

國立臺灣大學電機資訊學院生醫電子與資訊學研究所

碩士論文

Graduate Institute of Biomedical Electronics and Bioinformatics


College of Electrical Engineering and Computer Science

National Taiwan University

Master Thesis

應用 Kinect 感應器分析手指活動擷取系統之可行性

A Feasibility Analysis of  
a Finger Motion Capture System using Kinect



陳威誌

Wei-Chih Chen

指導教授：張璞曾 教授

Advisor: Fok-Ching Chong, Professor

中華民國 101 年 7 月

July, 2012

國立臺灣大學碩士學位論文  
口試委員會審定書

應用 Kinect 感應器分析手指活動擷取系統之可行性

A Feasibility Analysis of

a Finger Motion Capture System using Kinect

本論文係陳威誌 (R99945037) 在國立臺灣大學生醫電子與資訊學研究所完成之碩士學位論文，於民國 101 年 7 月 20 日承下列考試委員審查通過及口試及格，特此證明

口試委員：

陳威誌

(指導教授)

林新仁

陸哲駒

林育德

盧並裕

所長：

賴市龍

## 誌謝

本論文能夠順利完成，首先要感謝的是指導教授張璞曾教授，來自於張教授的細心指導和協助不僅止於課業上，老師的為人處事深深影響學生，使我更加堅定將來的方向，能夠被老師指導是學生我一輩子的榮幸，在此致上由衷的敬意與感恩。

感謝口試委員，林育德教授、盧並裕教授、陸哲駒教授、林耀仁教授，您們以豐富經驗提供指正和寶貴的意見，使得本篇論文能更加完善，也令我受益良多。在研究所這段日子裡，感謝王駿璋醫師、嚴天孝學長、劉思杰學長、曾友煌學長，學長們的幫助和指導，讓我在論文研究上更加成長及突破；總政、柏堂、宏瑞，和你們在一起經歷研究生活很精采、豐盛，讓我無時無刻都會回味無窮，非常謝謝你們。台北大學林伯星老師對於本論文的建議及指教，學生我從中學習到許多實驗以及完成論文的方法，在此致上最高感謝。

最後將能完成這篇論文的榮耀獻給我最摯愛的家人，您們的陪伴和關懷一直都是我最大的依靠，您們在生活上的照顧更讓我能夠全心全意地完成論文，謝謝。

## 摘要

本論文是建構在微軟公司推出的 Kinect 體感應器，利用此感應器 Light Coding 技術產生的深度資訊，來擷取真實空間中手指指尖的空間座標，並評估利用 Kinect 感應器在判斷手指活動上的可行性。在 Kinect 開發上，在此是利用 OpenNI 來進行感應器相關資訊的擷取；手指指尖偵測上，利用  $k$ -Curvature 演算法來找出指尖位置；在 Kinect 空間座標系的驗證上，本論文主要在 Z 軸深度資訊、X 軸與 Y 軸方向長度距離。手指指尖偵測部分則是在其偵測的穩定度分析，利用平均絕對值誤差率(Mean Absolute Percentage Error, MAPE)與均方誤差(Mean Squared Error, MSE)為評估工具。最後則是界定樣本假手手指彎曲量測的最大範圍。

在 Kinect 感應器的空間座標驗證上，本實驗的量測距離裡(50-130cm)，深度距離(Z 軸)誤差值會隨著距離的增加而成正比，但其深度平均誤差率均在 1%以內。水平及垂直距離的驗證上，本實驗發現在某些特定的深度距離內(80-110cm)，其誤差可以控制在可以接受的範圍內(5%)。手指指尖偵測演算法的穩定度分析上，除了中指指尖部分所量測之 X 座標以及 Y 座標的 MAPE 有超過 10 以外，其他均小於 10；雖然中指指尖部分較不穩定但其 MAPE 也都小於 50，所以都是合理的範圍內。在手指指尖偵測中，一般而言，每個手指的空間座標與平均值的平均誤差距離會隨著深度距離增加而提高。受限於光學上的限制，手指指尖偵測的模式在本實驗中所能偵測到的最大手指彎曲角度，約在 30-45 度左右。

根據以上的數據結果顯示，在本實驗中，要利用 Kinect 感應器來做為手指活動的擷取系統是可行的。雖然受限於硬體、以及光學原理，受試者手部位置及活動範圍必須被嚴格限制，但對於手部復健已有一定恢復程度的患者而言，藉由

Kinect 再加上適當的訓練模式，能使病患自行在居住地方進行密集且有趣的復健訓練，不需親自前往醫院，也可以大大提升手部功能。此外，對於臨床復健醫師而言，也可以藉由此系統間接得到患者復健及恢復程度，進而評估當下的復健模式或是修正接下來的訓練計劃。

關鍵字：Kinect 體感應器、OpenNI、OpenCV、Light Coding、深度影像、手指指尖偵測、 $k$ -Curvature、平均絕對值誤差率、均方誤差



## ABSTRACT

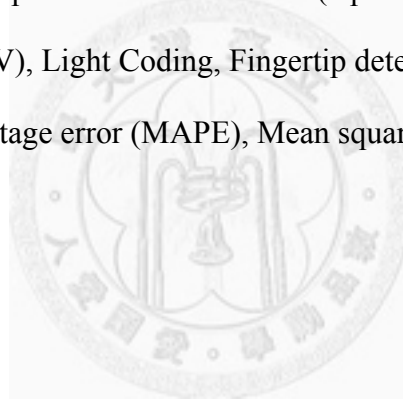
The system architecture of this thesis utilized concept modify from Microsoft Kinect sensor. We utilized depth information generated by Light Coding Technology in the Kinect to capture spatial coordinates of fingertip in real world space. Then, we assess the feasibility of using Kinect sensor on finger motion capture in real world. OpenNI is used to retrieve the needed information. In fingertip detection, we used k-curvature algorithm to find out the fingertip location. Validation of the Kinect space coordinates for Z-axis depth information, X-axis and Y-axis length distance were also done. In addition, the analysis of stability on fingertip detection algorithm was also presented. The mean absolute percentage error (MAPE) and mean squared error (MSE) are evaluation tools. This help to find out the maximum bending angle of sample finger.

In the verification experiment on the real space coordinates of the Kinect, we defined the depth measurement distance is from 50 cm to 130 cm. We found that the error value is proportional to the depth distance, but the average error rates are less than 1%. In the validation of horizontal and vertical distance, the error rate can be controlled within the acceptable rage (less than 5%) on the certain depth distance (80-110cm). In the stability analysis of fingertip detection algorithm, the MAPE value in tri-axial detection of each fingertip is mostly below 10. In general, average coordinates distance error is proportional to the depth distance within the measurement distance. Finger maximum bending angle that can be detected is about 30-45 degree, which is limited by the optical limitation.

According to the result in this experiment, it is feasible to use the Kinect as finger capture system. Although limited by the hardware, the subjects' position of hand and the

range of activities must be strictly limited. Nevertheless, for the patient with certain degree of recovery, using the Kinect within appropriate training mode can enable them to self-intensive training to live with interesting. Patient do not go to hospital can also improve hand function. In addition, clinical physician can currently assess the patients' condition by this system, and the clinical physician can also modify the training program as needed.

Key words: Kinect sensor, Open Natural Interaction (OpenNI), Open Source Computer Vision (OpenCV), Light Coding, Fingertip detection,  $k$ -Curvature, Mean absolute percentage error (MAPE), Mean squared error (MSE)



# 總目錄

誌謝 .....	i
摘要 .....	ii
ABSTRACT .....	iv
總目錄 .....	vi
圖目錄 .....	viii
表目錄 .....	xi
第一章 緒論 .....	1
1.1 前言 .....	1
1.2 相關文獻探討 .....	3
1.3 研究動機與方法 .....	7
1.4 論文架構 .....	8
第二章 原理介紹與方法 .....	9
2.1 微軟 Kinect 體感應器之介紹 .....	9
2.1.1 Kinect 體感應器之硬體描述 .....	9
2.1.2 Kinect 體感應器之深度資訊 .....	13
2.1.3 Kinect 體感應器之骨架追蹤系統 .....	16
2.2 Open Natural Interaction (OpenNI).....	22
2.2.1 OpenNI 架構 .....	22
2.2.2 OpenNI 之節點(Node).....	24
2.2.3 OpenNI 之能力(Capability).....	27
2.2.4 OpenNI 之座標系統 .....	29
2.3 NITE / OpenCV ( Open Source Computer Vision).....	31



2.3.1	NITE 之介紹 .....	31
2.3.2	OpenCV .....	33
第三章	系統設計與實驗方法 .....	34
3.1	整體系統架構分析 .....	34
3.2	使用者相關資訊追蹤 .....	36
3.3	手指指尖偵測流程 .....	42
第四章	驗證過程與實驗結果 .....	48
4.1	Kinect 三度空間座標系信賴度 .....	48
4.2	手指指尖偵測結果 .....	59
4.2.1	不同深度範圍偵測結果 .....	59
4.2.2	偵測穩定度分析 .....	62
4.3	手指指尖偵測之手指彎曲角度極限 .....	70
第五章	討論與未來展望 .....	72
5.1	討論 .....	72
5.2	未來展望 .....	75
參考文獻	.....	76

## 圖目錄

Figure 1.2.1 資料手套圖[1] .....	4
Figure 1.2.2 虛擬鋼琴模型[1] .....	4
Figure 1.2.3 手部活動輔助裝置[2] .....	5
Figure 1.2.4 VICON 手部動作偵測系統[3] .....	6
Figure 2.1.1 Kinect 體感應器 .....	10
Figure 2.1.2 Kinect 體感應器資料串流示意圖 .....	11
Figure 2.1.3 Kinect 體感應器架構圖 .....	11
Figure 2.1.4 Kinect IR projector 雷射空間投射示意圖 .....	14
Figure 2.1.5 空間中 Laser Speckle 示意圖 .....	15
Figure 2.1.6 Kinect 體感應器之深度產生器流程圖 .....	15
Figure 2.1.7 Pin Point Impression 示意圖[5] .....	17
Figure 2.1.8 深度影像資訊人體聚焦圖[5] .....	18
Figure 2.1.9 深度影像背景去除後示意圖[5] .....	18
Figure 2.1.10 人體部位彩色辨識區域圖 .....	19
Figure 2.1.11 模型匹配後的骨架系統[5] .....	20
Figure 2.1.12 PrimeSense 提供的人體骨架圖 .....	20
Figure 2.1.13 微軟 SDK 提供的人體骨架圖 .....	21
Figure 2.2.1 OpenNI 架構圖[7] .....	23
Figure 2.2.2 揮手動作[7] .....	25
Figure 2.2.3 使用者的手部位置[7] .....	25
Figure 2.2.4 三度空間場景中身體資訊[7] .....	26
Figure 2.2.5 深度影像經過視角轉換之後示意圖 .....	28

