機銷售量已超越個人電腦，而且根據專家預測，未來的智慧型手機銷售量成長迅速，銷售量甚至可達個人電腦銷售量的兩倍以上，另外，平板電腦銷售量也逐年增加。可見智慧型行動裝置已經漸漸成為市場的主流。

1.2 iOS 應用程式下載數據

App Store 是 Apple 公司創建的 iOS 應用程式發佈平台，於 2008 年啟用。依據不同應用程式的發佈情況，使用者可從 App Store 付費或免費下載程式，根據 Mobile Statistics 網站統計資訊[10]，App Store 中應用程式的下載量很龐大，如下圖所示。

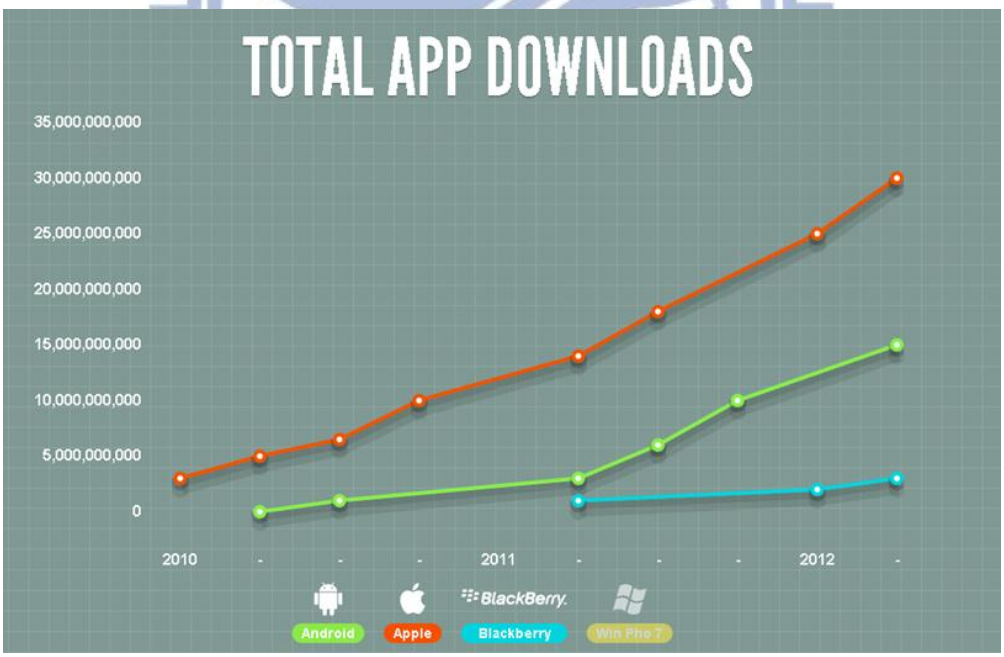


圖 2. 行動裝置作業系統的應用程式總下載量比較圖(節錄自[10])

圖 2 顯示從 2010 到 2012 年第二季幾種行動裝置作業系統的應用程式

總下載量，可以看出 Apple 公司的 iOS 應用程式下載量位居第一，而且於 2012 年已突破三百億的下載量，其下載次數仍持續增加中。根據以上數據可以得知目前在 iOS 行動裝置上發展應用程式十分普及。

1.3 研究動機與目的

六子棋從 2005 年發表至今有許多發展，包括成立台灣六子棋協會，舉辦六子棋公開賽，成為第十一屆奧林匹亞電腦賽局競賽的項目等。另外，如前兩節所述，近年來行動裝置的便利與開發，iOS 應用程式相當普遍。

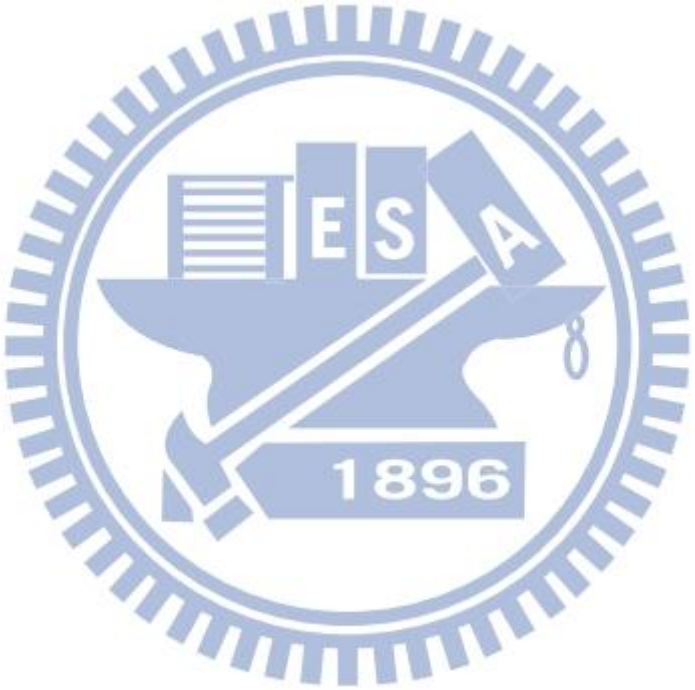
因以上所述，本論文設計及實作六子棋在 iOS 行動裝置上，並支援實驗室發展的人工智慧程式，使得六子棋有多樣性的發展。此外，藉由本次設計經驗，可以更了解 iOS 應用程式開發，往後更容易發展不同棋類或平台的一般化框架。

在本篇論文中，iOS 行動裝置上的六子棋遊戲設計與實作必須完成以下需求：

- 友善的使用者介面，包括可縮放的棋盤、可依揮動手勢拉出或隱藏的功能選項表、下子輔助線等。
- 移植以 C++開發的人工智慧程式，並修改程式以符合行動裝置需求。
- 建置資料庫以儲存大量詰棋盤面。

1.4 論文組織

本篇論文在第一章介紹了行動裝置及 iOS 應用程式的發展，亦提出了動機與目的。第二章研究背景將闡述本篇論文中使用的相關技術背景知識，例如 iOS、Objective-C、SQLite 等。第三章則是系統設計，講解系統架構及流程。第四章是實作方法，討論實作時遇到的困難與解決方法，並講解程式的使用者介面。第五章展示六子棋遊戲畫面，第六章則是結論以及未來展望。



第二章、研究背景

在本章中，首先在 2.1 節介紹六子棋的由來及規則，以及在 2.2 節介紹本實驗室發展的六子棋人工智慧程式—NCTU6。在 2.3 節則描述 iOS 基本架構及提供的技術，2.4 節介紹在 iOS 開發用的程式語言 Objective-C，最後於 2.5 節介紹在程式中用於存取資料庫的 SQLite。

2.1 六子棋

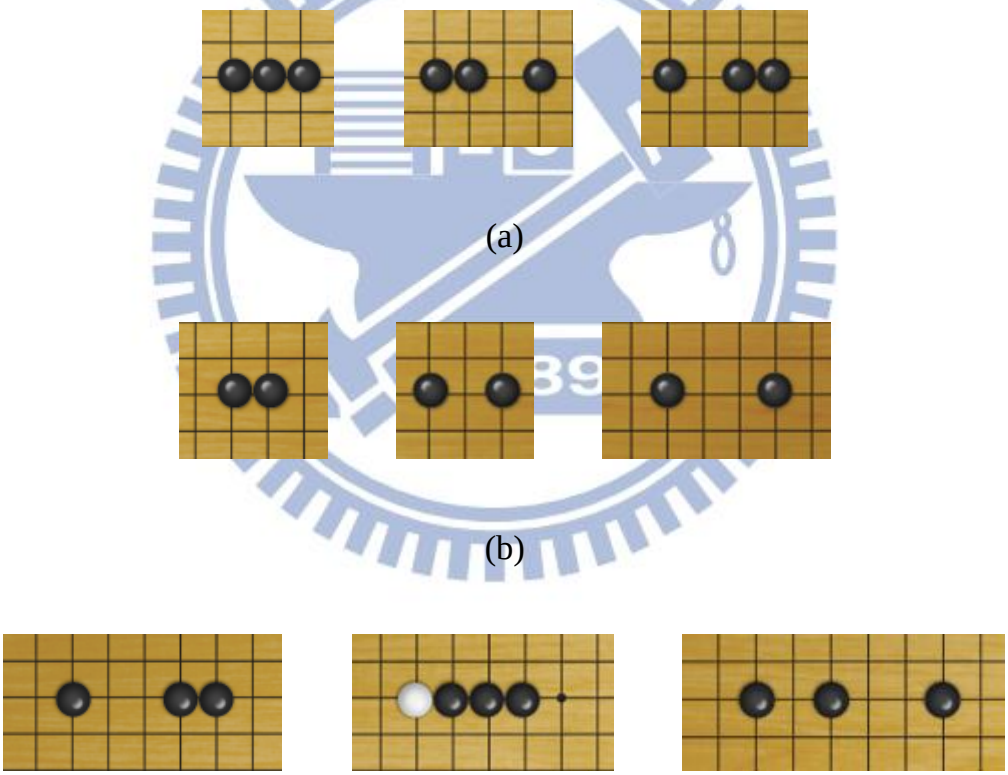
六子棋是交通大學吳毅成教授於 2005 年 9 月發表於第十一屆電腦賽局發展學術研討會(ACG 11)的一種棋類遊戲[14][16][17]，六子棋的特色是規則簡單，玩法複雜，遊戲公平。六子棋於 2006 年成為奧林匹亞電腦賽局競賽項目之一，也陸續有六子棋公開賽舉辦，還有六子棋協會已成立。規則是黑的第一手下一子，之後黑白雙方輪流各下兩子，連成六子以上的玩家獲勝。如圖 3 所示，白方已連成六子，所以白方獲勝。

