**HomeCraft**

**以擴增實境（AR）技術實現室內居家佈置**

指導老師：陳慶永 教授

參賽組員：宋睿哲、許勇晉、郭祐勛

**1.前言**

**1.1研究動機**

在城市人口密集化的趨勢越來越大的這個時代，幾乎一座城市的精華都會集中在某一個地段，因此到導致土地的價格節節上漲，為了擴展生活的空間，數也數不盡高樓大廈紛紛林立而起，如此一來，人們的生活的家可不是那簡單就能擺設的了，為了建築中的空間最大化利用，建築商往往不會將通道設計得很寬大，如果要擺設家具往往要考量通道是否能容納家具的尺寸，以免發生家具到貨了卻進不了家門的情形，更甚者可能需要租借一臺吊車才能搬運進來，更別提當家具搬進來了之後，完全不適合家裡的情形，這樣花費的人力還有時間的成本又更大了。

**1.2研究目的**

為了避免家具擺放不合適的情形發生，家具大廠IKEA在2014年推出一款App，能讓客戶在買家具之前，利用AR的技術，透過相機看家具擺在居家的樣子，而不用實際搬家具，且如果不喜歡擺設，也不用費力氣去搬移，預先就可以知道擺設的樣子；但前提是你要先有一本IKEA家具項目的書，把書放到你想要家具放的位置，然後再把相機對著那本書，才能夠做家具的擺設。

但IKEA為了要增加自家產品的銷售量而把那款App和書進行了綑綁，所以為了解決需要書才能進行擺設的問題，我們希望利用iOS 11推出的ARKit來開發一個任何地方都能進行家具擺設的App。本專題的目的是設計一個簡單且不受限制的室內居家佈置App，概念如圖1概念圖所示。

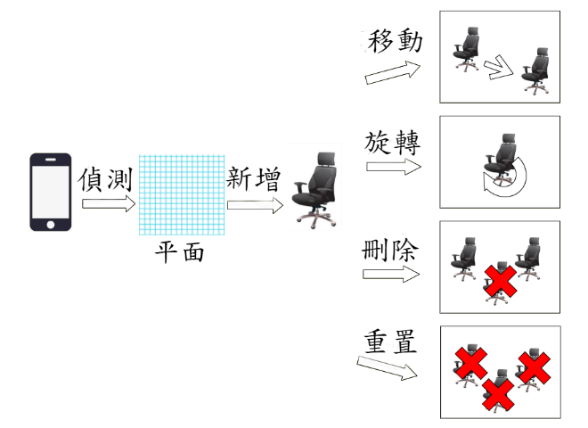


圖1. 系統概念圖

**2.AR簡介**

AR技術中文為擴增實境（Augmented Reality，簡稱AR），是指透過攝影機影像的位置及角度精算並加上圖像分析技術，讓螢幕上的虛擬世界能夠與現實世界場景進行結合與互動的技術。擴增實境目前已經被廣泛應在生活中，像是導航、娛樂、模擬練習等，由 Niantic、Pokémon 與任天堂三方打造的遊戲 Pokémon Go，就是主打擴增實境技術，玩家可以利用手機的鏡頭來查看周遭的神奇寶貝再點擊手機或裝置捕捉。擴增實境的應用非常多元化，因為可以輕易滲入到生活中，所以普及率相對比虛擬實境高出許多。在去年的WWDC 2017上，蘋果發表了ARKit，讓更多的開發者能進入到AR的世界中。

**3.ARKit**

**3.1什麼是ARKit?**

ARKit是一系列新的iOS開發工具，針對iOS系統開發者，能更方便AR環境開發的SDK。過去開發者建置AR技術時會面臨巨大的技術瓶頸，但『ARKit』提供大部分的基礎建設，大幅縮短開發所需的時間。

ARKit可幫我們做的事：

1.世界追踪：追踪設備的方向和位置，並檢測真實世界的表面。

2.圖像檢測：在真實世界環境中識別的圖像。

3.面部追踪：檢測到的人臉姿態，拓撲和表情的信息。

**3.2ARKit使用的開發工具、執行的硬體規格**

開發工具為Xcode 9或以上版本，運行ARKit的設備必須要A9或以上的處理器，且系統版本須要iOS 11或更高版本。例如：iphone 6s以上、iPad(2017)等。