Figure 2.2.6 OpenNI 真實世界座標系統.....	30
Figure 2.3.1 NITE 方塊圖.....	32
Figure 3.1.1 系統架構圖.....	35
Figure 3.2.1 NITE 支援的人體關節圖[6].....	37
Figure 3.2.2 建立人體骨架流程圖[6].....	38
Figure 3.2.3 校正姿勢 Psi [6].....	40
Figure 3.3.1 手指指尖偵測流程圖.....	42
Figure 3.3.2 Kinect 體感應器深度影像.....	43
Figure 3.3.3 Kinect 手部位置示意圖.....	44
Figure 3.3.4 手部位置深度影像萃取圖.....	45
Figure 3.3.5 手部輪廓圖.....	45
Figure 3.3.6 手指指尖夾角示意圖.....	47
Figure 3.3.7 手指指尖示意圖.....	47
Figure 4.1.1 Kinect 體感應器真實世界視野範圍.....	50
Figure 4.1.2 Leica DISTO™ D3a BT 產品圖.....	50
Figure 4.1.3 深度驗證實驗側視圖.....	51
Figure 4.1.4 深度驗證實驗俯視圖.....	52
Figure 4.1.5 Kinect 在 OpenNI 下深度資料性質圖[6].....	53
Figure 4.1.6 雷射測距儀與 Kinect 感應器所量測到深度距離對應圖.....	53
Figure 4.1.7 Kinect 體感應器深度距離誤差值示意圖.....	54
Figure 4.1.8 體感應器單位距離之深度值平均誤差率.....	54
Figure 4.1.9 量測物.....	55
Figure 4.1.10 不同深度水平平均距離量測結果.....	56
Figure 4.1.11 不同深度水平距離量測誤差結果.....	56
Figure 4.1.12 不同深度垂直平均距離量測結果.....	57

Figure 4.1.13 不同深度垂直距離量測誤差結果.....	57
Figure 4.1.14 不同深度水平距離平均誤差率.....	58
Figure 4.1.15 不同深度垂直距離平均誤差率.....	58
Figure 4.2.1 假手模型.....	59
Figure 4.2.2 大拇指 X、Y、Z 座標在不同深度距離之 MAPE .....	64
Figure 4.2.3 食指 X、Y、Z 座標在不同深度距離之 MAPE .....	64
Figure 4.2.4 中指 X、Y、Z 座標在不同深度距離之 MAPE .....	65
Figure 4.2.5 無名指 X、Y、Z 座標在不同深度距離之 MAPE .....	65
Figure 4.2.6 小拇指 X、Y、Z 座標在不同深度距離之 MAPE .....	66
Figure 4.2.7 大拇指 X、Y、Z 座標在不同深度距離之 MSE .....	67
Figure 4.2.8 食指 X、Y、Z 座標在不同深度距離之 MSE .....	67
Figure 4.2.9 中指 X、Y、Z 座標在不同深度距離之 MSE .....	68
Figure 4.2.10 無名指 X、Y、Z 座標在不同深度距離之 MSE .....	68
Figure 4.2.11 無名指 X、Y、Z 座標在不同深度距離之 MSE .....	69
Figure 4.2.12 量測座標與平均值座標之平均誤差距離.....	69
Figure 4.3.1 手指彎曲角度測試假手示意圖.....	70

# 表目錄

Table 2.1.1 Kinect 體感應器規格.....	12
Table 3.2.1 Callback 分析表[6] .....	41
Table 4.1.1 Leica DISTOTM D3a BT 基本技術參數[12] .....	51
Table 4.2.1 不同深度間距指尖偵測圖 .....	60
Table 4.2.2 手指指尖偵測研究設備及開發環境 .....	61
Table 4.2.3 MAPE 評估標準表[26].....	63
Table 4.3.1 假手在不同深度間距下的最大彎曲角度( $^{\circ}$ )約略值.....	71



# 第一章 緒論

## 1.1 前言

人類全身上下，除了皮膚之外，與外界接觸最多、最頻繁的部分就是手，所以手也是受傷機會最高的部位。根據美國醫學會在具機能性殘障標準裡認定，喪失一支手臂等於喪失 60% 人體機能，若手部關節的斷裂則等於喪失手臂機能的 90% 或整個人體機能的 54%，超過人體活動功能的一半。手的重要性由此可知。然而造成手指殘疾或活動功能喪失的原因，往往不單單只是手部外傷、腦中風後所導致的手指活動功能喪失也不在少數。這些失去正常活動功能的手指不論是在接下來的工作或一般正常生活中都會帶來相對的不方便及不適。

骨骼肌肉疾病的運動治療有三個目的，增進關節活動度、增強肌肉力量及耐力、提升心肺耐力及全身適能。目前在臨床及實際家庭中，最常被拿來訓練手指靈活度的方式有：綁鞋帶、扣鈕扣、拿筆塗鴉、拿捏豆子等。以上之訓練模式早已行之有年，其主要目的不外乎用來提升手指關節的活度能力及手指肌肉的強度，再藉由現有的評估量表(巴氏量表、Jebsen Test of Hand Function 或 Wolf Motor Function Test)來評估其進步情形。

隨著科技的進步，微軟在西元 2010 年底推出了名為身體就是控制器的 Kinect 感應器，應用於 Xbox 360 主機的周邊設備。Kinect 感應器有三個鏡頭，中間是 RGB 彩色影像鏡頭，左右兩邊分別是紅外線發射器以及紅外線 CMOS 接收器；紅外線鏡頭主要目的是用來定義空間中的深度資訊，其中最重要的技術為 Light Coding

Technology。Light Coding 主要是利用紅外線的感應器接收由紅外線發射器在空間中所產生不同的 Laser Speckle，經由換算後得到空間中的深度資訊。得到相關空間中的深度資訊後，可以藉由座標轉換推導出相對應的水平、垂直座標，因此就可以建構出真實三度空間中的座標系。



## 1.2 相關文獻探討

一般而言，在復健的領域中，會需要清楚明白手指相對活動位置彎曲角度、手指施力大小…等等，通常都用於手部復健上；而會需要進行手部復健的人，主要是因為手部曾經受過外傷或是腦中風後所造成的手部功能喪失的病患。常見的手部評估、訓練、量測系統有，資料手套、外骨骼系統、VICON 量測系統…等等。

在最近的研究中，Thomas J. Lord [1]等人利用資料手套(如下圖 Figure 1.2.1 所示)來擷取相關的手指彎曲角度，在搭配虛擬實境的彈鋼琴遊戲(如下圖 Figure 1.2.2 所示)來建構出一套適合腦中風後病人手指復健的系統。他們利用資料手套中的十個可彎曲的感測器，每兩個感測器為一組只針對一根手指；上下各一個感測器緊靠著手指的關節，再利用微處理器將感測器變化的資訊無線傳出到電腦端，得到相對應手指關節的彎曲角度。進而設計出一套手指復健的訓練系統。



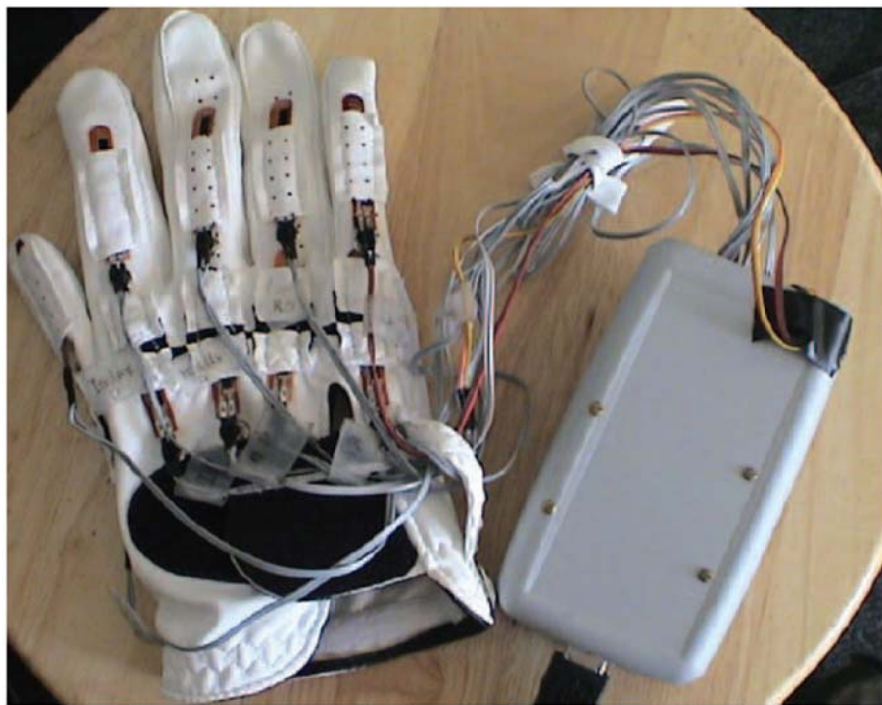


Figure 1.2.1 資料手套圖[1]



Figure 1.2.2 虛擬鋼琴模型[1]

外骨骼的輔助系統運用在手部復健上也是相當常見的，Satoshi Ito [2]等人設計了一套手部活動的輔助裝置(如下圖 Figure 1.2.3 所示)，它可以輔助手指活動功能喪失的病人，將其應用在機械式的手部復健系統上。其中他們可以就由這些外骨骼系統的活動位置來得知目前受試者手指的彎曲角度，以及將其資訊運用在適當的手部復健過程中。