圖 3. 六子棋棋局

而在六子棋中，迫著(Threats)的定義[14][17]如下：若一方需要下一顆

子以避免對方連成六顆以上的子獲勝，則稱對方有單迫著；若一方要下兩顆子以避免對方獲勝，則稱對方有雙迫著；而若一方要下三顆子才能避免獲勝時，稱對方有三迫著，且對方必勝。

依據以上理論衍生出死三、活三、活二：若僅再下一顆子，可以產生單迫著，稱為「死三」；若再下一顆子，就可以產生雙迫著，稱為「活三」；若再下兩顆子，可以產生雙迫著，則稱為「活二」。下方圖 4 顯示了幾個黑方有活三、活二、死三的例子。



(a)

(b)

(c)

圖 4. (a)活三 (b)活二 (c)死三

2.2 NCTU6

NCTU6 是交大吳毅成教授實驗室團隊所研發的六子棋 AI 程式，中文稱為「交大六號」。為了增強棋力或加快運算速度，NCTU6 加入了一些本實驗室在六子棋中的研究，如：Threat-space Search[16]、Relevance-zone-oriented Proof Search[19]等。

NCTU6 曾在比賽中獲得不少獎項[7][18][20]，例如：國際奧林匹亞電腦賽局競賽六子棋組冠軍和人腦對電腦大賽中獲勝，詳細項目及成果如下表 1 所示。

年份	比賽項目	NCTU6 比賽結果
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

表 1. NCTU6 比賽成果表

2.3 iOS

iOS[6]是在 iPod touch、iPhone 和 iPad 上運行的作業系統，負責管理 mobile 的硬體及提供開發基礎應用程式所需的技術。以下子節將分別描述

iOS 的架構和本論文研究開發使用到 iOS 提供的相關技術。

2.3.1 iOS 架構

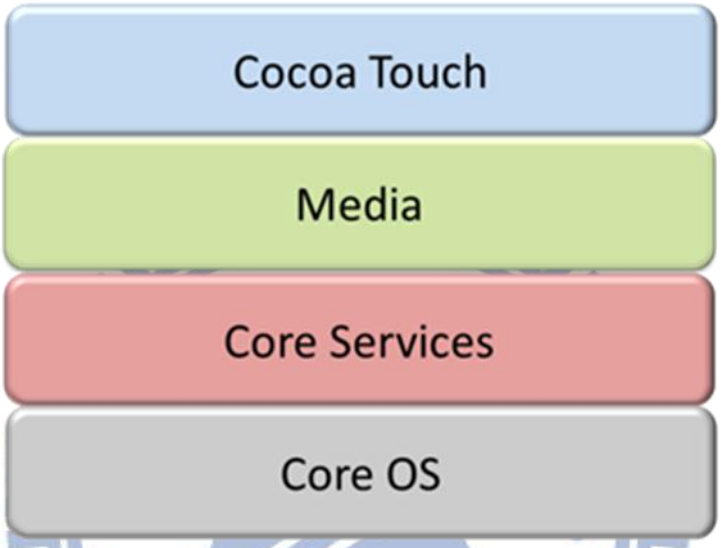


圖 5. iOS 架構圖

從圖 5 中可看到 iOS 架構主要分成四層，分別為 Cocoa Touch、Media、Core Services 及 Core OS，以下將簡略描述各層的技術及提供的服務。

- Cocoa Touch Layer

  此層包含開發 iOS 應用程式所需的關鍵框架，例如：Event Kit UI Framework、UIKit Framework 和 Game Kit Framework 等。

- Media Layer

  此層提供圖形、音訊和視頻技術，透過結合這些技術可讓 mobile 使用者有更精彩的多媒體使用體驗。開發者可利用此層提供的框架更快速

創造圖形或動畫效果，例如：Core Graphics Framework、Core Text Framework、OpenAL Framework 及 OpenGL ES Framework 等。

- Core Services Layer

  此層提供所有應用程式可用的基礎系統服務，但通常不會被應用程式直接使用。例如：iCloud Storage。而可用的框架有 Core Foundation Framework、Core Location Framework 和 Core Media Foundation Framework 等。

- Core OS Layer

  此底層包含許多可達成其他技術的基礎框架，這些功能通常應用於其他框架，但幾乎不會直接被應用程式使用，例如：Accelerate Framework、Core Bluetooth 及 Security Framework 等。

2.3.2 使用的 iOS APIs

另外，iOS 提供了許多 APIs (Application Programming Interface) [5]供開發者使用，在本實作中常用的 APIs 如下所示。

- NSNotification

  使用 NSNotification 物件可以將資訊封裝並發送廣播給其他物件。NSNotification 物件中包括識別用的名稱、發送廣播的物件、其他相關的資訊。

- NSNotificationCenter

  此類別提供在程式中廣播的機制，物件可使用 NSNotificationCenter 物

件搭配 NSNotification 物件發送廣播訊息及註冊指定接收的訊息。

- NSString

  NSString 物件是在 iOS 中使用的不可變字串物件。

- NSThread

  使用 NSThread 物件可以實作及控制多線程。

- NSTimer

  顧名思義，使用 NSTimer 物件可以實作出計時器，定時也可以重複地去觸發功能。

- UIAlertView

  使用 UIAlertView 物件可以實作彈出的互動視窗。

- UIButton

  UIButton 類別提供幾種制式的按鈕實作方法，開發者可以設定按鈕在接收到特定的觸碰事件時送出對應行動的訊息。

- UIImageView

  UIImageView 物件是提供放置圖像的容器，因為繼承 UIView 類別，所以也可以完成簡易的動畫效果。

- UILabel

  UILabel 物件是用於顯示文字訊息，使用者只能閱讀 UILabel 上的字，不能更改文字內容。

- UIScrollView

  使用 UIScrollView 物件可以方便地實作縮放畫面內容及移動畫面可視區域。

- UISegmentedControl

此類別提供開發者實作水平的控制選項，像是由幾個 UIButton 物件所組成的控制選項。

- UISlider

  此類別提供開發者實作滑動式的控制選項，如果其中只有兩個選項，則看起來像一般電源開關。

- UITextField

  UITextField 物件可提供使用者編輯文字資訊的訊息框。

- UITouch

  藉由 UITouch 物件，開發者可以取得使用者觸碰螢幕事件的內容，例如：觸碰位置、點擊次數。

- UIViewController

  此類別提供基本的畫面物件管理，在此種物件內可以添加許多的畫面物件，例如：UIImageView、UIButton 等物件。

以上列出了許多實用的類別，使用這些類別可以讓實作使用者介面與訊息傳遞更加便利。

2.3.3 iOS 支援的手勢

在 iOS 行動裝置上，大部分使用者的輸入都是來自手指觸碰螢幕。iOS 支援許多手勢的輸入，下圖列出 iOS 行動裝置支援的手勢。

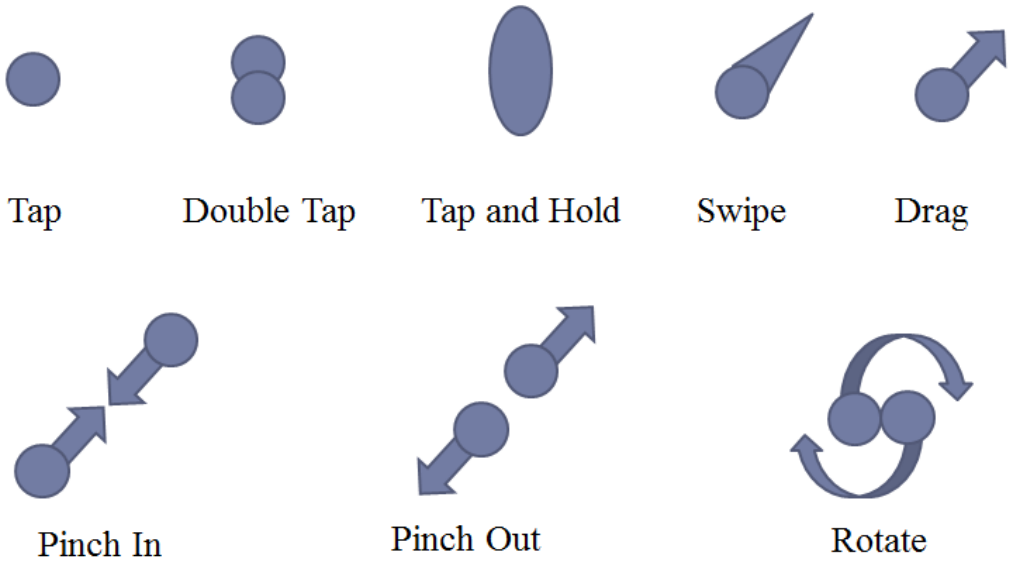


圖 6. iOS 支援手勢示意圖 [3]

圖 6 列出在 iOS 行動裝置上常見的手勢，例如 Tap 即點擊，常用於按下按鈕；Pinch In 和 Pinch Out 則常用於放大縮小畫面。在本論文中，應用手勢的部分將在 3.2.1 節敘述。

2.4 Objective-C

Objective-C[2]是以 C 語言為基礎加上物件導向特性而成的程式語言，其物件導向的語法源於 Smalltalk 語言 — 最早的物件導向語言之一[1]。目前 Objective-C 主要用於開發 iOS 應用程式，Objective-C 有以下特性：