**3.3ARKit世界組成**

ARKit的世界是由許多SCNScene所構成的3D世界，而SCNScene是由許多SCNNode所組成。而這些SCNNode。

**3.4ARKit與SceneKit的框架關係圖**

ARKit負責將真實世界畫面轉變為一個3D場景，這一個轉變的過程主要分為兩個環節：由ARCamera負責捕捉攝像頭畫面，由ARSession負責搭建3D場景。

SceneKit負責將虛擬物體現實在3D場景。ARSCNView顯示3D虛擬增強現實的視圖。關係如圖2 ARKit與SceneKit的框架關係圖所示。

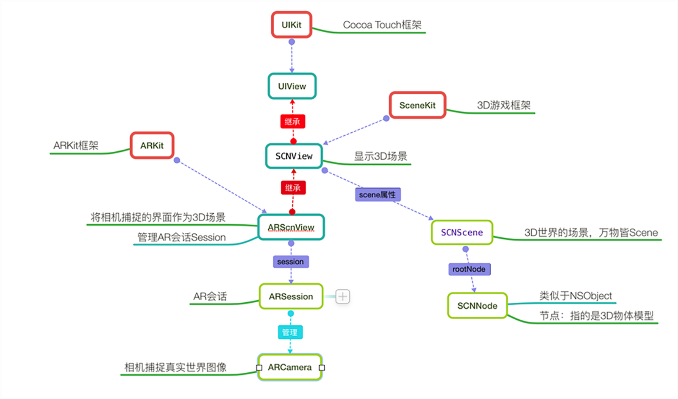


圖2. ARKit與SceneKit的框架關係圖

**3.5ARKit座標**

以手機直立為基準，面相手機，手機中心點為原點(0,0,0)，左右水平線為X軸，方向為由左至右，上下垂直線為Y軸，方向為由下至上，前後水平線為Z軸，方向為由前至後，如圖3 ARKit座標圖所示。

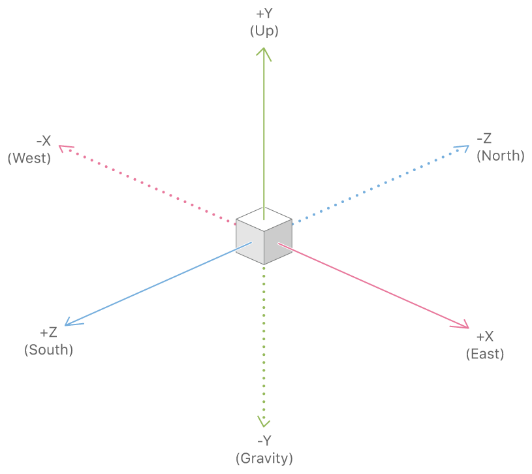


圖3. ARKit座標圖

**3.6平面偵測**

ARKit的基本過程涉及從iOS設備相機讀取視頻幀，對於每幀處理圖像並提取特徵點。一旦你有了圖像的特徵，你就可以跟踪多個幀的特徵，當用戶在世界各地移動時，你可以獲取這些相應的點並估計3D姿態信息。

對於平面檢測，一旦3D中有多個特徵點，您就可以嘗試將平面擬合到這些點上，並在比例，方向和位置方面找到最佳匹配。

影響特徵點(平面偵測)因素：

1. 光線不足

2. 缺乏紋理

3. 快速移動

**4.系統架構**

此App提供四個功能，如圖4使用案例圖所示。分別為新增、移動、旋轉、刪除。



圖4. 使用案例圖

使用者開啟App後，需找一塊平面，拿著相機對著地面，如果相機偵測有平面，畫面會顯示藍色的網狀方格子。

偵測完平面後，使用者可以點選左上角的選單，首先先選擇家具的種類，例如：家電、桌子、椅子、床…等。點選種類之後，可以選擇不同的樣式，，選完後，在平面上點一下，就可以新增家具。如圖5新增之活動圖所示。



圖5. 新增之活動圖

放置完家具後，使用者可以選擇對家具進行移動，直接拖移家具即可，如圖6移動之活動圖所示。



圖6. 移動之活動圖

如果使用者想要旋轉家具，必須先點擊家具，才可以旋轉。旋轉有兩種方式，一種是在點擊家具後，右上角會出現兩個按鈕，Delete和Rotate，點選Rotate按鈕，每點一次就可以讓家具向逆時針旋轉90度；另一種旋轉方式是用兩指做旋轉，一隻手指點在家具上，另一隻手之以畫圓的方式控制家具的旋轉角度。如圖7旋轉之活動圖所示。



圖7. 旋轉之活動圖

如要刪除家具，也分成兩種刪除，一種是刪除單一家具，一種是刪除全部的家具。單一刪除家具，要先點選家具，點選後，右上角會出現兩個按鈕，Delete和Rotate，點選Delete按鈕後即可刪除單一家具。刪除全部家具，只需點選右下角的RESET按鈕，即可刪除全部家具。如圖8刪除之活動圖所示。



圖8. 刪除之活動圖

家具、平面和ARSCNView的關係如圖9物件圖所示。ARSCNView負責現實與虛擬的圖象，所以ARSCNView可以顯示0個或多個平面，也可以顯示0個或多個家具物件。一個平面可以放置0個或多個家具物件，而家具物件只能放在一個平面上。



圖9. 物件圖

**5.ARKit實作**

**5.1產生家具於鏡頭前**

首先先新增一個3D視圖SCNScene，來源為art.scnassets/Anni chan資料夾底下的Anni chan.dae 3D模型，如圖10第22行。之後我們將SCNSCcene的所有節點加到一個SCNNode，如圖10第24到27行。調整圖形大小用modelNode.scale= SCNVector3(0.05,0.05,0.05)，SCNVector3中的數字為X,Y,Z軸的縮放倍數，小於0縮小，大於0放大，如圖10第29行。modelNode.position=SCNVector3Make(0, -0.5, -0.5)為設定模型的座標，SCNVector3Make中的三個參數，分別為X座標、Y座標、Z座標，設成(0, -0.5, -0.5)是為了讓模型保持在鏡頭前，手機的中心點為(0, 0, 0)，如果不改座標，模型會出現在中心點以上，有可能會被切到，所以Y軸往下0.5，而如果不改Z軸，模型會像是貼者鏡頭，所以要往後才能看到模型。如圖10第31行。最後再把模型加入ARSCNView中，如圖10第33行。