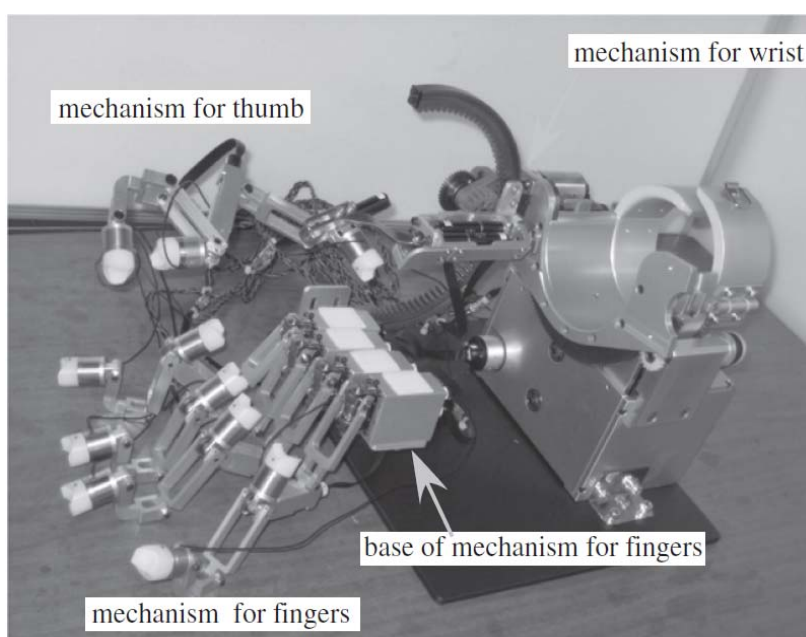


Figure 1.2.3 手部活動輔助裝置[2]

VICON 動作捕捉系統在復健領域中，常被拿來做為動作分析之用。一套完整的 VICON 系統通常包含有數個動作捕捉攝影機、相關軟體、套件、以及系統主機；目前 VICON 系統可以利用小型反光點貼於手指指尖的位置，再利用系統的攝影機來進行手指指尖各點在三度空間中位置的收集。接著，再利用收集到的資料，進行手部動作的分析及辨識。(如下圖 Figure 1.2.4 所示)



Figure 1.2.4 VICON 手部動作偵測系統[3]



### 1.3 研究動機與方法

在現行的手部評估、訓練、量測的系統中，大都需要受試者配戴相關的輔具，如：資料手套(如圖 Figure 1.2.1)、外骨骼系統(如圖 Figure 1.2.3)，或是建構一個量測系統就需要耗費相當程度的金錢及時間、系統反而更複雜，如：VICON(如圖 Figure 1.2.4)。「科技始終來自於人性」，隨著時代的進步，人類追求高科技或高品質生活的動力是從來沒有停歇過。因此，我們希望可以藉由現有的科學技術，建立一個簡單且低成本手指活動的擷取系統。

在本論文中，我們建構在微軟公司所提出來的 Kinect 體感應器，利用其中重要的技術(Light Coding Technology)，提出了一個有別於以往的手指活動的擷取系統。首先經由 OpenNI 來調控 Kinect 感應器，來得到其中的空間深度資訊，再加上手指指尖偵測的影像演算法取得空間中手指指尖的相對座標，並偵測每個手指的最大可偵測彎曲程度，因此希望可以將此建構在將來的手指靈活度的訓練系統上面。

在我們所提出來的手指活動擷取模式中，我們提供了一個最方便的偵測模式，有別於以往的資料手套或外骨骼系統，受試者不需要配帶任何的輔具或做任何的記號，讓受試者可以很方便的使用；而微軟 Kinect 感應器目前在市面上的售價約新台幣五千元以內，價格更是遠比 VICON 低許多，並且搭配現有的線上免費簡易開發平台，其困難度也大大的降低許多、使用性也提升不少。

## 1.4 論文架構

本論文共分為五章，第一章為緒論，前言、相關文獻探討部分主要是描述預先知識及近年來與本論文相關的研究，接著會介紹研究的動機、目的與方法。第二章為原理介紹，原理介紹可分為三大部分。第一部分主要在說明微軟公司所提出的 Kinect 體感應器的硬體架構與本論文會大量用到主要功能；第二部分會著重在應用程式介面(API; Application Program Interface) OpenNI 的詳細介紹；第三部分則是在相關的中介軟體 NITE，以及跨平台電腦視覺資料庫 OpenCV 的介紹。接下來的第三章與第四章為本論文最重要的兩個章節。第三章探討的是整個系統的架構、細部流程以及本論文的實作方法。第四章則是所有系統架構及相關資料的驗證結果與分析。最後第五章會就實驗所得到的結果提出完整的結論與說明，並檢討其實驗過程；探討將來的發展方向以及繼續努力的空間。

## 第二章 原理介紹與方法

### 2.1 微軟 Kinect 體感應器之介紹

#### 2.1.1 Kinect 體感應器之硬體描述

微軟公司在 2010 年的年底推出了一套令全世界玩家眼睛為之一亮的遊戲體感應器，Kinect for Xbox 360，簡稱為 Kinect。Kinect 體感應器主要是 Xbox360 遊戲的周邊產品，它可以讓玩家利用自然的身體語言以及話語來達到玩遊戲的目的，也就是微軟公司推出這套遊戲的主要賣點「身體就是控制器」！Kinect 體感應器是靠攝影鏡頭來捕捉使用者的動作，它一次可以擷取三種資訊，分別是彩色影像、3D 深度影像以及聲音的資訊。首先是 Kinect 體感應器的機身上有三個鏡頭(如下圖 Figure 2.1.1 所示)，中間的鏡頭是一般常見的 RGB 彩色攝影機、最左邊的鏡頭是紅外線發射器、最右邊的鏡頭則是紅外線 CMOS 攝影機，最左與最右的兩個鏡頭合稱為為 3D 深度感應器用來構成 3D 空間的深度資訊，Kinect 體感應器主要就是靠 3D 深度感應器來偵測使用者的位置及動作。

中間的 RGB 彩色攝影機可以拿來便是玩家的身分(靠人臉辨識或是身體特徵)、以及辨識基本的臉部表情，此外也能應用在擴增實境的遊戲上、或是視訊通話時；同時 Kinect 體感應器還搭配了自動追焦技術，它可以利用底座的馬達隨著對焦物體移動時也跟著一起左右轉動，左右可以旋轉各 28 度來追蹤玩家位置。Kinect 體感應器在本身的硬體裡面也內建了陣列式的麥克風系統，可以做為語音辨識之用，陣列式麥克風的好處是可以藉由多組麥克風同時收音，比對後消除掉雜音，等於

是大大地提高了信噪比，讓玩家的聲音可以更加清楚的傳遞。RGB 彩色攝影機、3D 深度感應器加上一整排的麥克風陣列，因此一台 Kinect 體感應器可以產生三種不同的資料串流:彩色影像、深度影像以及聲音的串流(如下圖 Figure 2.1.2 所示)。

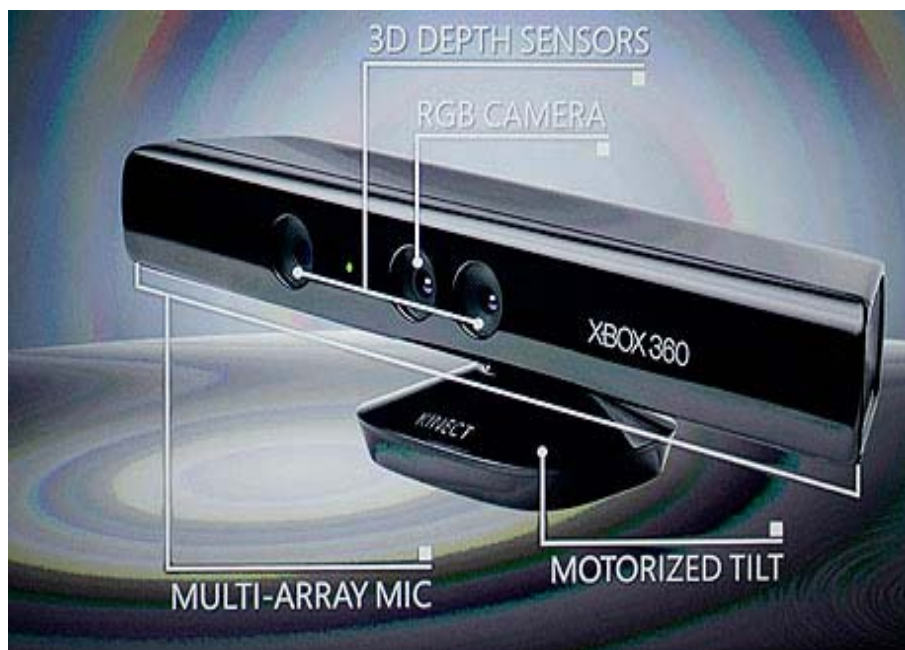


Figure 2.1.1 Kinect 體感應器

(圖片來源: Games Blog)

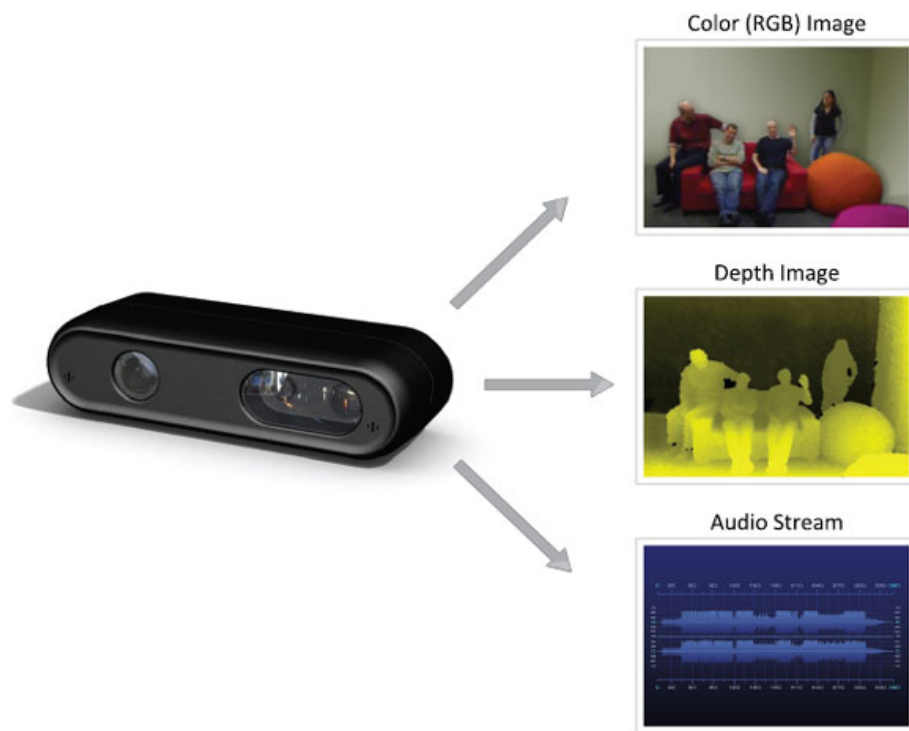


Figure 2.1.2 Kinect 體感應器資料串流示意圖

(圖片來源: PrimeSense)