- General-purpose

  在電腦軟體的領域中，通用程式語言意指設計成可編寫應用於廣泛的領域，並非為了特定領域的用途而生，例如 C++也是一個通用程式語

言，而 XML 則非此類語言。

- High-level

  高階語言是經過高度封裝的程式語言，相對於低階語言，高階語言較接近人類日常的語言，可讀性較高。

- Object-oriented

  物件導向是一種程式的開發方式，將物件作為程式中的基本單元，提高程式的重用性與擴充性。

- Thin runtime system

  Objective-C 使用 C 寫成的執行庫，容量很小，不像許多物件導向系統在執行時需要用到較大的虛擬機器。

- No garbage collection mechanism at first

  在較早的版本中，不支援垃圾回收機制。後來某些版本加入此功能，Apple 公司在 Mac OS X 10.5 中提供實作。

基於以上特性，使用 Objective-C 開發的程式容量通常不會比其原始碼及用到的函式庫大太多，而且可以利用現存的許多 C 語言的語法，是一個實用的語言。

2.5 SQLite

SQLite[12]是一個符合 ACID 特性的關聯式資料庫管理系統，在 iOS 程式開發中，客戶端的大型資料儲存通常是使用 SQLite 資料庫。ACID 特性如下所示：

- Atomicity(原子性)

  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，像是沒執行過。

- Consistency(一致性)

  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(獨立性)

  兩個事務的執行互不干擾，事務時間不會相互影響。

- Durability(持久性)

  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。

因為具有以上特性，SQLite 使我們更方便存取與管理資料庫，在本論文中，知識庫模組實作就是使用 SQLite。

第三章、系統設計

本章將介紹系統設計，在 3.1 中介紹系統整體架構，而在 3.2 到 3.4 各節則講述各模組工作內容與模式。

3.1 系統架構

我們把系統化分成四個部分討論，包括控制模組、GUI 模組、AI 程式之支援、知識庫模組，如下圖所示。

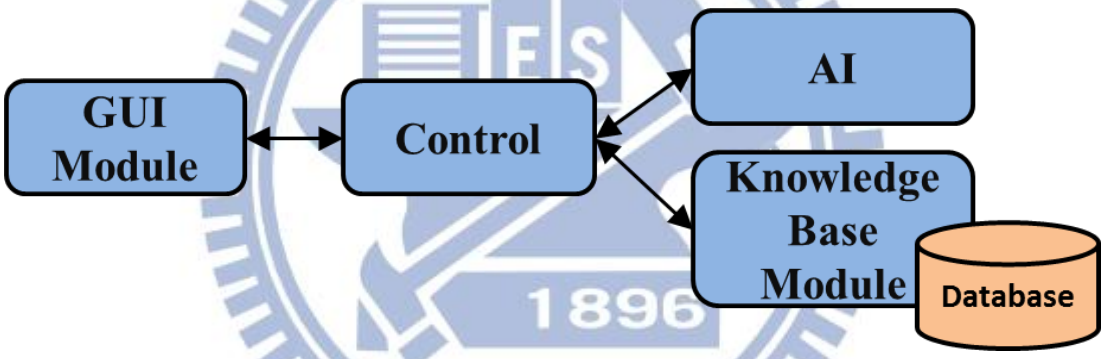


圖 7. 系統架構圖

圖 7 顯示本系統的架構，控制模組主要負責其它各模組間的訊息傳遞；GUI 模組管理使用者的互動介面與畫面配置；AI 模組則提供呼叫 AI 運算的介面，並存取 AI 運算結果；知識庫模組則負責存取資料庫中棋類相關的知識內容，例如：棋譜、使用者資訊。各模組負責的內容與溝通將詳述於 3.2、3.3、3.4 子節。

3.2 GUI 模組

GUI 模組負責控管與使用者互動的操作介面，GUI 模組包含 Touch、View Controller、Animation、Line 這幾個元件，如圖 8 所示。各元件內容將分述如 3.2.1、3.2.2 小節。

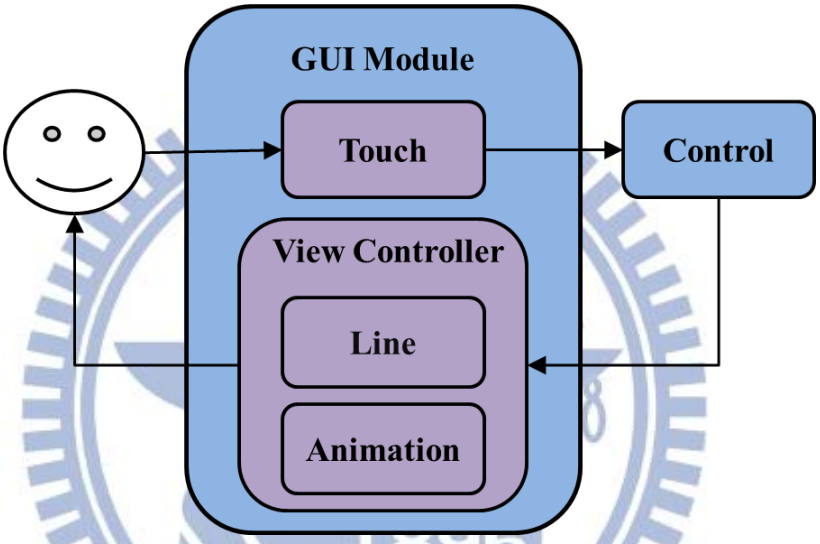


圖 8. GUI 模組架構圖

3.2.1. Touch

Touch 負責處理使用者的輸入，使用者對遊戲的操作主要透過觸碰手機螢幕。在 2.3.3 節中提到 iOS 支援許多手勢輸入，圖 9 展示在本系統中使用到的幾種手勢，以下將分項加以探討。

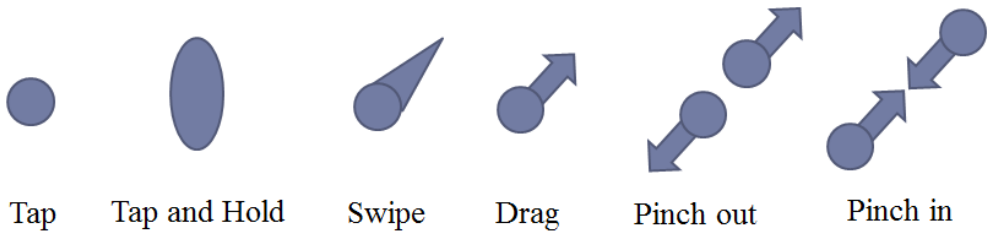


圖 9. 本系統使用的手勢示意圖

Tap

Tap 手勢在本系統中用於按鈕與下子行為。

- 按鈕

  當使用者的手指按下並抽離畫面上的按鈕時，Control 模組會處理此觸碰事件，並觸發對應的方法通知 GUI 模組執行下子的畫面顯示。

- 下子

  下子流程如下：

  (1) Control 模組收到棋盤上的 tap 事件。

  (2) 計算最接近觸碰位置的棋點座標。

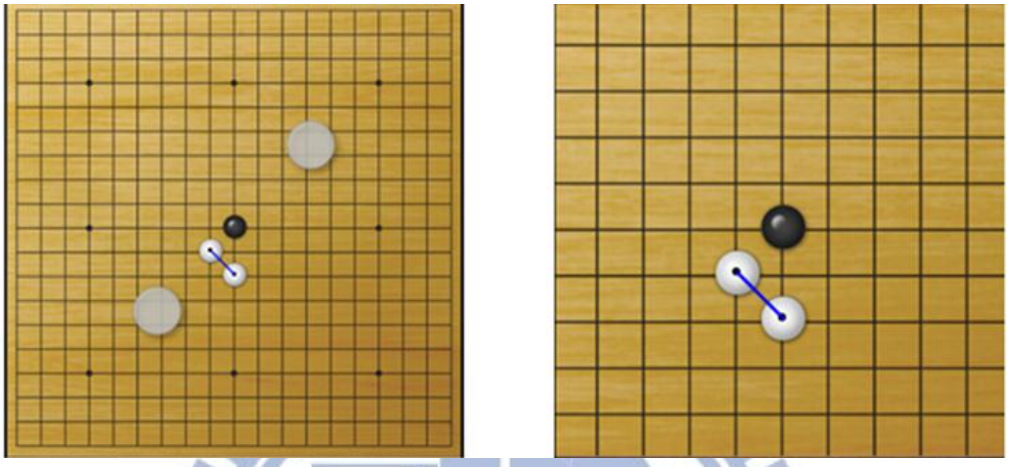
  (3) 檢查是否為合法步。

  (4) 在該棋點座標上放上目前下棋方顏色的子。

Pinch in/out

在本系統中，pinch in/out 手勢只用於放大縮小棋盤，如下方圖 10 所

示，(a)是棋盤放大前的預設畫面；(b)則是經過使用者 pinch out 手勢後，棋盤放大的畫面。



(a) (b)