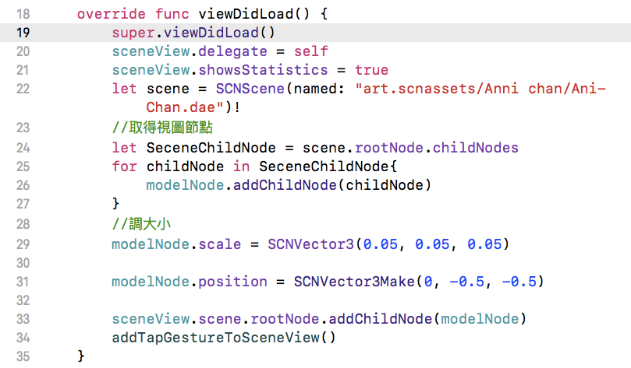


圖10. 產生虛擬物件於鏡頭前程式碼

**5.2點擊方式加入家具**

首先，先實作UITapGestureRecognizer函式，如圖11第109至119行，並取得手指點擊的位置，如圖11第111行所示。然而手指點及位置為2D，AR世界為3D，那要如何轉換呢？ARSCNView提供了兩個hitTest()方法，一個是hitTest( \_, types:)，回傳與真實世界平面相交的所有交點，回傳型態為ARHitTestResult。另一個是hitTest(\_:options:)，回傳與虛擬物件相交的所有交點，回傳型態為SCNHitTestResult。

如何判斷手指與真實平面和虛擬物件有交點，是透過相機發出一條搜索光線，只要與平面或家具有交點就回傳。

所以我們透過hitTest( \_, types:)將手指的位置轉換至3D座標，並取出符合的第一個交點，如圖11第111、112行。之後再創造出3D物件，也就是家具。

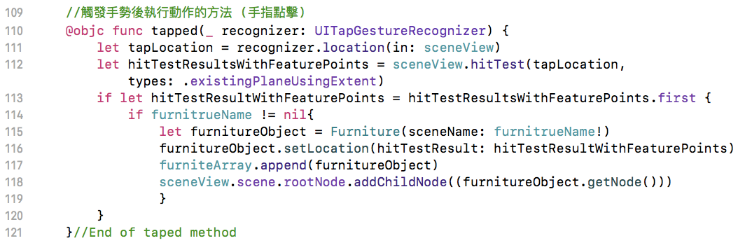


圖11. 新增之程式碼

竟然要創造一個物件，就必須要有類別，所以我們定義了一個Furniture類別，定義如圖12所示。我們透過建構者函式接收一個字串，如圖12第19行，再根據字串去建構SCNscene，然後把SCNscene的所有節點加入SCNcene，並取名為「AddedObject」，如圖12第20行至24行。之後我們定義兩個函式，一個是setLocation()用來設定家具的座標，這個函式接收了一個參數，參數為搜索光線與平面的交點，參數經過世界座標的轉換後，把家具座標設定為參數。另一個函式是getNode()，返回家具的節點。

讓我們回到圖11的創造3D物件的部分，首先先判斷furnitureName是否有值，有值的話，新增家具，並呼叫家具的setLocation()方法，設定家具座標，然後把家具加入一個陣列，以便之後做清除，最後再利用家具的getNode()將家具加入sceneView中。如圖11第113行至第118行。



圖12. Furniture類別

**5.3移動家具**

首先，先實作UIPanGestureRecognizer函式，如圖13第232行至261行，UIPanGestureRecognizer的手勢分成三種狀態，第一個狀態為began，是在你移動前，手指點螢幕並按住不放時的情況。大二個狀態為changed，是你手指在螢幕上移動時的情況。第三個狀態為ended，是在你移動完，手指從螢幕上放開時的情況。

移動家具，首先必須確定你手指有移動到家具，所以在began的狀態，要判斷手指點螢幕不放的位置是否有家具，有才移動，如圖13第236行至248行。先取出手指在螢幕上的二維位置，再利用ARSCNView提供的hitTest()方法，來轉成3D的座標，但因為是要確定手指的搜索光線是否有和虛擬的物件有交點，所以選擇hitTest(\_:options:)，如圖13第237行。之後取出與搜索光線相交的第一個點，取完之後判斷節點的名字，如果有名字，代表有點到家具，沒有則代表點到平面，所以不做任何事。判斷完交點的名字後，如果點到的交點剛好是根節點，將nodeIsPaned設為交點，如果不是，我們要取他的父節點，直到根節點再把nodeIsPaned設為根節點，根節點的名字我們這邊取「AddedObject」，為什麼要取交點的根節點，有些3D的模型會由一些3D模型組合起來，例如：一張床的3D模型，可以由床的木板、枕頭、棉被組成，而這些3D模型都有各至的座標，如圖15所示，棉被的節點名稱為「group\_4」，整體床的節點，也是根節點的名字為「SketchUp」。如果不取根節點的話，會如圖14所示，家具會分裂，可能指移動到家具的某個部分。