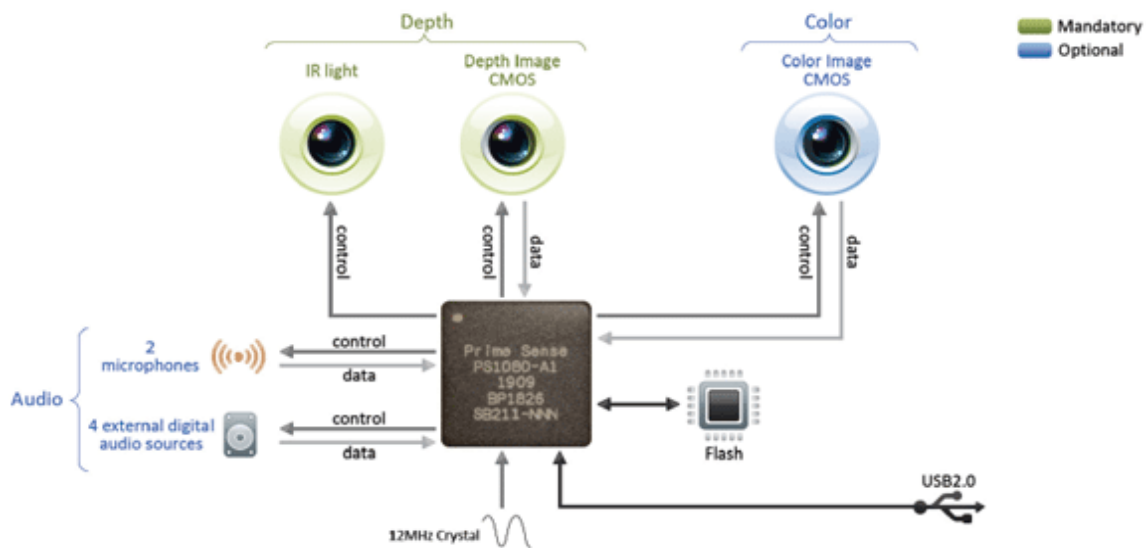


Figure 2.1.3 Kinect 體感應器架構圖

(圖片來源: PrimeSense)



Table 2.1.1 Kinect 體感應器規格

Kinect 體感應器規格	
視野角度	水平視野: 57 度 垂直視野: 43 度 實體傾斜範圍: 正負 27 度
底座馬達旋轉	左右各 28 度
每秒畫格	30FPS
深度影像解析度	640*480
彩色影像解析度	640*480



## 2.1.2 Kinect 體感應器之深度資訊

微軟公司在 2010 年的四月，對外正式公布與一家以色列公司 PrimeSense 合作。PrimeSense 這家公司是提供動作感測系統的公司，他們擁有體感偵測裝置 PrimeSensor，以及感測晶片 PS1080(如圖 Figure 2.1.3)。PrimeSense 這家公司用來產生 3D 影像的技術並不是 TOF(Time of Flight) 的技術，而是 Light Coding 的技術。在此稍微簡單介紹一下 TOF 技術，它主要是去計算光線飛行的時間，一開始會先有一個裝置發出脈衝光，並且在發射處去接受目標物的反射光，藉由量測時間差算出目標物的距離[4]。

Light Coding 的技術理論是利用連續光(近紅外線)對量測空間進行編碼，經由感測器讀取編碼的光線後，交由晶片運算進行解碼後，得到一張具有景深的圖像。Kinect 體感應器就是以紅外線發出人眼無法看到的 class 1 雷射光，雷射光穿過濾波器透過鏡頭前的擴散片均勻地投射在量測的空間中(如下圖 Figure 2.1.4)，當這些雷射光照射到物體、或是穿透玻璃時，會形成隨機的反射斑點(如下圖 Figure 2.1.5)，稱之為雷射散斑(Laser Speckle)。雷射散斑具有高度的隨機性，也就是說在不同物體表面上、不同的曲面以及不同的距離(深度)時，會產生不同的雷射散斑圖案；明確地來說，在 Kinect 體感應器鏡頭前的空間中任何兩處的雷射散斑都是不一樣的，這也就等於是將整個空間的每一個地方都加上了記號，所以任何物體進入該空間、或移動時，就可以確實的紀錄物體的位置[4]。

3D 深度感測器中兩個鏡頭是專門來產生深度影像用的，透過 Kinect 體感應器上最左邊的紅外線投影鏡頭，將雷射光投射在空間中，再利用最右邊的紅外線 CMOS 攝影機不停地來擷取空間中所有雷射散斑圖案，並將其轉換成三度空間的