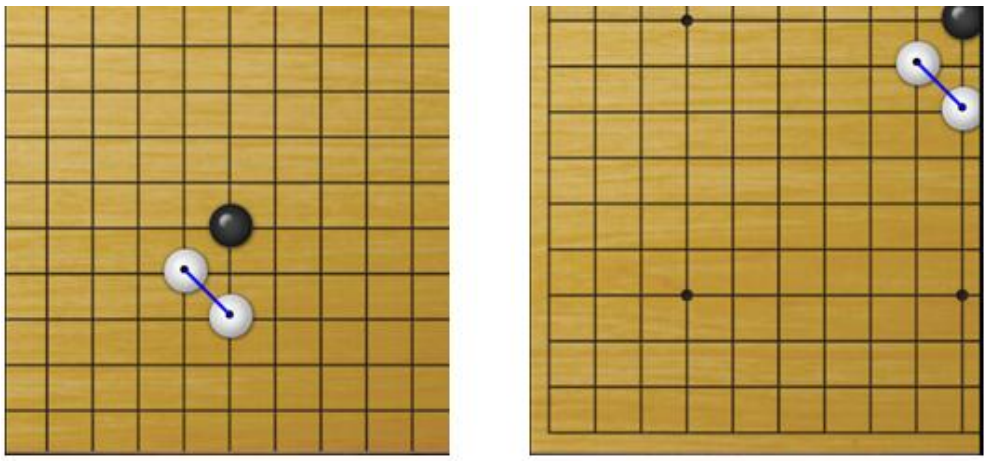
圖 10. (a) 預設棋盤畫面 (b) 棋盤放大畫面

Swipe

在本系統中，swipe 手勢會使用於移動棋盤可視範圍及拉出或隱藏選項表，分別於以下詳述。

- 移動棋盤可視範圍

  當棋盤被放大時，因為螢幕尺寸限制，使用者會看不見棋盤上某些區域，此時可以透過 swipe 手勢移動棋盤改變可視區域。如圖 11 中，(a)顯示棋盤中央區域；(b)顯示當使用者往右上角做 swipe 手勢後，使用者可視範圍則變成棋盤左下角。



(a) (b)

圖 11. (a) 可視範圍在棋盤中央 (b) 可視範圍移至左下角

- 拉出或隱藏選項表

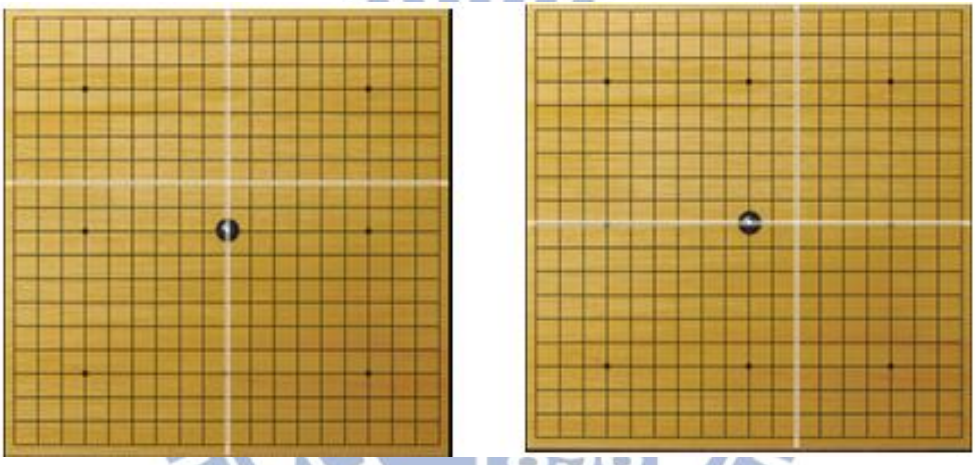
(a) (b)

圖 12. (a) 預設隱藏的選項表 (b) 滑動顯示選項按鈕

在選項表上有一些功能按鈕，例如「離開」、「重玩」、「悔棋」等。選項表預設狀態是隱藏，使用者可用 Swipe 手勢拉出選項表，系統在偵測到使用者在選項表所在範圍移動觸碰點時，選項表即被拉出；另外，使用者也可透過點擊(Tap)選項表所在範圍來顯示選項表。選項表在被拉開的五秒後回復隱藏狀態。在選項表拉開的狀態時，使用者也可用向下的 Swipe 手勢讓選項表隱藏。

Tap and Hold, Drag

使用者的手指觸碰螢幕一段時間(Tap and Hold)後，畫面上會出現十字輔助線，輔助線部分會在 3.2.2 節中再加以描述。出現十字輔助線時，使用者可拖曳(Drag)觸碰點來移動十字輔助線的中心點，當手指放開時 GUI 模組會執行下子動作。



(a) (b)

圖 13. (a) 出現十字輔助線 (b) 移動十字輔助線

如圖 13 中所示，(a)當使用者觸碰棋盤上座標(9, 7)的棋點一段時間後，畫面上出現白色十字輔助線；(b)使用者拖曳觸碰點將十字輔助線的中心點移至棋盤上座標(11, 9)的棋點。

3.2.2. View Controller

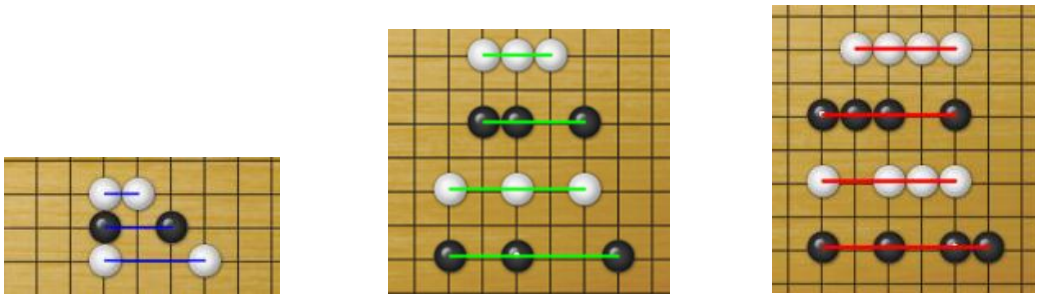
View Controller 是顯示並管理畫面上的 UI 物件，其中包含兩個元件：Line 和 Animation。Line 提供十字輔助線與提示線，Animation 則負責動畫部分，分述如下：

十字輔助線

在 iOS 行動裝置中，由於螢幕限制，所以棋盤尺寸較小，使用者較難下子於正確的位置。十字輔助線可讓使用者確認觸碰於棋盤上的棋點位置，使用十字輔助線的示意圖如 3.2.1 節中圖 13 所示。其顯示流程如下：

1. 使用者長按後(Tap and Hold)顯示十字輔助線。
2. 使用者拖曳觸碰點時，十字輔助線隨之移動。
3. 當使用者手指離開螢幕結束觸碰，十字輔助線從畫面上消失，並下子於該棋點。

提示線



(a) (b) (c)

圖 14. 提示線 (a) 活二 (b) 活三、死三 (c) 雙迫著、單迫著

在遊戲進行時，提示線能夠提示使用者六子棋棋局的盤面中較具有威脅性的棋型。圖 14 列出幾個畫上提示線的六子棋棋型，其中(a) 活二提示線為藍色； (b) 活三、死三提示線為綠色；(c) 標示迫著的輔助線為紅色。在遊戲中，提示線能讓使用者作為攻擊或防守的參考依據，如下方圖 15 所示。

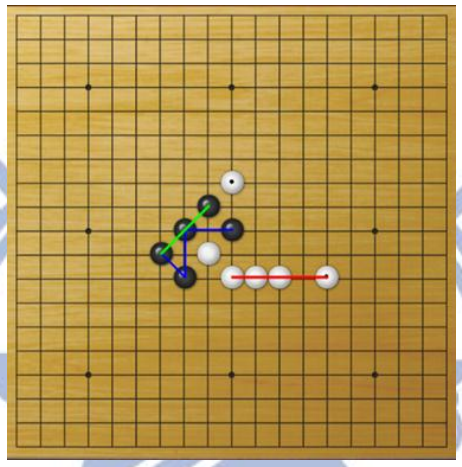


圖 15. 遊戲中六子棋棋局的提示線

動畫

在遊戲中，動畫效果透過 Animation 來實現。Animation 在遊戲中用於棋子移動，滑出選項表及圖片或訊息的淡入淡出。本程式利用 iOS 提供的 API—UIView 實作動畫功能，例如在詰棋闖關中淡入或淡出關卡選項表，實作一方法來實現動畫，在此方法中傳入以下參數：要移動的物件指標、目的地座標及移動完畢時物件的透明度。座標用兩個浮點數表示，透明度用一個浮點數—介於 0 到 1 之間的 alpha 值表示。

3.3 AI 程式之支援

為了增加棋類遊戲的趣味性，避免使用者找不到可對弈的玩家時，棋類 AI 是個不可或缺的角色。然而，程式開發人員通常另外使用 C/C++撰寫 AI，除了 C/C++普及度較高以外，C/C++編譯執行速度也較 objective-C 快[11]，我們實驗室所開發的 AI 程式 NCTU6 也是由 C++寫成。目前移植到 iOS 行動裝置的 AI 程式分為四個等級(rank 1~4)，rank l 的 AI 不使用搜尋，計算所需的空間和時間較少；rank4 則是目前棋力最強的版本。

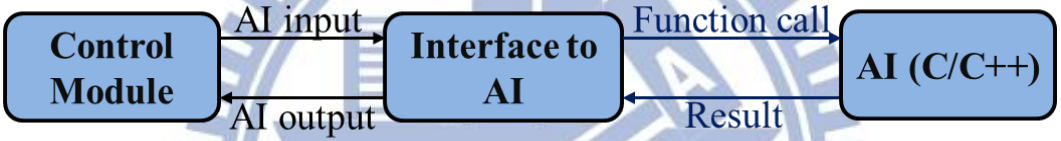


圖 16. AI 程式溝通示意圖

為了把 AI 移植到 iOS 行動裝置上，我們的做法是將 AI 主程式包裝成一個 AI 介面，控制模組和 AI 溝通方式如圖 16 所示。在需要 AI 運算時，控制模組會先新增一條線程去執行 AI 運算，在 AI input 中傳遞目前棋步資訊給 AI 介面。AI 介面將代表棋步的字串轉換成 AI 需要的輸入格式字串，例如：SGF 格式，再呼叫 AI 程式運算。當 AI 運算完成，AI 介面取得結果並使用 NSNotification 通知主線程，控制模組接收到通知後再取得結果，並要求 GUI 模組下子在指定位置。