確定手指點螢幕不放的位置有家具後，開始移動手指，所以執行changed狀態，做家具的座標改變，而這裡我們呼叫moveNodeByPanGR的函式來改變家具的座標。如圖13第251至252行。最後手指放開動作執行ended狀態，一樣要更改家具的座標，所以呼叫moveNodeByPanGR的函式，最後把用到的參數清空，以便下次呼叫內容為正確，如圖13第253行至256行。

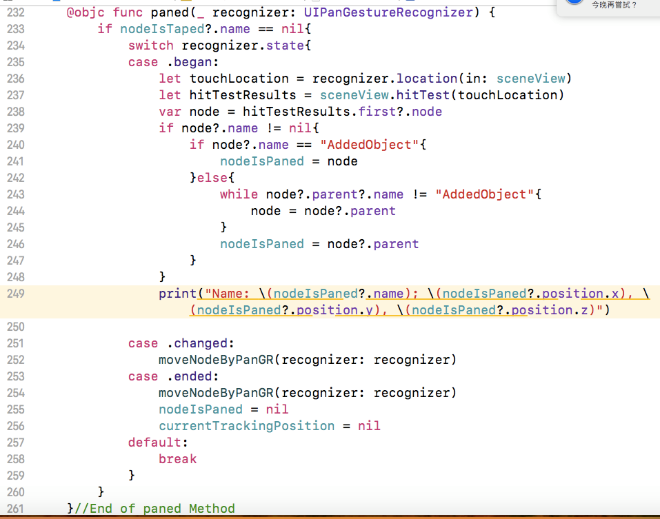


圖13. 移動之程式碼

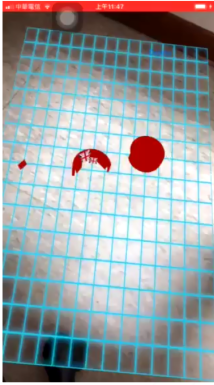


圖14. 非根節點的移動結果

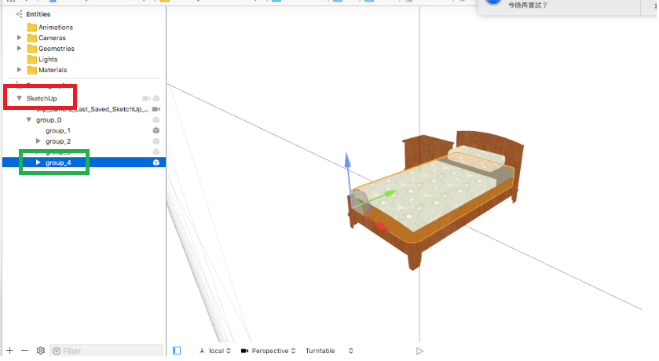


圖15. 家具節點樹狀架構

moveNodeByPanGR函式一開始利用nodeIsPaned判斷手指點螢幕不放的位置是否有家具，取手指在螢幕上的座標，這個座標會隨著手指移動而改變，如圖16第264行，然後利用ARSCNView的hitTest( \_, types:)方法，取得與現實世界的交點。這裡使用hitTest( \_, types:)方發法是因為我們要讓物體在平面上移動，而平面是以現實平面偵測出來的，所以取與現實的交點，並取第一個相交的點，如圖16第265至267行。取得交點後進行世界座標的轉換，如圖16第268行。最後直接將家具座標改成交點座標，如圖16第269行。

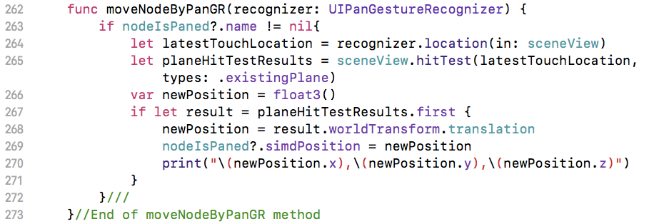


圖16. 改變家具位置之程式碼

**5.4旋轉家具**

首先，先實作UIRotationGestureRecognizer函式，如圖17第275行至第293行。旋轉的手勢為一隻手指點著物件不放，另一手指以畫圓方式移動，所以先取得手指點擊的位置，如圖17第277行所示。取得後利用hitTest(\_:options:)找虛擬家具相交的第一個交點，並判斷是否有交點，如圖17第278行至第280行。有交點的話，開始判斷是否為根節點，如果是，取出另一隻手指移動的角度，再把node.eulerAngles改成手指移動的角度，這邊我們只改變Y軸，讓家具能在平面上做旋轉，如圖17第281行至第284行。如果不是根節點，一直取到根節點後再把node.eulerAngles改成手指移動的角度，不然會只旋轉家具的某部分，如圖17第285行至第289行。