深度影像(如下圖 Figure 2.1.6 所示)。

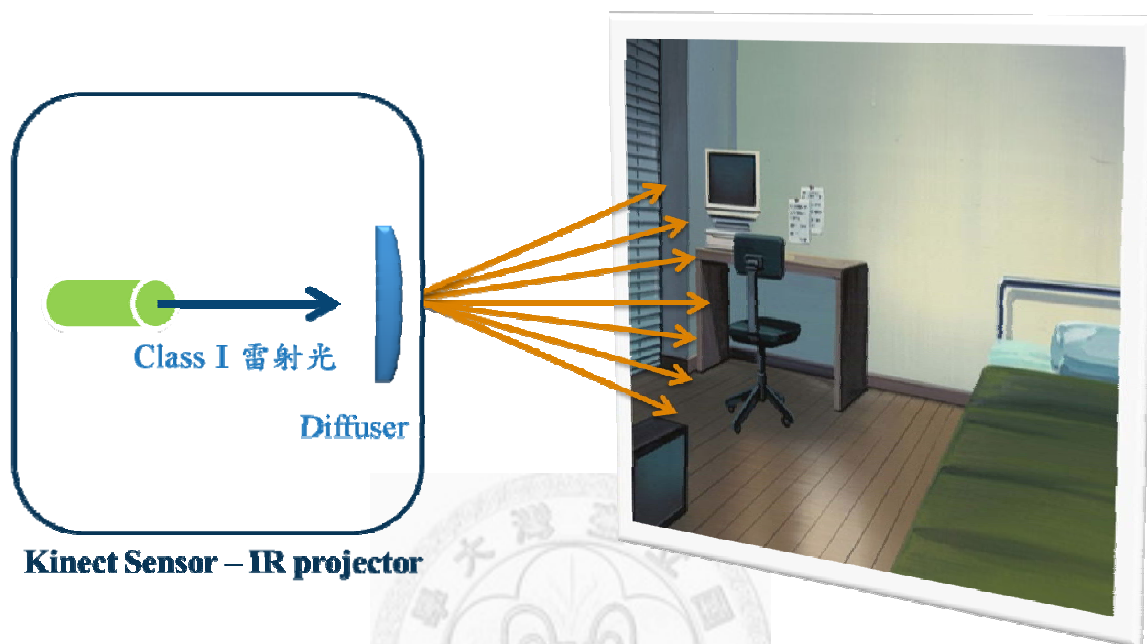


Figure 2.1.4 Kinect IR projector 雷射空間投射示意圖

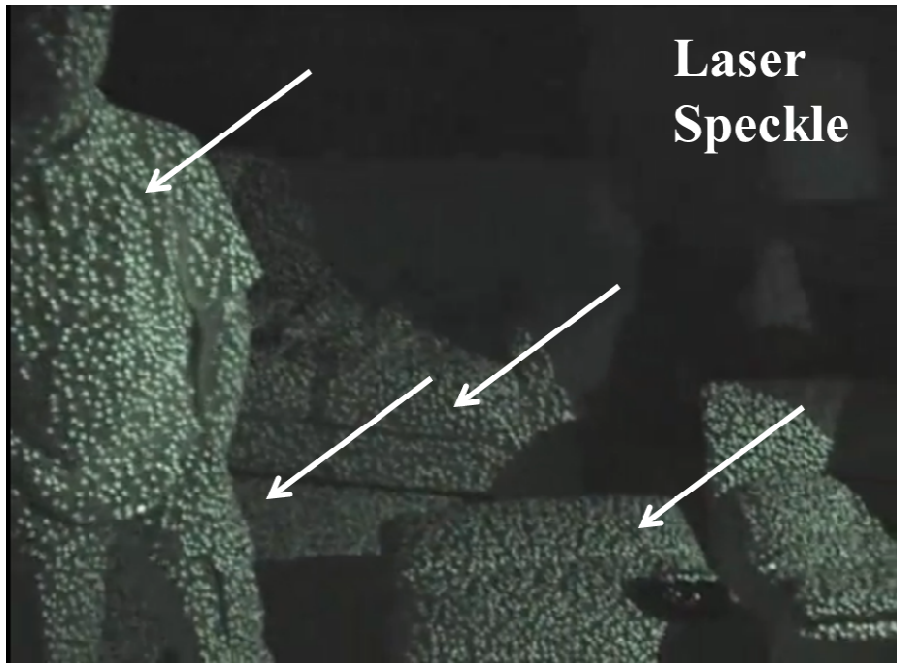


Figure 2.1.5 空間中 Laser Speckle 示意圖

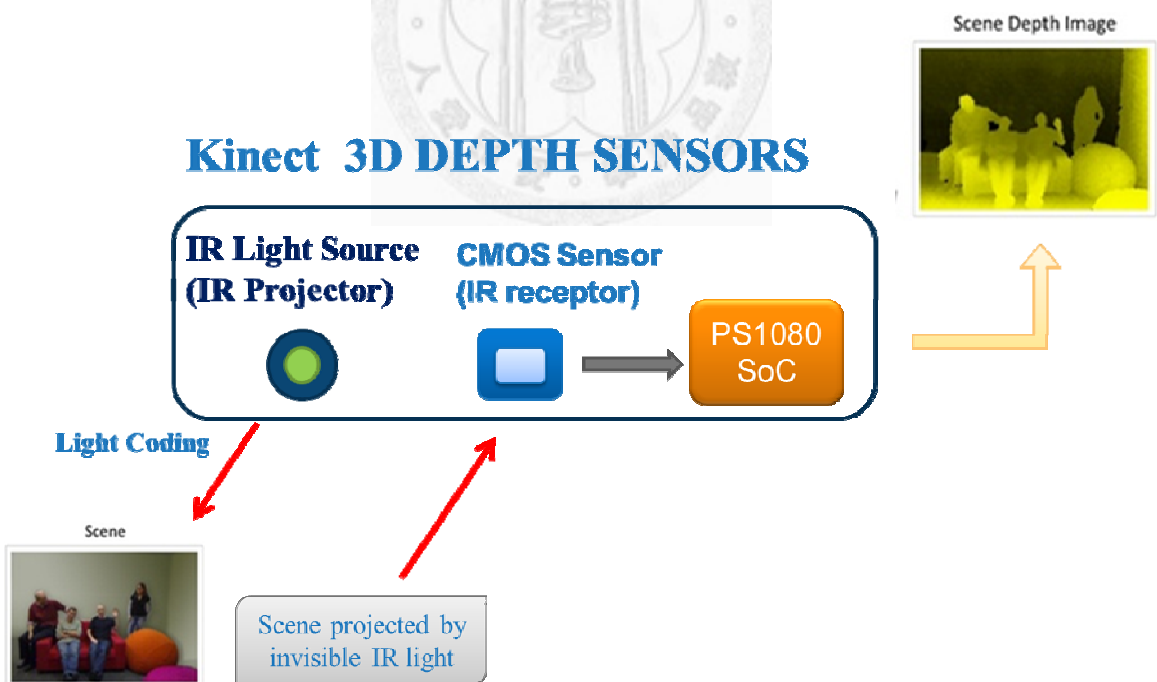


Figure 2.1.6 Kinect 體感應器之深度產生器流程圖

### 2.1.3 Kinect 體感應器之骨架追蹤系統

「你就是控制器」，這是微軟公司所提出的 Kinect 體感應器中最著名也最令人印象深刻的廣告詞。Kinect 體感應器開創了一種新的遊戲互動模式，而如此嶄新的娛樂動方式是如何產生的？首先，是在上一章節有說明過的三度空間深度資訊，利用這些 3D 深度資訊接下來才有另一個關鍵技術：骨架追蹤系統。骨架追蹤系統可以得到空間中玩家身體相關部位的位置資訊，藉由這些位置資訊進而衍生出一系列的互動模式。在本章節會為 Kinect 體感應器的骨架追蹤系統詳加敘述說明。

Kinect 體感應器中骨架追蹤處理流程的核心是一個無論周圍環境的光照條件是如何，Kinect 體感應器都可以藉由紅外線 CMOS 感測器來接收空間中的資訊。該感測器通過黑白光譜的方式來感知環境：純黑或純白代表無窮遠或無窮近。黑白間的灰色地帶對應物體到傳感器的物理距離。它收集視野範圍內的每一點，並形成一幅代表周圍環境的景深圖像。Kinect 體感應器以 30FPS 的速度來產生深度影像串流，可以立即 3D 地呈現出當下的周圍環境。就如同 pin point impression 玩具可能更容易理解這一技術，將手或身體的某一部分按壓在這種玩具上，就可以產生你身體某一部位的簡單 3D 模型(如下圖 Figure 2.1.7 所示)[5]。



Figure 2.1.7 Pin Point Impression 示意圖[5]

Kinect 體感應器需要做的下個步驟是尋找圖像中較可能是人體的移動物體(如下圖 Figure 2.1.8 所示),就像人眼下意識地聚焦在移動物體上那樣。接著 Kinect 體感應器會對 3D 深度影像圖進行像素級評估,來辨別人體的不同部位。同時,這一過程必須以優化的預處理來縮短響應時間。Kinect 體感應器採用分割策略來將人體從背景環境中濾除出來,就如同從一堆複雜的訊號中萃取出有用信號。在這一個步驟中,每個被追蹤的玩家在 3D 深度影像中創建了所謂的分割遮罩,這是一種將背景物體剔除後的景深圖像(如下圖 Figure 2.1.9 所示)。在後面的處理流程中僅僅傳送分割遮罩的部分,以減輕 Kinect 體感應器的計算量[5]。

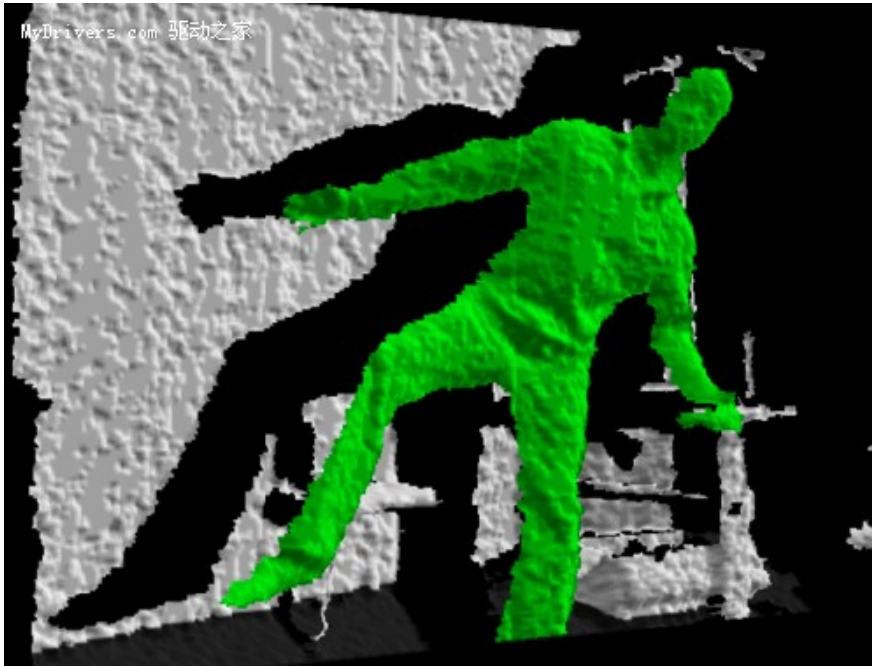


Figure 2.1.8 深度影像資訊人體聚焦圖[5]



Figure 2.1.9 深度影像背景去除後示意圖[5]

最後也是最重要的一步，接著是微軟公司內部自行開發出來的辨識機制；因為一開始的 3D 深度影像資料串流的產生是利用 PrimeSense 這家公司所開發出來的技術。真正的重點在這裡發生了，經由分割萃取後所得到的玩家影像，他的每一個像素都會被傳送到一個人體部位辨識的機器學習系統(Machine Learning System)中。在這個系統裡會去決定某些特定的像素是屬於身體部位的可能性(如下圖 Figure 2.1.10 所示)。比如，一個像素有 90%的機率是屬於右手，30%的機率是屬於胸部。接下來再將所有可能性輸入接下來的處理流程，藉由模型匹配的方式來生成最後的骨架系統(如下圖 Figure 2.1.11 所示) [5]。



Figure 2.1.10 人體部位彩色辨識區域圖

(圖取自: Kinect Body Tracking Reaps Renown by Rob Knies)



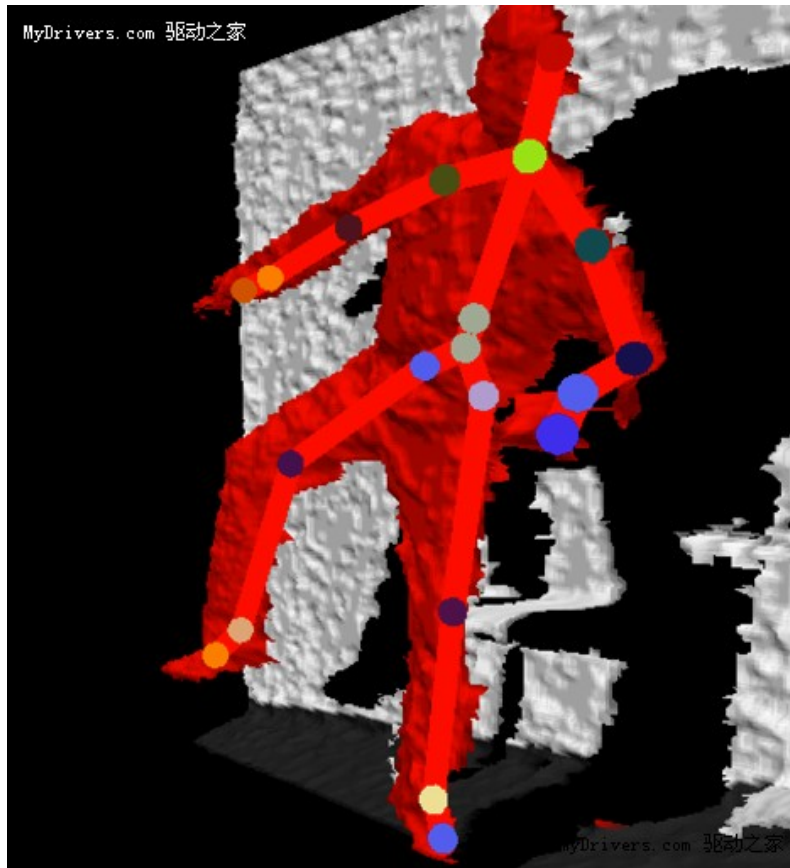


Figure 2.1.11 模型匹配後的骨架系統[5]

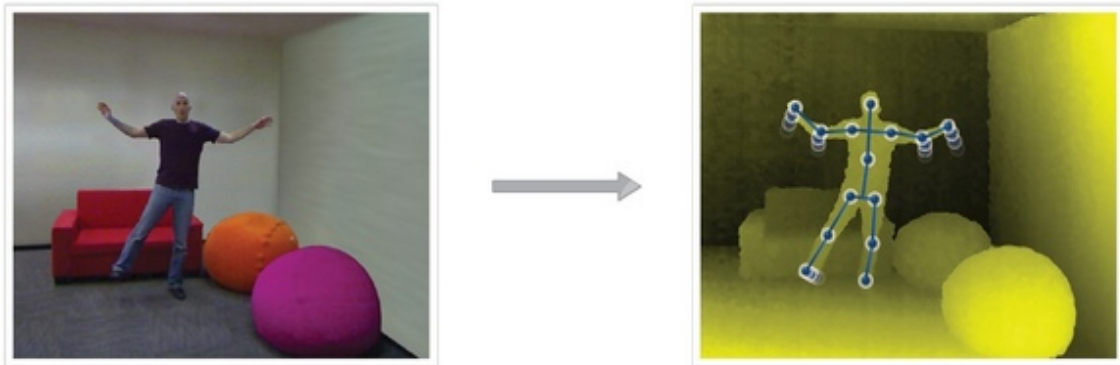


Figure 2.1.12 PrimeSense 提供的人體骨架圖

(圖片來源: PrimeSense)