3.4 知識庫模組

知識庫能讓應用程式使用者學習更多遊戲技巧與相關知識，是遊戲中很重要的角色。本專案提供詰棋譜資料庫的使用方法。由於棋譜資料量通常很龐大，而且常需要區分成不同難度，此模組結合 SQLite 提供方便開發時建立，查詢，修改，刪除資料的方法，並可調整亂數選取盤面匯入、是否加入 AI 支援對弈、在 GUI 的呈現方式等，知識庫模組架構如圖 17 所示。

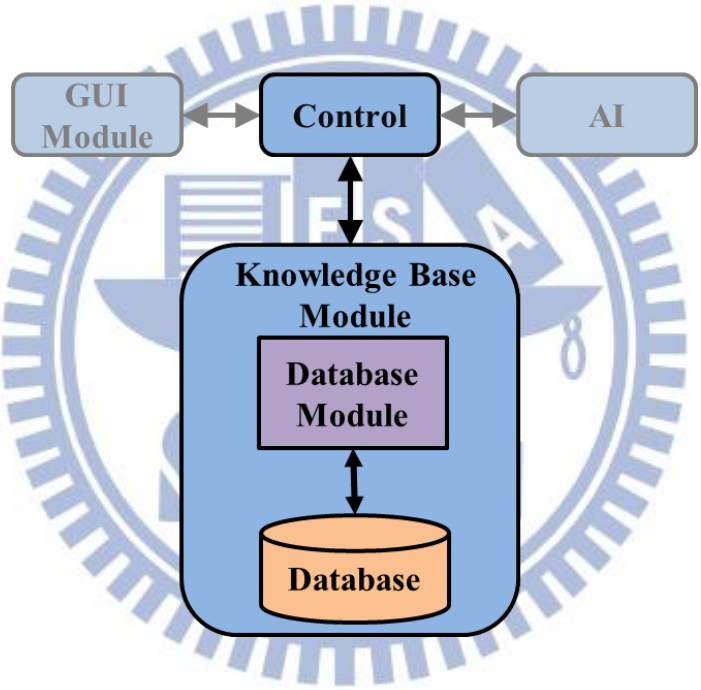


圖 17. 知識庫模組架構圖

在此模組中，主要使用 SQLite 管理詰棋資料。我們將所有詰棋譜儲存並匯入成為一個資料庫，在之後要使用詰棋譜時，Control 模組透過資料庫模組進行讀取，再透過 GUI 模組顯示詰棋譜的畫面；使用者在解詰棋時，則用 AI 程式進行對弈。

第四章、實作方法

在本章中，將介紹 iOS 六子棋遊戲程式的實作情況及遇到的問題，在 4.1 到 4.3 節中將按照不同模組分別描述之。

4.1 GUI 模組

在本次 iOS 六子棋程式中，GUI 模組的實作使得使用者介面較流暢，以下使用圖 18 至圖 22 作為範例說明。

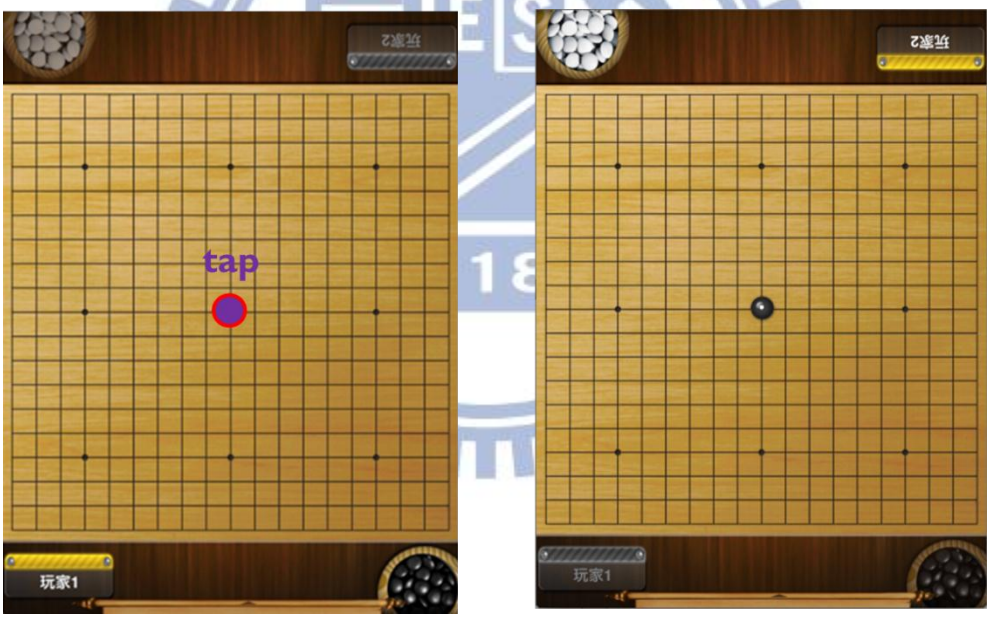


圖 18. 在棋盤上點擊(tap)下子

如圖 18 所示，使用者在棋盤中央點擊(tap)一下，當使用者手指離開螢幕時，即開始下子流程：計算使用者觸碰點靠近的棋點，檢查下子於該棋點是否為合法步，確認是合法步則開始移動棋子的動畫，將黑棋從右下角

移動到該棋點；若是白方下子回合，則由左上角開始移動白子。某方下子回合結束時，另一方的棋碗和名牌會變亮，提示輪該方下子，如圖中黑方下子前，黑方的棋碗和名牌是亮的；黑方下完第一手後，換成白方的棋碗和名牌變亮。另外，目前剛完成的該手棋子的中央會有小點(若是黑子標示白點，若是白子則標示黑點)，提示使用者剛剛下的棋子位置。

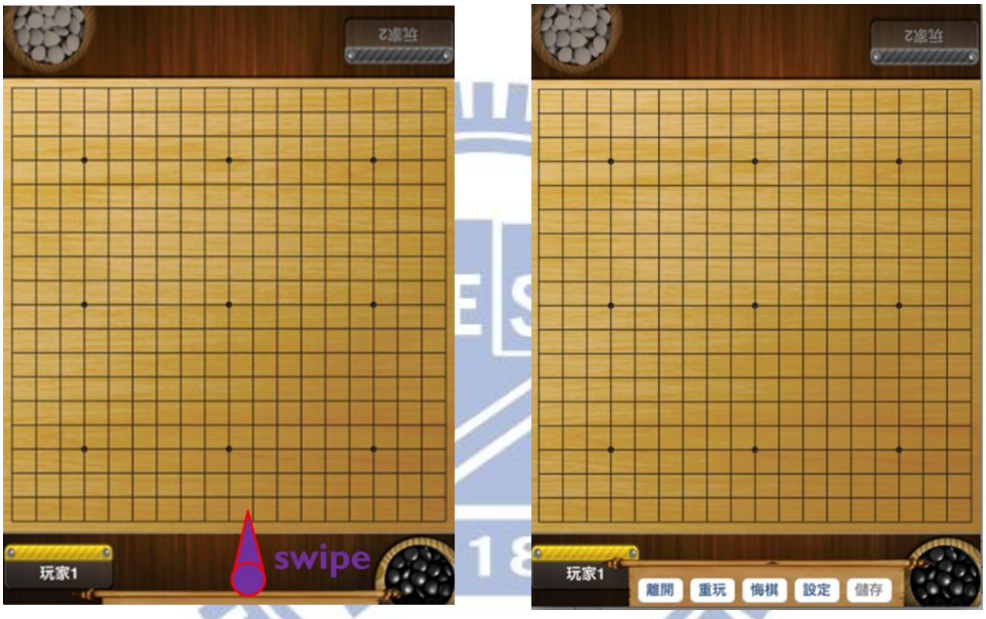


圖 19. 滑動(swipe)顯示選項表

在圖 19 中顯示使用者可以藉由滑動(swipe)手勢拉動顯示選項表，選項表上有「離開」、「重玩」、「悔棋」、「設定」、「儲存」按鈕，以下說明各按鈕功能。

- 離開：關閉正在進行的遊戲畫面，回到主頁面顯示。
- 重玩：關閉正在進行的遊戲畫面，初始遊戲資訊後再重新載入新局遊戲畫面。

- 悔棋：若在按下悔棋時，某方已完成下子，即黑方的第一手已經下了一子或之後某一方已下完兩子，則移除上一手下的子(除了黑方第一手是移除一子外，其餘都移除兩子)。但當某一方尚未結束其下子回合，即某方在要下兩子的回合中只放了一子，則只移除該子。
- 設定：按下設定鈕時，系統會跳出設定視窗，使用者可開啟或取消提示線功能和音效，如下方圖 20 所示。