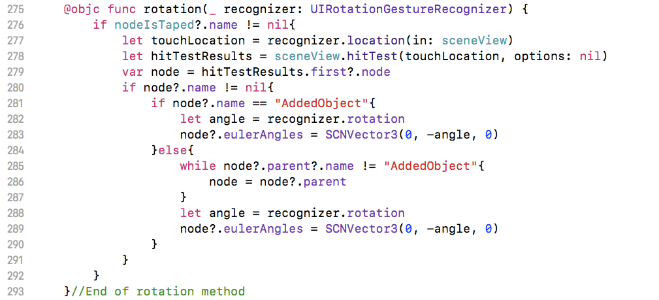


圖17. 旋轉程式碼

**5.5刪除，重置家具**

刪除家具為刪除單一家具，首先要知道我們要刪除哪一個家具，所以我們透過改寫UITapGestureRecognizer函式，來找到要刪除的家具，改寫如圖18所示。再新增物件前，利用hitTest(\_:options:)找虛擬家具相交的第一個交點，如果沒有，才做新增家具，如果有，代表點擊的位置有家具，把nodeIsTaped設為交點，來做為刪除的依據。

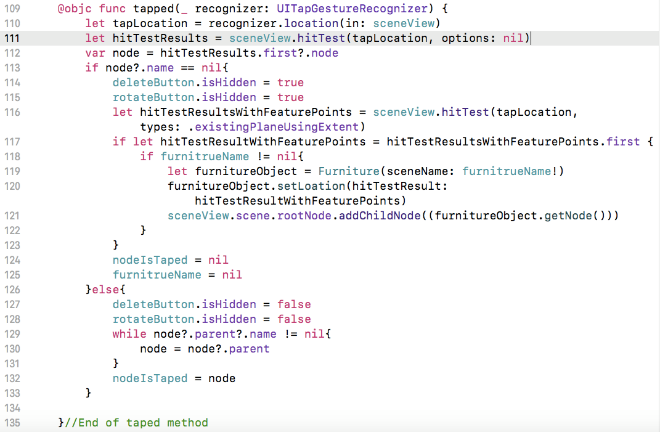


圖18. 改寫新增程式碼

已經知道要刪除哪一個家具後，從其父節點的子節點數組中移除該節點，每個家具的根節點的父節點為AR世界的大地座標，所以從大地座標移除家具節點，如圖19第303行所示。

清除為清除所有的家具，所以我們利用新增時，把家具加入的陣列，來進行所有家具的刪除，如圖19第315行至第317行。最後再把陣列清空，如圖19第318行。

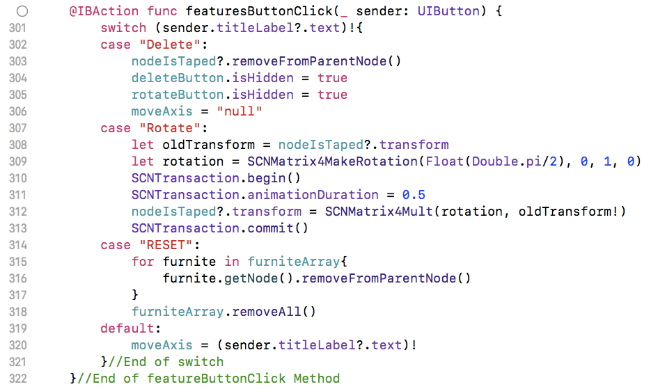
****

圖19. 刪除&重置程式碼

**6.結論與成果**

AR開發的技術現階段才正開始起步，不論是Apple陣營的ARKit，或Google陣營的ARCore，都是在2017後半年才發表並釋出，因此能夠做到的事情是非常有限的，做出來得實際效果也未必能達到人們的期望，但這些SDK能夠讓我們快速地進入AR世界，縱使我們的程式開發經驗不足，也能簡單的做出一個AR相關的APP，而這些SDK也不斷地更新中，目前ARKit已經更新到1.5版，它也帶來更多的功能，讓我們能在AR世界中做出更多不同的事情。

目前ARKit還是有些缺陷，在進行環境偵測時，必須非常注意燈光以及地板的材質，一些會反光的地面或者太暗的地方，可能就會使得它偵測不到，還有產生出來的AR物件，想要完美的融入現實還是有些難度，人類所畫出來的東西，跟現實還是有差距的，後者我想可能難以克服，但前者，我們相信隨著ARKit的更新，一定能夠改善。未來這套SDK會如何發展，是一件讓人值得關注的一件事。



圖20. 實作成果圖1



圖21. 實作成果圖2

****

圖22. 實作成果圖3

****

圖23. 實作成果圖4

**參考文獻**

[1] weicy（2017年12月14日）ARKit從入門到精通（2）-ARKit工作原理及流程介紹**。**取自：<https://hk.saowen.com/a/e092ca0288b0e4943249ffb6fdab82f3a0b88eac3d6d298f7c816406decc8456>

[2] Christopher Webb-Orenstein(2017,Aug 27)ARKit and CoreLocation: Part One. Message posted to：

<https://medium.com/journey-of-one-thousand-apps/arkit-and-corelocation-part-one-fc7cb2fa0150>