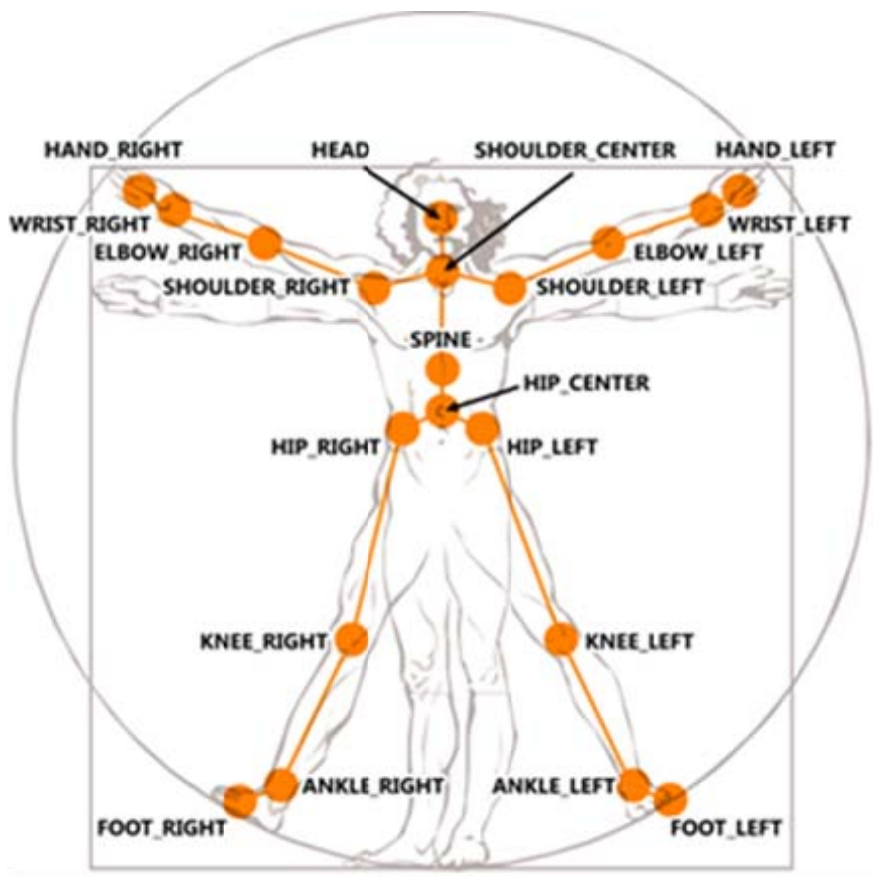


Figure 2.1.13 微軟 SDK 提供的人體骨架圖

(圖取自：微軟 Kinect 開發教學官網)

## 2.2 Open Natural Interaction (OpenNI)

### 2.2.1 OpenNI 架構

OpenNI 可以翻譯為「開放式自然操作」，它定義包含了語音、手勢、身體動作...等等，基本上就是比較直覺、操作者身上不需要其他特殊裝置的操作方式。OpenNI 本身則是定義了撰寫自然操作程式所需要的 API(Application Program Interface)，提供一個多語言（主要是 C/C++）、跨平台的 Framework；藉此提供了一個標準的介面，讓程式開發者要使用視覺、聲音相關感應器，以及對於這些資料、分析的中介軟體（Middleware）時，可以更為方便[6][7]。

OpenNI 的架構圖(如下圖 Figure 2.2.1 所示)基本上分為三層，最上層是應用程式 (Application)，也就是我們這些程式開發者自己要撰寫的部分；最下方的一層則是硬體的部分，目前 OpenNI 支援的硬體，包含了：3D Sensor、RGB Camera、IR Camera、Audio Device 這四類。不過以目前來說，會用 OpenNI 的人，主要應該就是用 Kinect 體感應器，而如果能有對應的驅動程式的話，其他類似的裝置，應該也是有機會可以讓 OpenNI 來存取的。而中間這層就是 OpenNI 的部分，它除了負責和硬體的溝通外，也在自身內部預留了加上中介軟體(Middleware)的空間，可以用來做手勢辨識、或是追蹤之類的處理。OpenNI 目前在 Middleware 的部分，定義了下面四種元件：

1. 全身分析 (Full Body Analysis)：由感應器取得的資料，產生身體的相關資訊，例如關節、相對位置與角度、質心等等。
2. 手部分析 (Hand Point Analysis)：追蹤手的位置。
3. 手勢偵測 (Gesture Detection)：辨識預先定義好的手勢，例如揮手。

4. 場景分析 (Scene Analyzer)：分析場景內的資訊，例如：分離前景和背景、地板的座標軸、辨識場景內的不同物體。

目前 PrimeSense 也已經提供了一套 NITE 當作最主要的 Middleware、提供上面所列的功能；而如果有更進階的需求的話，也可以自己寫一套相容於 OpenNI 的 Middleware 來使用[6][7]。

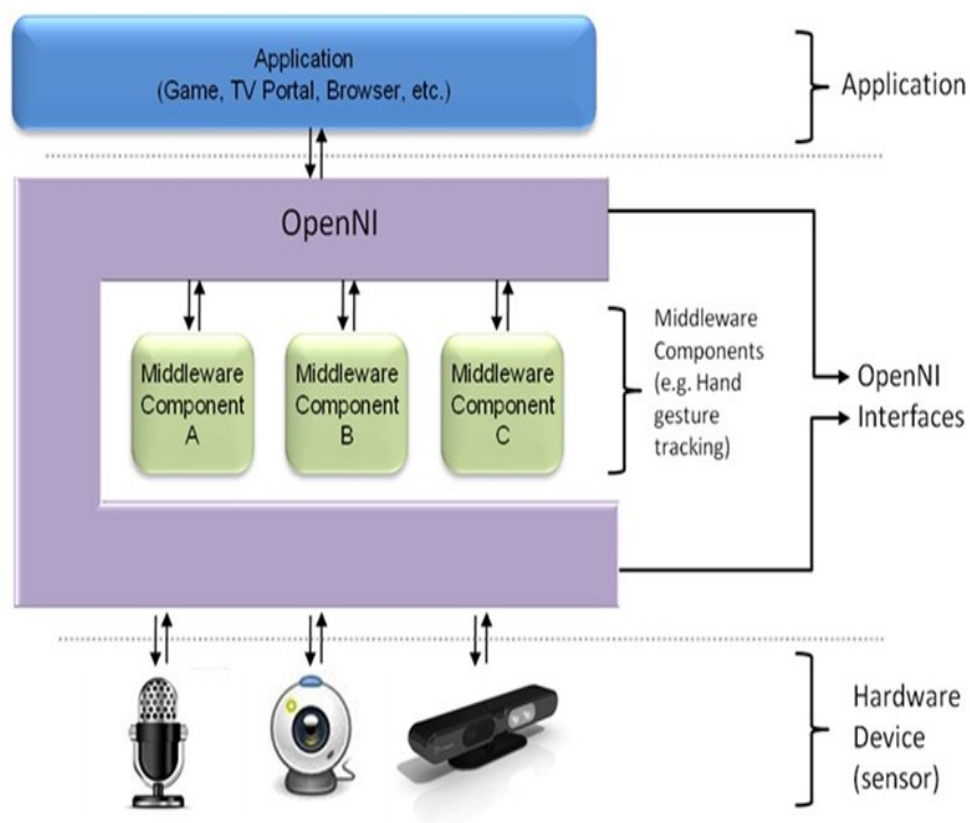


Figure 2.2.1 OpenNI 架構圖[7]

## 2.2.2 OpenNI 之節點(Node)

在 OpenNI 裡，定義了所謂的「Production Node」來代表內部的基本單元，包括了硬體部分的感應器，以及 OpenNI 所提供的功能；這些 Production node 分為下面三大類：

### 一、感應器相關 (Sensor Related) Production Nodes

- 裝置 (Device)：代表實體裝置的節點，用來做這些設備的設定。
- 深度產生器 (Depth Generator)：產生深度資訊圖 (Depth Map) 的節點。
- 影像產生器 (Image Generator)：產生彩色影像圖 (Colored Image Maps) 的節點。
- 紅外線影像產生器 (IR Generator)：產生紅外線影像圖 (IR Image Maps) 的節點。
- 聲音產生器 (Audio Generator)：產生聲音串流 (Audio Stream) 的節點。

### 二、中介軟體相關 (Middleware Related) Production Nodes

- 手勢通知產生器 (Gestures Alert Generator)：當辨識到特定的手勢時，呼叫應用程式的 Callback (如下圖 Figure 2.2.2 所示)。
- 場景分析器 (Scene Analyzer)：分析場景，包括分離前景與背景、識別場景內的不同物體、偵測地板。主要的輸出會是標記過的深度資訊圖 (Labeled Depth map)。
- 手部位置產生器 (Hand Point Generator)：支援手部偵測與追蹤，當偵測到手、或追蹤手的位置時，會產生一個通知訊息(如下圖 Figure 2.2.3 所示)。

- 使用者產生器 (User Generator)：產生一個 3D 場景中完整、或局部的身體資訊 (如下圖 Figure 2.2.4 所示)。

### 三、錄製／撥放

- 錄製器 (Recorder)：用來記錄資料用的。
- 撥放器 (Player)：讀取記錄下來的資料，並撥放出來。
- 編解碼器 (Codec)：用來壓縮、解壓縮紀錄資料。

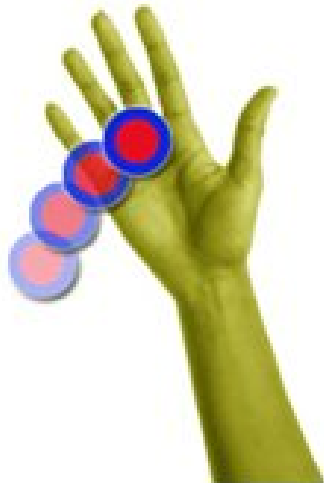


Figure 2.2.2 揮手動作[7]