圖 20. 設定視窗

- 儲存：將目前棋譜用字串表示存入資料庫，若棋盤上尚未有子則此按鈕顯示為灰色，按下後無作用。

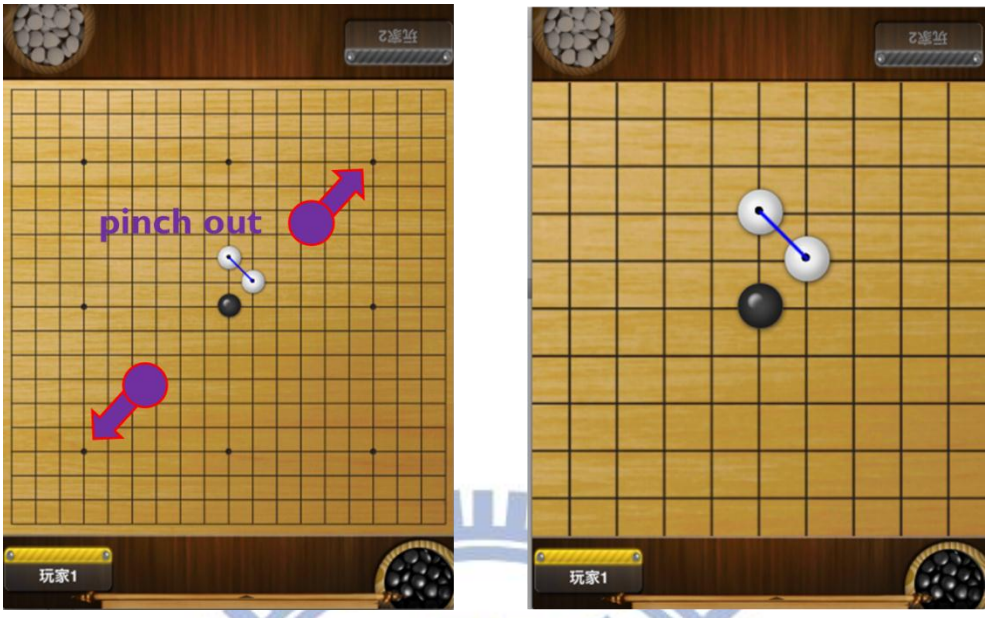


圖 21. 使用雙指 pinch out 放大棋盤

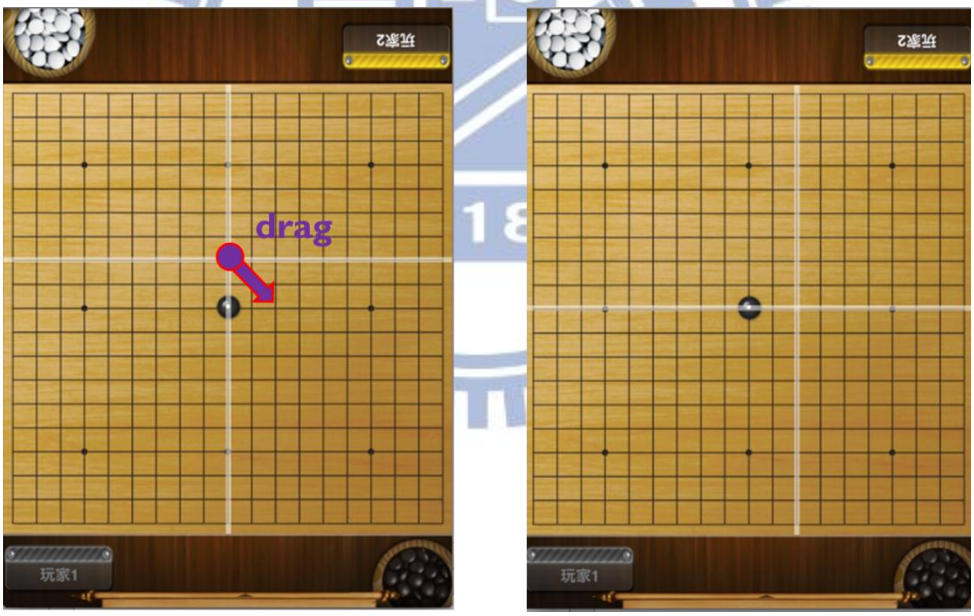


圖 22. 拖曳(drag)觸碰點以移動十字輔助線

在圖 21 中，使用者用雙指 pinch out 將棋盤畫面放大。而圖 22 中，使

用者按住螢幕中棋盤一段時間(tap and hold)，畫面上出現白色輔助線，拖曳觸碰點可移動輔助線確認下子位置。

由於不同的 iOS 裝置有不同的顯示器規格，目前市面上 iOS 行動裝置中最常見的是 iPhone 及 iPad。在本實作中，為了符合螢幕限制，會偵測目前裝置是 iPhone 或是 iPad 來調整合適的畫面物件尺寸，如圖 23 所示。

(a) (b)

圖 23. (a) iPhone 模擬器畫面 (b) iPad 模擬器畫面

4.2 AI 程式之支援

在本次實作中，移植了由本實驗室開發的 AI 程式—NCTU6，由於行動裝置的計算資源如處理器、記憶體等資源較一般個人電腦弱，所以在移

植前調整了一些參數和資料，此舉會弱化移植到行動裝置上的 AI 程式。

為了提供不同棋力的玩家可選擇和不同的 AI 程式對弈，目前的 AI 程式分為四個等級(此程式的 AI 等級配置在未來可能會有變動)，等級一不進行搜尋，是目前棋力最弱的版本，而等級四為棋力最強的版本。為了增加程式的趣味性，在等級一的實作中，AI 會從前三高評估分數的下法中隨機選擇一步，所以在相同的盤面中，AI 的應對方式可能不同。

iOS 版 AI 程式等級	1	2	3	4
與 NCTU6 對弈勝率(%)	2.84	12.93	28.13	38.49

表 2. iOS 版 AI 程式與 NCTU6 對弈勝率表

表 2 顯示 iOS 行動裝置上各等級的 AI 程式和 NCTU6 經過 700 盤對弈的勝率(經四捨五入至小數點後第二位)，可以發現雖然經過一些調整弱化，但是移植到 iOS 行動裝置上的 AI 程式仍維持一定程度的棋力。

不過在記憶體部分仍遇到一些困難，以下將先討論 iOS 行動裝置上的記憶體限制。

	iPad	**iPad2**	**new iPad**
Memory size	256 MB	512 MB	1024 MB

表 3. Apple iPad 規格[8]

	iPhone	iPhone 3G	iPhone 3GS	iPhone 4	iPhone 4S
Memory size	128 MB		256 MB	512 MB	

表 4. Apple iPhone 規格[8]

表 3 和表 4 列出一些 iOS 行動裝置的記憶體規格，其中記憶體容量最大的是 new iPad，其記憶體有約 1 Gigabyte，但和一般個人電腦相較之下仍較少。

	Main thread	Secondary thread
Stack size	1024 KB	512 KB

表 5. iOS 預設的線程(thread)堆疊(stack)容量[15]

本程式在需要 AI 運算時，會新增一條線程來計算，以免影響使用者介面的運作，表 5 則列出 iOS 線程堆疊容量的限制。另外，從幾次實際測試和網路蒐集的資料[13]發現當一個 iOS 應用程式記憶體使用量達到 20 至 30 Megabytes 時，應用程式可能會崩潰。由以上得知，在 iOS 行動裝置上執行 AI 計算時，必須謹慎使用記憶體。

在 NCTU6 程式中，記憶體主要使用在棋型表(pattern table)和搜尋樹(search tree)的資料結構。其中，較難刪減容量的部分是棋型表，以下將介紹 NCTU6 中的棋型表和縮減後的查詢方法。

目前正規的六子棋棋盤是 19x19，為了方便查詢一條線(19 個棋點)上棋型的分數以評估下子位置的好壞，NCTU6 使用棋型表儲存一條線上所

有可能棋型的評估值。此棋型表為了包含 19 個棋點所有可能產生的棋型，需要包含長度從 1 到 19 的單線棋型(line pattern)，以長度 19 的單線棋型為例，需用 2^{19} entries 表示棋子分布狀況，而每個 entry 需要 19 bytes 紀錄 19 個點的資訊(如攻擊、防守分數等)，所以此棋型表所需的記憶體容量至少為 $19 \times 2^{19} + 18 \times 2^{18} + \ldots + 1 \times 2^{1}$ bytes，在實際情況中，NCTU6 的棋型表大小約為 19 Megabytes。

為了縮減 NCTU6 中棋型表的記憶體使用量，所以最多只記錄到長度為 15 的單線棋型資訊，這樣棋型表所需記憶體量至少為 $15 \times 2^{15} + 14 \times 2^{14} + \ldots + 1 \times 2^{1}$ bytes，而本次實作中棋型表的實際使用量約為 1 Megabyte。

在查棋型表時，因為長度 15 的單線棋型無法完整表示長度為 19 的單線棋型，所以我們實作了一些方法來拆解棋型，在此介紹其中較簡單的一種方法：將一條線上 19 個棋點拆解成兩條長度為 15 的子線(sub-line)，分別為最左的 15 個棋點及最右的 15 個棋點，並採用查詢出較高的評估值。

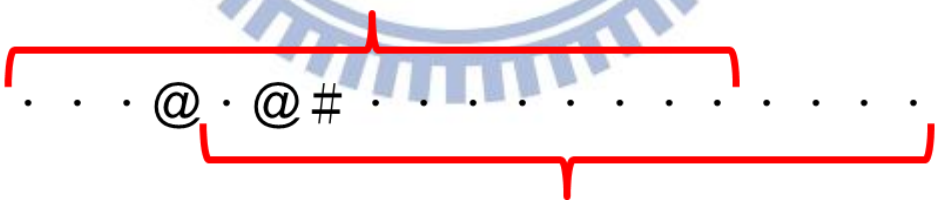


圖 24. 長度為 15 的棋型表查詢範例

如圖 24 所示，“·”代表棋盤上的空點，“@”代表我方的棋子，“#”代表要評估的下子位置。在查詢評估分數時，此長度為 19 的單線棋型分成長度為 15 的子線 (圖中兩條紅線包含的位置)，左邊子線查出下子後造成的棋型是活三；而右邊子線查出下子後造成的棋型是死二。在 NCTU6