[3] Apple Inc (2018) ARKit Documentation. Message posted to：

<https://developer.apple.com/documentation/arkit/>

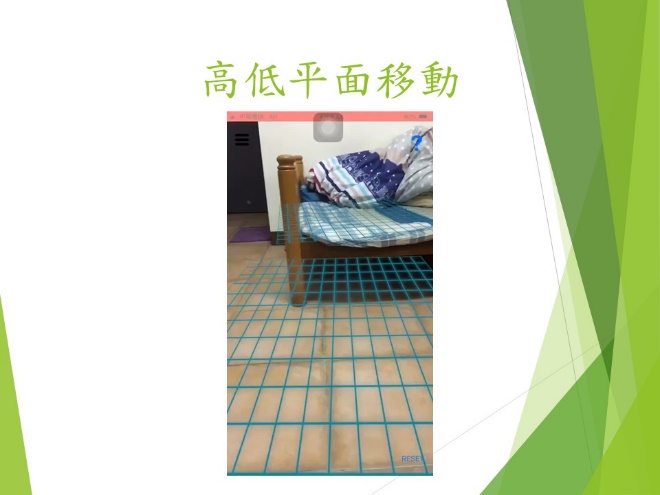
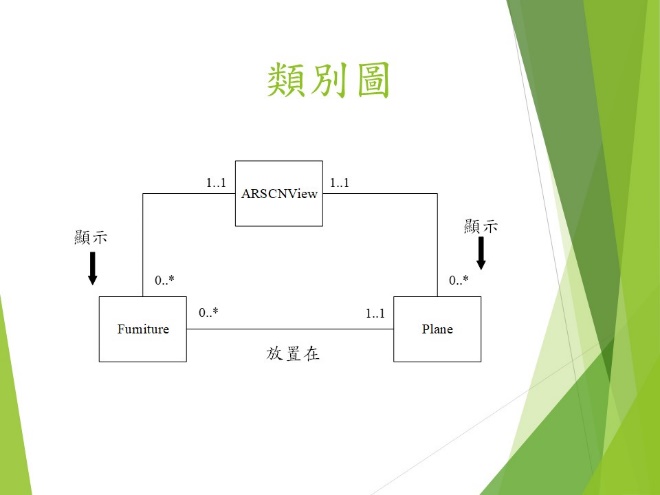
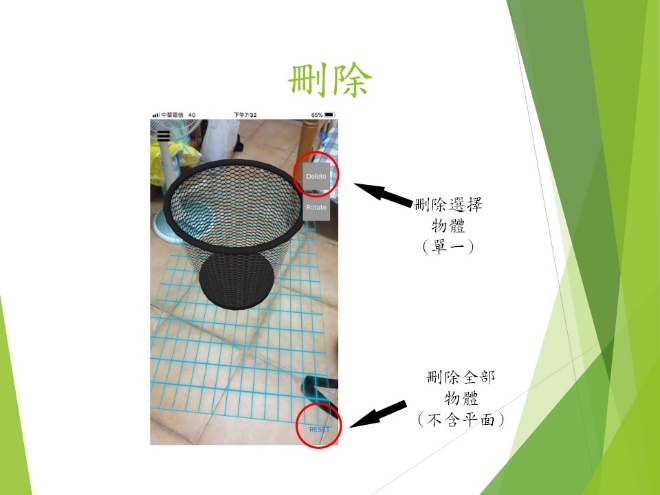
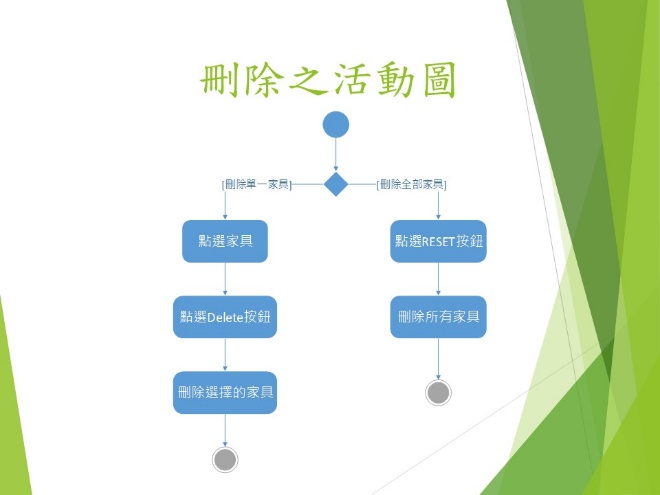
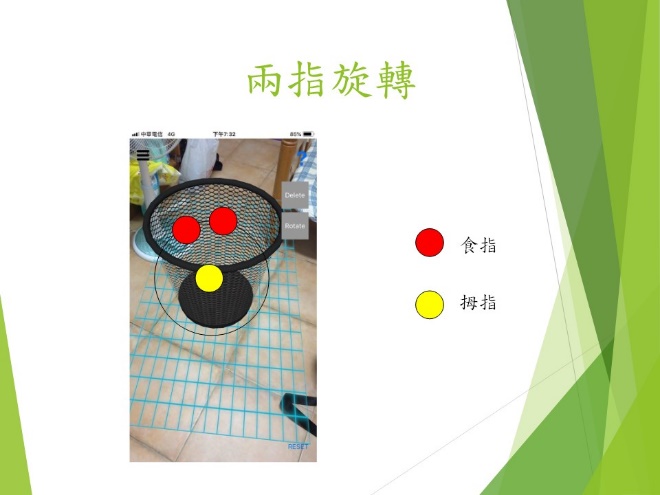
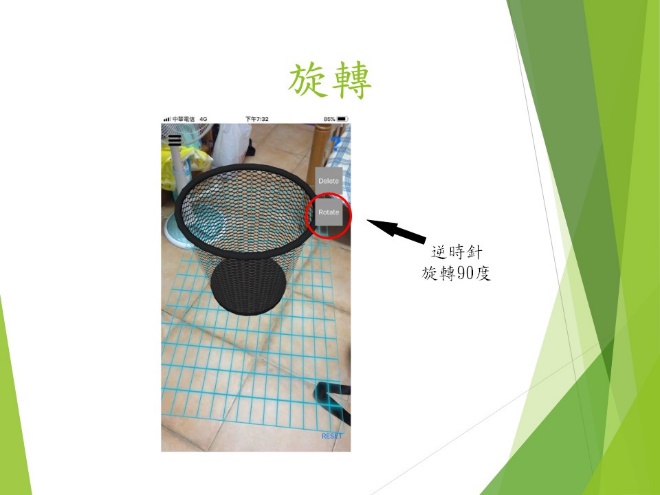