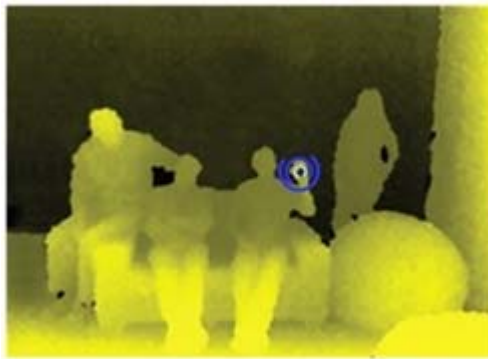


Figure 2.2.3 使用者的手部位置[7]

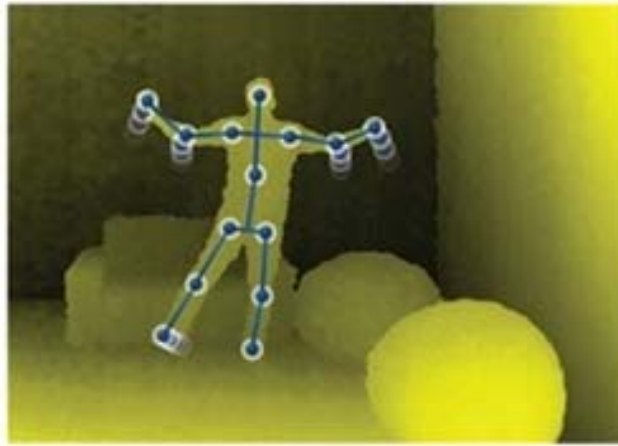


Figure 2.2.4 三度空間場景中身體資訊[7]

上面這些 Production Node 基本上都是會由不同的模組各自實作的。像是以感應器相關的部分，就是會由 OpenNI 相容的裝置提供，以目前來說，主要就是 Kinect 體感應器的驅動程式會提供這深度、影像產生的功能。而中介軟體相關的部分，則是由不同的 Middleware 各自提供；目前的來源應該只有 NITE[6]。

在層級上，感應器相關的 Production Node 算是最底層的，直接存取設備的資料，所以應用程式可以直接使用這類 Production Node。而中介軟體相關的 Production Node 由於是靠感應器的資料來做處理的，所以他們的層級則比感應器的高一層、必須要在有感應器相關的 Production Node 的情況下才可以使用[6]。

而在有了上面的這些 Production Node 後，就可以透過組合這些節點來建立所謂的「Production Chain」，並以此進行資料的處理流程。比如說要產生使用者的資料的話，就是會透過「使用者產生器」(User Generator) 去存取更低層的「深度產生器」(Depth Generator)；而這樣的一個節點序列，就是所謂的「Production Chain」[6]。

### 2.2.3 OpenNI 之能力(Capability)

OpenNI 的「Capability」機制是用來增強中介軟體和硬體裝置的彈性的；這些不同的能力都是非必要性的，各家廠商所提供的不同的中介軟體和裝置，可以自己決定要提供那些能力。而 OpenNI 則是負責定義好一些可以使用的 Capability，讓程式開發者可以快速地找到符合自己需求的中介軟體或裝置。目前的 OpenNI 所支援的 Capability 則如下：

- 替換視角 (Alternative View)：讓各類型的 Map Generator (深度、影像、紅外線) 可以轉換到別的視角(如下圖 Figure 2.2.5 所示)，就好像攝影機在別的位置一樣。這個功能可以快速地替不同的感應器產生的內容作對位。
- 裁切 (Cropping)：讓各類型的 Map Generator (深度、影像、紅外線) 輸出結果可以被裁切、降低解析度；例如：VGA 可以裁切成 QVGA。這對效能的增進很有用。
- 畫面同步 (Frame Sync)：讓兩個感應器產生結果同步化，藉此可以同步取得不同感應器的資料。
- 鏡像 (Mirror)：把產生的結果鏡像左右顛倒。
- 姿勢偵測 (Pose Detection)：使用者產生器 (User Generator) 可以偵測出使用者特定的姿勢。
- 骨架 (Skeleton)：讓使用者產生器 (User Generator) 可以產生使用者的骨架資料。包含骨架關節的位置、並包含追蹤骨架位置和使用者的校正的能力。
- 使用者位置 (User Position)：讓深度產生器 (Depth Generator) 可以針對指定的場景區域、最佳化輸出的深度影像。



- 錯誤狀態 (Error State)：讓節點可以回報他本身的錯誤狀態。
- Lock Aware: 讓節點可以在 context 範圍外被鎖定。

基本上，這些 Capability 許多都是只針對特定的 Production Node 才會有的，例如: Skeleton 就只有 User 這類的 Node 才会有[6]。



Figure 2.2.5 深度影像經過視角轉換之後示意圖

## 2.2.4 OpenNI 之座標系統

OpenNI 可以將 Kinect 體感應器中的 3D 深度影像以及彩色(RGB)影像輸出在電腦螢幕上；當這些影像被完整地呈現在電腦螢幕時，由電腦螢幕影像呈現的座標系勢必會與 Kinect 體感應器前的 3D 空間座標系大不相同；在本章節會為這兩種不一樣的座標系敘述說明。OpenNI 所定義的座標系可分為以下兩種：[6]

### 一、投影座標系統(Projective Coordinate System)

在此處投影所代表的意義，實際上就是電腦三度空間圖學裡面，將一個三度空間場景中的所有東西，以一個指定好的方向與方法投射在一個設定好的平面上，產生一張平面(2D)的影像。在 OpenNI 的投影座標系統裡，和一般的電腦影像得座標系是相同的，影像的最左上角是(0,0)原點，向右為正的 X 軸，向下為正的 Y 軸。由於投影座標系統裡的 X、Y 座標的單位是像素(pixel)，Kinect 體感應器所產生的 Z 軸座標單位是公厘(mm)，所以單位是不一致的，並不適合直接拿投影座標系統中的座標直接做三度空間的處理。

### 二、真實世界座標系統(Real World Coordinate System)

在取得投影座標系的資料後，需要透過 OpenNI 裡的深度產生器(Depth Generator)所提供的 ConvertProjectiveToRealWorld()這個函式，將投影座標系上的座標點轉換成真實世界三度空間座標系的座標點。OpenNI 的真實座標系統是以 Kinect 體感應器為原點(0,0)的座標系統；Kinect 體感應器的正前方為正的 Z 軸，右方為正的 X 軸、往上則是正的 Y 軸。所以實際上這個座標系統所採用的是左手座標系統的形式(如下圖 Figure 2.2.6 所示)。而在真實世界裡的座標系統，X 軸、Y 軸、Z 軸的單位都是一致的(mm)。

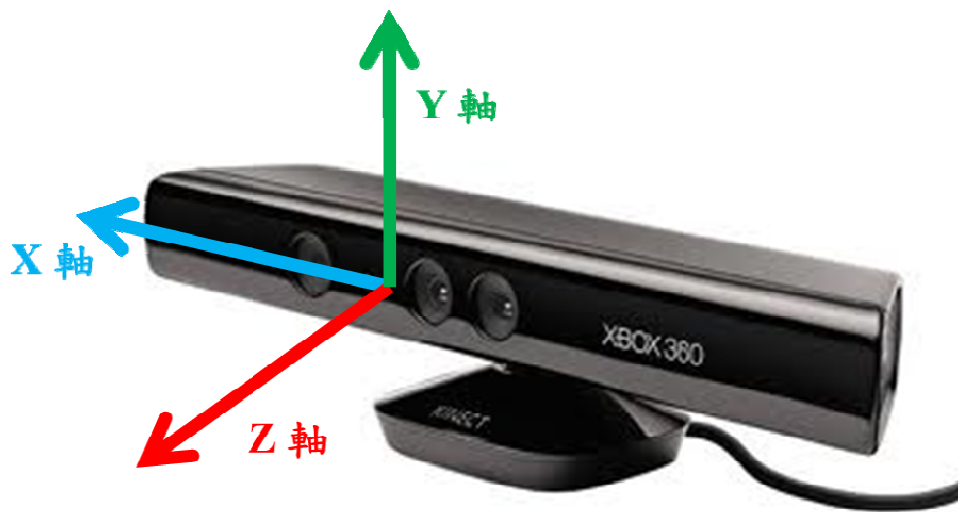


Figure 2.2.6 OpenNI 真實世界座標系統



## 2.3 NITE / OpenCV ( Open Source Computer Vision)

### 2.3.1 NITE 之介紹

NITE 類似一個工具箱，它可以讓應用端基於使用者的手部活動去建立一套處理流程，所謂的手部活動就像是一些手勢的追蹤或是提供一個手部的點位置資訊。NITE 基本上是建構在 OpenNI 之上的，PrimeSense 這家公司為 OpenNI 的模組提供一些完整的軟體工具，可以把這些軟體工具想像成「NITE 演算法」。在不久的將來，這些軟體模組或許可以直接應用在實質的硬體上，就像是手部追蹤的模組(手部產生器)可以支援提供單一手部位的點資訊[8]。

NITE 在 OpenNI 的架構裡包含了以下幾個層級(由下至上) [8]:

- OpenNI 模組區塊: NITE 在 OpenNI 的模組區塊裡提供了包括手勢產生器以及手部追蹤產生器。此外，NITE 還有提供了像場景分析的模組和使用者產生器中的骨架追蹤能力。
- OpenNI 本身的基礎區塊。
- 調節控制區塊: 可以接受一連串的点，並安排使之進入最適當的 NITE 控制模組裡。
- 控制區塊: 每個控制端接受一連串的点並將其辨識為有意義的活動給控制端。這些控制端會從應用端呼叫可以改變當下在調節控制區塊正在動作的控制端，因此就可以定義了應用端資料的流向。