中，活三的評估值較死二高，因此採用左邊子線查詢出的活三評估值。

用上述的方法和原本直接用完整長度的單線棋型查詢出的評估值會有些誤差，圖 25 顯示用各種子線長度進行實驗所得出的誤差比例折線圖。

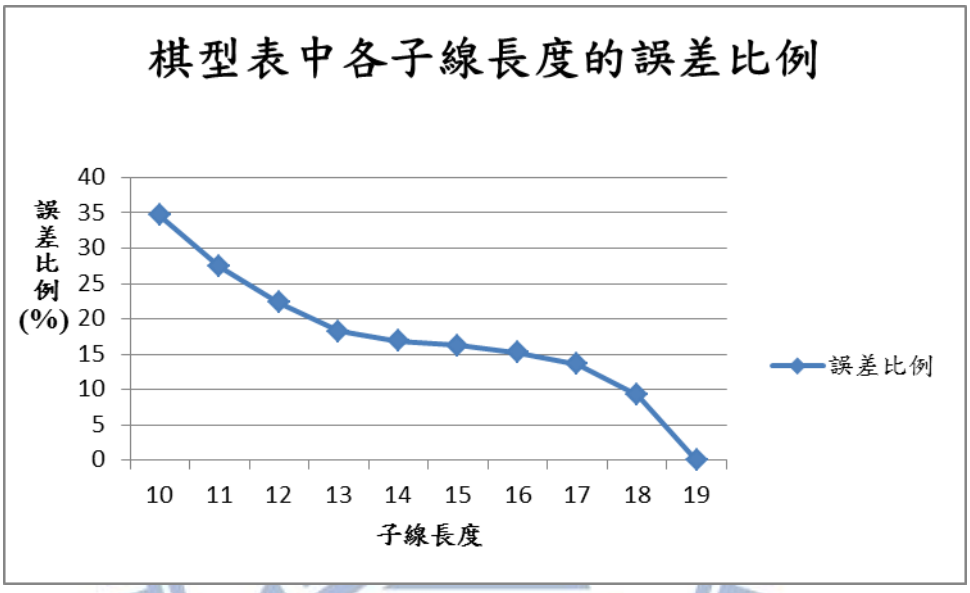


圖 25. 各子線長度的誤差比例折線圖

圖 25 的實驗數據中，在所有的棋型裡去除勝利棋型，使用前述方法查出的子線攻擊或防守評估值，相較於原本完整單線棋型查詢出的攻擊與防守的評估值，若兩者相異即是誤差。我們分析子線長度為 10 至 19 的棋型表誤差值，子線長度愈大則誤差比例愈小，使用前述方法查詢子線長度為 19 的評估值即與原本完整單線棋型查詢出的評估值完全相同。

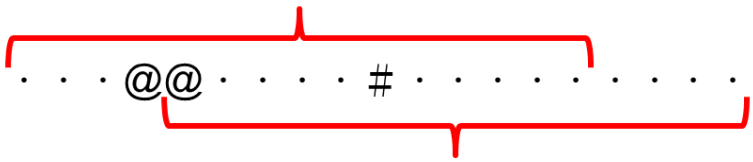


圖 26. 子線長度為 15 的棋型表查詢誤差範例

而子線長度 15 的棋型表是記憶體容量與誤差比例均可接受的。例如圖 26 所示，原本完整單線棋型查詢下子在“#”位置的評估值為活一，這是由於程式中判定下子在該位置對左邊“@@”已形成的死二無意義；但使用前述方法查詢時，使用左邊長度為 15 的子線查詢出活一，右邊長度為 15 的子線則查詢出死二，死二與活一的強度相當，然我們為了簡化而選擇較高的死二評估值，因此在此選擇死二為評估值，和原本查詢的評估值有誤差，但此種誤差的影響不大，實作中採用長度為 15 的棋型表。

	使用完整棋型表	使用縮減後的棋型表
記憶體消耗量	約 26.5 Megabytes	約 7.5 Megabytes
每手思考時間	約 58 秒	約 75 秒

表 6. AI 程式棋型表縮減前後的記憶體消耗量與思考時間比較表

如表 6 所示，在等級四 AI 程式執行時，使用縮減後子線長度為 15 的棋型表和原本完整的棋型表相比，雖然每手思考時間較長，但消耗的記憶體量減少許多。另外，目前應用程式下載檔案大小約為 8 Megabytes；去除 AI 程式與棋型表的應用程式下載檔則約為 6 Megabytes。

4.3 知識庫模組

在本次實作中，我們從網路上名為 Littlegolem[9]的伺服器上蒐集一些棋譜，並使用 NCTU6 協助找出約 6000 盤有趣的詰棋盤面，再由第二屆交大六子棋公開賽的冠軍—王智功先生挑選約 530 盤結棋題目匯入資料庫。

目前實作中，詰棋闖關遊戲分成 8 大關，每關有 12 局，前 4 大關每局的詰棋盤面皆是固定的；後 4 大關中，使用者選擇某局詰棋時，會從 10 個詰棋盤面中隨機挑選一個盤面供使用者解詰棋。在詰棋的資料庫中，8 大關的詰棋盤面與資訊分別存在 8 個 tables 中，前 4 個 tables 各有 12 個詰棋盤面，後 4 個 tables 各有 120 個盤面。

本實作中也將使用者資訊儲存於資料庫中，包含玩家名稱、是否開啟畫線及音效功能、詰棋闖關進度、玩家儲存的棋譜資訊。

第五章、遊戲製作與畫面展示

本章將描述實驗環境及展示目前六子棋遊戲流程中的畫面，其遊戲畫面設計與流程編排在未來可能會修改。程式實驗環境如下表所示：

Mac OS 版本	Mac OS X Snow Leopard 10.6.8, Mac OS X Mountain Lion10.8.0
開發環境	Xcode 4.2, Xcode 4.4.1
模擬器 iOS 版本	iOS 4.3, iOS 5.0, iOS 5.1
實機測試裝置	iPad 2, iPhone 4

表 7. 程式實驗環境

圖 27 至圖 36 展示 iOS 六子棋程式的遊戲畫面，畫面皆截取自 iPad 2，各遊戲畫面及流程將在以下詳述。

圖 27. 六子棋程式主頁面

圖 28. 設定頁面

- iOS 六子棋程式主頁面(圖 27)：

  頁面上有四個按鈕，前三個按鈕分別表示不同的遊戲模式。按下「挑戰電腦」按鈕後會開啟挑戰電腦的設定頁面如圖 28 所示;「雙人對弈」模式與「挑戰電腦」模式的遊戲畫面相似，只是換成兩個玩家對戰，本章不再加以敘述;「詰棋闖關」遊戲畫面則如圖 33 至圖 36 所示。

- 設定頁面(圖 28)：

  圖為挑戰電腦模式中，遊戲前的設定頁面，使用者可輸入玩家名稱；選擇 AI 等級；開啟或關閉劃線功能及音效；選擇使用者執棋顏色(先手或後手)。按下「開始遊戲」按鈕後，則開始遊戲，遊戲畫面如圖 29 至圖 32 所示。

- 開啟提示線功能的棋局畫面(下頁圖 29)：

  棋盤上有活二、死三和單迫著棋型，提示線分別標示為藍色、綠色、紅色。還有，黑方的棋碗及名牌發亮，提醒使用者目前輪黑方下子。

- 顯示選項表的畫面(下頁圖 30)：

  經過使用者點擊或滑動顯示出選項表，上面有離開、重玩、悔棋、設定、儲存的按鈕可使用。

- 使用十字輔助線畫面(下頁圖 31)：

  使用者長按於棋盤上，自動顯示白色的十字輔助線幫助使用者下子在正確位置。

- 遊戲結束畫面(下頁圖 32)：

  圖為棋局結束時，黑方獲勝的畫面。在棋盤上靠近黑方部份顯示「勝」訊息，白方則顯示「敗」訊息。下方的選項表會自動升起，此時使用者不能悔棋，但可以選擇其他按鈕:「離開」、「重玩」、「設定」、「儲存」。