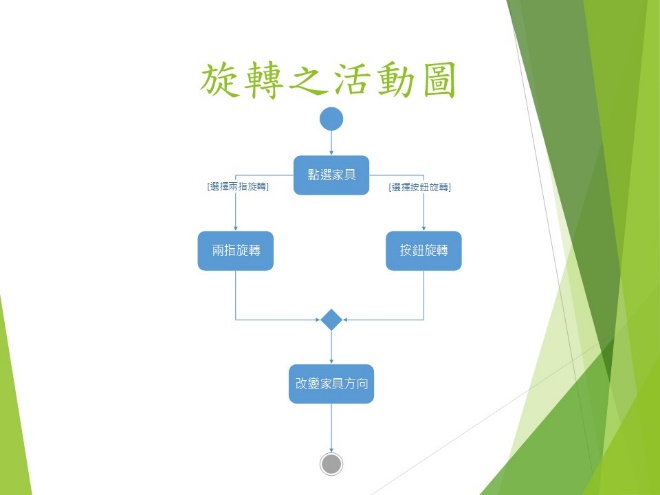
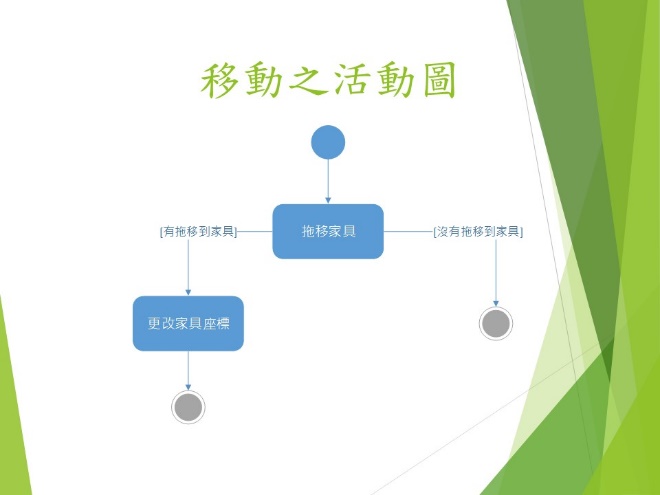
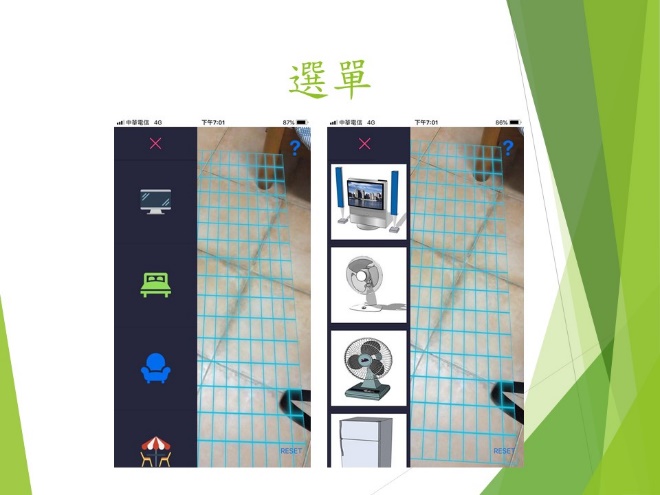
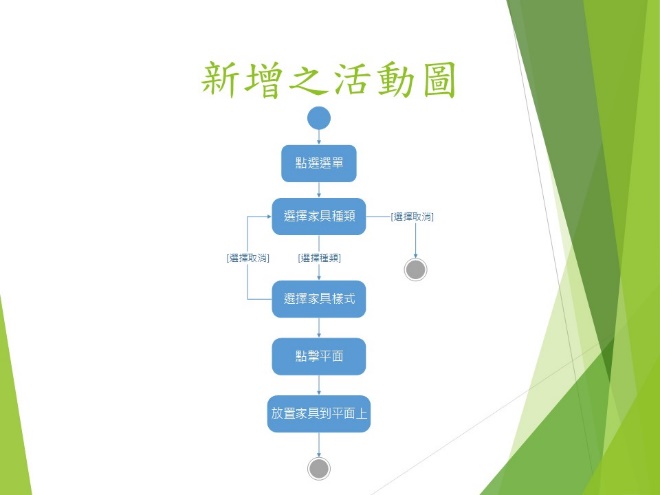
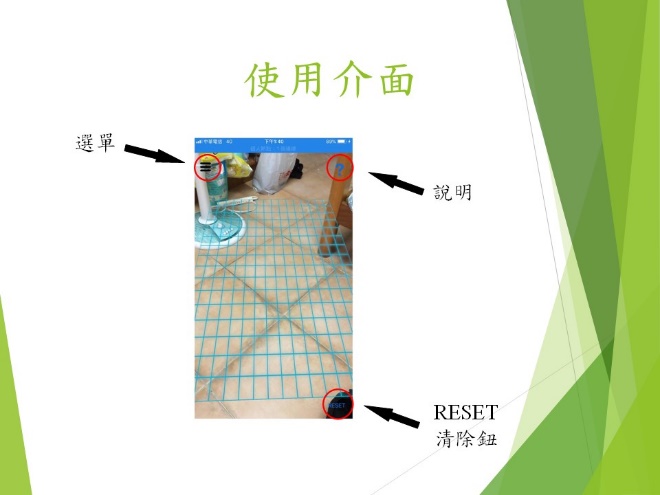
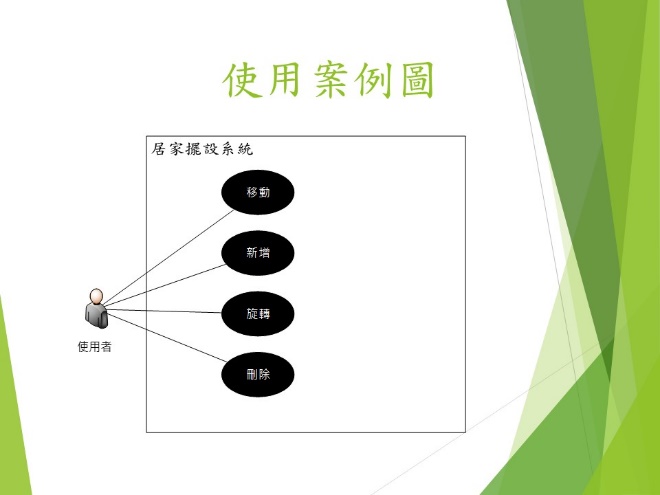
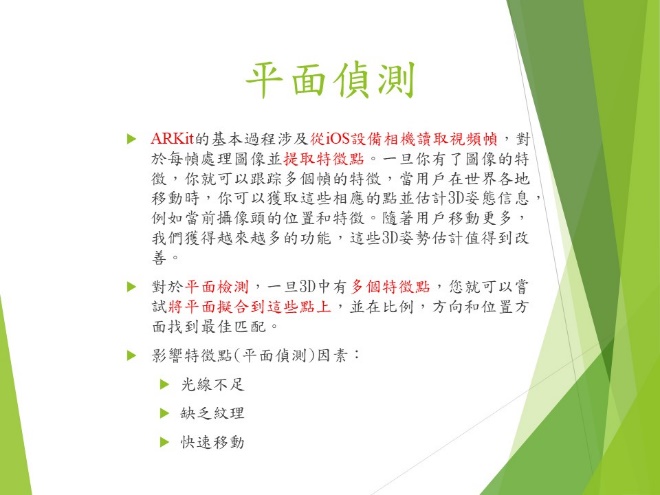
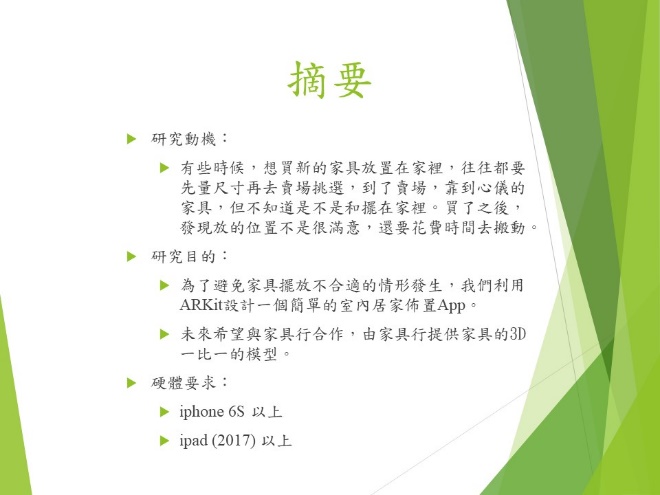
[4] Yalantis (2017,10 19) Side-Menu.iOS. Message posted to：

<https://github.com/Yalantis/Side-Menu.iOS>

[5] Jayven N(2017,Dec 17) ARKit Tutorial: Detecting Horizontal Planes and Adding 3D Objects with SceneKit. Message posted to：<https://www.appcoda.com/arkit-horizontal-plane/>

[6] 陳坤松，SketchUp 2017 室內設計繪圖實務，旗標 (2017)

**附錄**

****