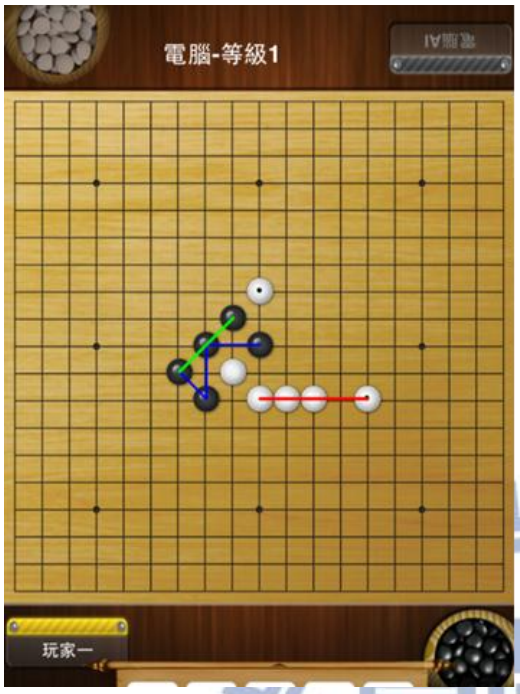


圖 29. 有提示線的棋局畫面

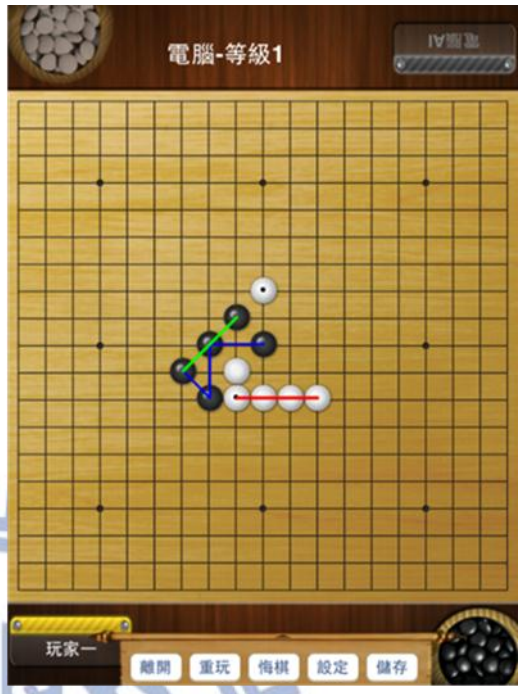


圖 30. 顯示選項表

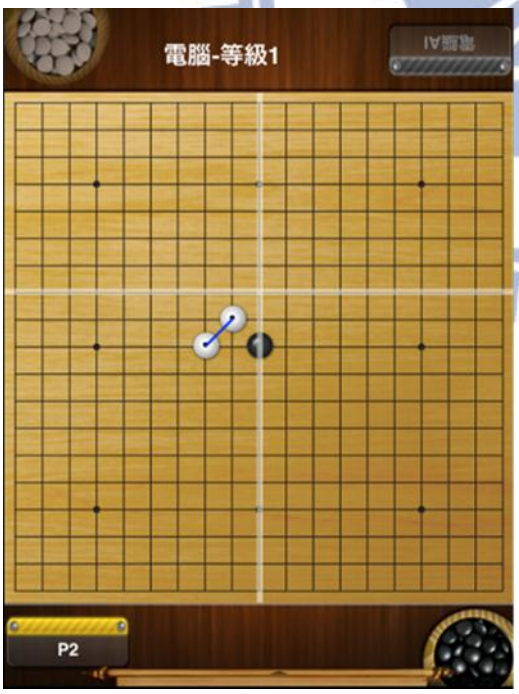


圖 31. 使用十字輔助線

圖 32. 勝利畫面

圖 33. 關卡選擇

圖 34. 棋局選擇

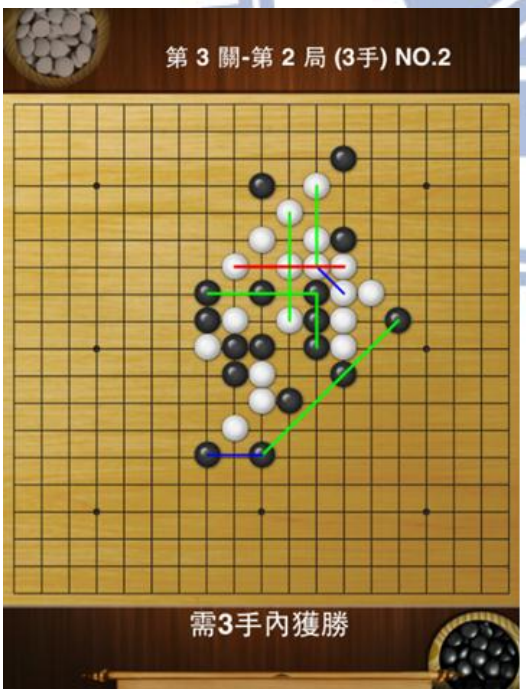


圖 35. 詰棋遊戲畫面

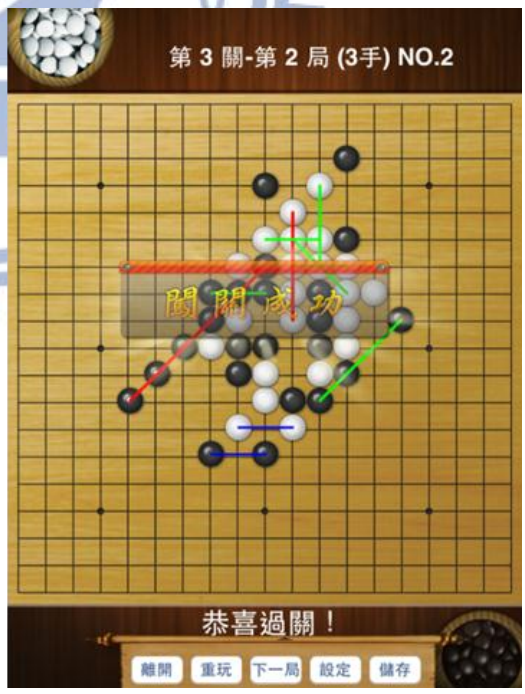


圖 36. 成功解詰棋畫面

- 關卡選擇(圖 33)：

  此為詰棋闖關模式中的關卡選擇，目前共八大關，使用者可用上下滑動來檢視各關卡，畫面上使用者闖關進度是完成第三關的第一局，每關需完成 12 局詰棋遊戲才能解鎖下一關，尚未解鎖的關卡則顯示灰色的鎖，使用者按下此種關卡的按鈕無作用。

- 棋局選擇(圖 34)：

  目前位於詰棋闖關的第三大關中，因為使用者目前只完成第一局，所以可選的棋局只有第 1 局及的 2 局，剩下的題目皆尚未解鎖顯示灰色的鎖，按下尚未解鎖的棋局按鈕則無反應。

- 詰棋遊戲畫面(圖 35)：

  詰棋闖關模式中的棋局畫面，上方顯示這是第三關第 2 局，使用者需在 3 手內獲勝才算成功解題。此時若使用者下完一手，下方的「需 3 手內獲勝」訊息則會變為「需 2 手內獲勝」提示使用者剩餘的手數。

- 成功解詰棋畫面(圖 36)：

  當使用者成功解開詰棋棋局，出現闖關成功的畫面，選項表會自動升起，選項表上的「悔棋」按鈕會被「下一局」按鈕取代，使用者可按「下一局」按鈕開始下一局詰棋遊戲。

由以上遊戲畫面展示與說明，可以看出此 iOS 六子棋程式有流暢的使用者介面，也支援和使用者對弈的 AI 程式與詰棋遊戲。在經過實驗室裡幾位同學試玩後，覺得挑戰電腦與詰棋闖關很有趣也有挑戰性，而且在遊戲過程中也可以不斷學習六子棋相關知識與增長棋力。

第六章、結論與未來展望

在本篇論文中，描述在 iOS 平台上開發的六子棋遊戲，並提出系統設計架構，包含 GUI 模組、Control 模組、AI 程式支援及知識庫模組，此架構也可以套用在 iOS 行動裝置上其它棋類遊戲程式的開發上。

此 iOS 六子棋程式具有友善的使用者介面，使用者可用多種手勢操作遊戲，十字輔助線及提示線能幫助使用者順利下子與思考。在本程式中也提供詰棋闖關模式讓使用者可以自我訓練，還有記錄了使用者資訊，如玩家名稱、闖關進度等。

為了移植 AI 程式到 iOS 行動裝置上，我們縮減棋型表的容量。在 NCTU6 中棋型表原本大小約為 19 Megabytes，縮減後變成約 1 Megabyte，並使用新的查表方法，在實作方法中也有實驗切割成不同長度的子線在查表時的誤差比例。

這次的設計方法使得在 iOS 行動裝置上開發其它棋類遊戲變得更容易，並提供其它平台的遊戲設計做為參考。目前，本實驗室考慮繼續在 iOS 行動裝置上開發象棋、圍棋等棋類遊戲。

本程式在未來可以延伸實作一些部分，例如：提供使用者在網路上連線對弈功能，加入使用者詰棋闖關排行榜系統，增加更多的詰棋盤面並調整關卡設計。另外，將來也可以分割模組內的程式，發展成開發 iOS 裝置上棋類遊戲的基本框架。

參考文獻

[1] Apple Inc., The Objective-C Programming Language. Available at https://developer.apple.com/library/mac/#documentation/Cocoa/Conceptual/ObjectiveC/Introduction/introObjectiveC.html.

[2] Cox, Brad, Object-Oriented Programming, An Evolutionary Approach, Addison-Wesley, 1987

[3] Event Handling Guide for iOS. Available at http://developer.apple.com/library/ios/#DOCUMENTATION/EventHandling/Conceptual/EventHandlingiPhoneOS/GestureRecognizers/GestureRecognizers.html

[4] International Data Corporation, IDC. Available at http://www.idc.com.

[5] iOS Developer Library, framework. Available at http://developer.apple.com/library/ios/navigation/#section=Frameworks

[6] iOS Technology Overview. Available at http://developer.apple.com/library/ios/#documentation/Miscellaneous/Conceptual/iPhoneOSTechOverview/Introduction/Introduction.html

[7] Lin, P.-H., and Wu, I.-C., NCTU6 Wins Man-Machine Connect6 Championship 2009, ICGA Journal, Vol. 32(4), pp. 230–232, 2009.

[8] List of iOS devices from Wikipedia. Available at http://en.wikipedia.org/wiki/List_of_iOS_devices.

[9] Littlegolem. Available at http://www.littlegolem.net/jsp/index.jsp

[10] Mobile Statistics. Available at http://www.mobilestatistics.com/

[11] Armin Roehrl, and Stefan Schmiedl, Linux Journal, Objective-C: the More

Flexible C++. Available at http://www.linuxjournal.com/article/6009.

[12] SQLite. Available at http://www.sqlite.org/.

[13] Stackoverflow.com, Memory uses limit on iPhone, 2010.

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

[15] Threading Programming Guide. Available at http://developer.apple.com/library/ios/#documentation/Cocoa/Conceptual/Multithreading/CreatingThreads/CreatingThreads.html.

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

[17] 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.

[18] Wu, I.-C., and Lin, P.-H., NCTU6-Lite Wins Connect6 Tournament, ICGA Journal, Vol. 31(4), pp. 240–243, 2008.

[19] Wu, I.-C., and Lin, P.-H., Relevance-Zone-Oriented Proof Search for Connect6, to appear in the IEEE Transactions on Computational Intelligence and AI in Games, 2010.

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