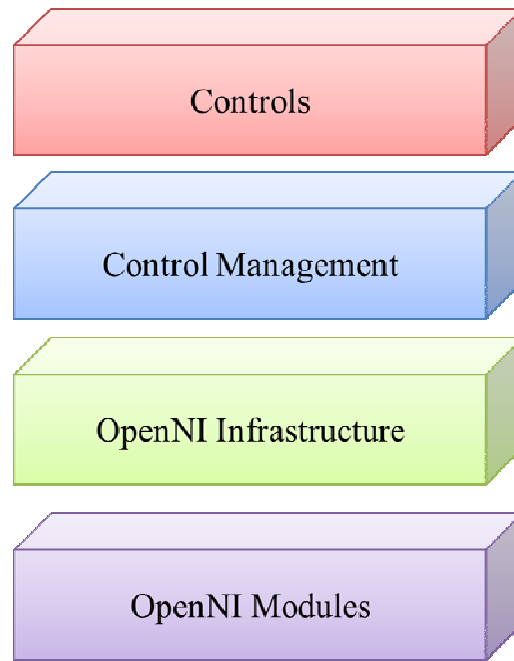


Figure 2.3.1 NITE 方塊圖



### 2.3.2 OpenCV

OpenCV 全名為 Open Source Computer Vision Library，是由 Intel 公司所授權發行，為 Open Source 圖形演算法的函式庫(Image Process Library)可以製做圖片、視訊、矩陣運算、統計、圖論、資料儲存的相關 C 語言程式設計，如:影像處理、圖形識別、電腦圖學等，也可以整合不同圖檔格式的矩陣運算，應用在靜態圖片(BMP、JPG、TIF、PNG)，動態 Webcam 的影像處理。這個函式庫是由 C 和 C++程式語言所寫而成，可以在 Linus、Windows 和 Mac 的作業平台上來實施。在本論文裡，主要是利用 OpenCV 的函式庫來做為手指指尖偵測的工具[9]。

OpenCV 的組成為許多圖形處理的資料結構及演算法所疊合而成，因此需要一些基本的資料結構基礎，比如說 struct 怎麼使用，如何 release 資料結構的空間等基本操作，會使用到 linked list 的算少數。功能可比擬 Matlab，處理速度比 Matlab 快上許多倍，更可以整合 C 語言相關函式庫，做更多元的功能應用。

## 第三章 系統設計與實驗方法

### 3.1 整體系統架構分析

本實驗的系統架構設計，著重在以下幾個步驟，電腦主機端與 Kinect 體感應器的溝通與連結，如何有效正確的從 Kinect 體感應器得到所需的影像資料，使其影像可以做為接下來的影像處理之用。因此，主要可以分為兩個部分，第一部分為利用 OpenNI 將 Kinect 體感應器裡的資料產生器打開與連結，將所需資料串流輸出處理與呈現；另一個部分則是利用影像處理分析的演算法得到影像中的重要資訊，並將其處理與呈現。

在整個實驗步驟的流程中，可以用圖 Figure 3.1.1 來表示。對於 Kinect 影感應器與 OpenNI 而言，每個進入鏡頭的人都把它定義為新的使用者(New User)。這整個系統架構圖有可以把它細分為兩大部分，第一部分(如圖 Figure 3.1.1：綠色區塊)為使用者追蹤以及姿勢的校正，這一部分在下個章節會詳加說明；第二部分(如圖 Figure 3.1.1：紫色區塊)則是利用 Kinect 體感應器的深度攝影機所得到的深度影像來進行影像處理，手指指尖的偵測。第一部分主要是用來擷取身體每個大關節的位置資訊，某些特定位置資訊可以在第二部分手指指尖偵測的區塊內被使用到；還有一些重要的關節空間座標資訊也都會在後面的決定手指彎曲角度時被參考使用。

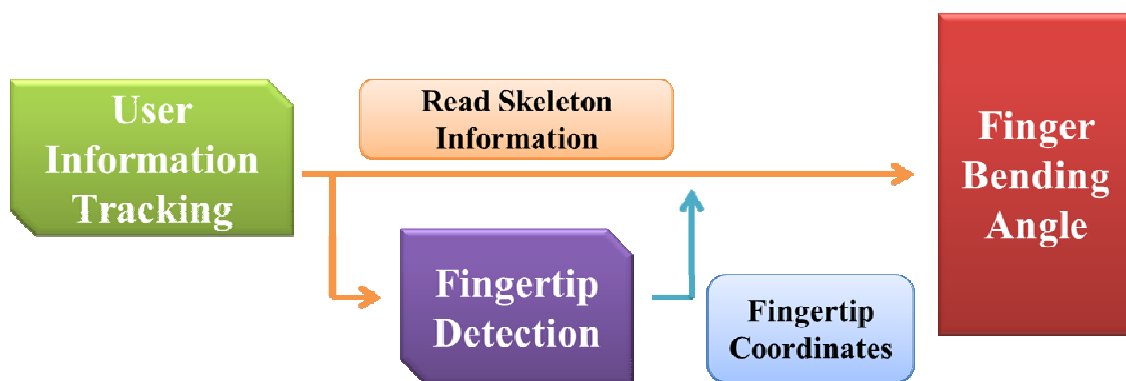


Figure 3.1.1 系統架構圖





## 3.2 使用者相關資訊追蹤

本章節所要探討的為系統架構圖的第一部分(如圖 Figure 3.1.1: 綠色區塊), 在這個方框裡的流程的最終產物是為了要得到使用者身體的所有關節資訊; 在 OpenNI 的定義裡基本上是可以得到所有關節的位置(Position)、方向(Orientation) 兩種資訊, 在本實驗中最為重要的是關節的位置資訊(Position)。而目前 OpenNI 裡, 總共定義了 24 個關節(XnSkeletonJoint), 分別為:

XN\_SKEL\_HEAD,  
XN\_SKEL\_NECK,  
XN\_SKEL\_TORSO,  
XN\_SKEL\_WAIST  
XN\_SKEL\_LEFT\_COLLAR, XN\_SKEL\_RIGHT\_COLLAR,  
XN\_SKEL\_LEFT\_SHOULDER, XN\_SKEL\_RIGHT\_SHOULDER,  
XN\_SKEL\_LEFT\_ELBOW, XN\_SKEL\_RIGHT\_ELBOW,  
XN\_SKEL\_LEFT\_WRIST, XN\_SKEL\_RIGHT\_WRIST,  
XN\_SKEL\_LEFT\_HAND, XN\_SKEL\_RIGHT\_HAND,  
XN\_SKEL\_LEFT\_FINGERTIP, XN\_SKEL\_RIGHT\_FINGERTIP  
XN\_SKEL\_LEFT\_HIP, XN\_SKEL\_RIGHT\_HIP,  
XN\_SKEL\_LEFT\_KNEE, XN\_SKEL\_RIGHT\_KNEE,  
XN\_SKEL\_LEFT\_ANKLE, XN\_SKEL\_RIGHT\_ANKLE,  
XN\_SKEL\_LEFT\_FOOT, XN\_SKEL\_RIGHT\_FOOT

OpenNI 雖然定義了這麼多的關節點, 但實際上我們在透過 NITE 這個中介軟體(Middleware)來分析骨架時, 其實並沒有辦法將所有的關節資訊一一地擷取出來,

因為目前 NITE 對於一些較細部的身體點資訊還沒辦法明確的定義出來。我們可以透過 `xn::SkeletonCapability` 裡的成員函式 `EnumerateActiveJoints()` 來取得所有可以用的關節資訊，將所有可以用的關節資訊整理可以把它畫成一張圖(如圖 Figure 3.2.1 所示)[6]。

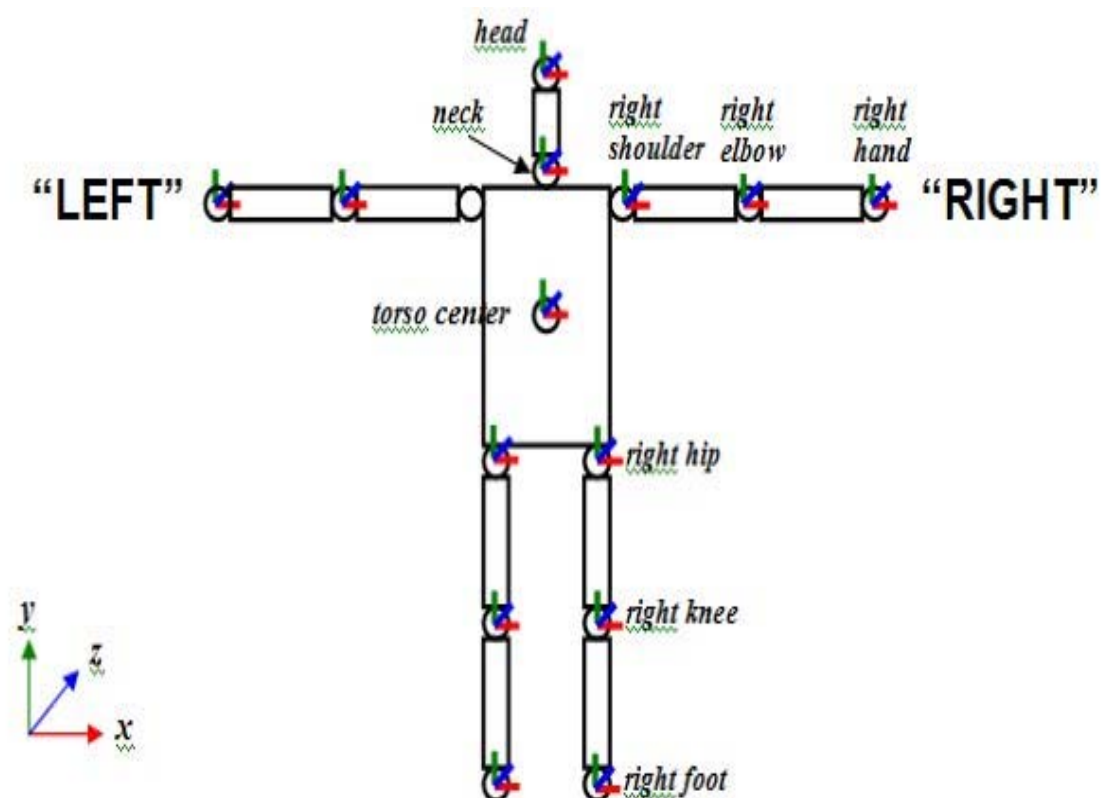


Figure 3.2.1 NITE 支援的人體關節圖[6]

分析完 OpenNI 裡 NITE 所能提供的關節資訊後，接下來就要利用 OpenNI 來建立一套人體骨架系統來得到最後所需要的所有關節資訊。首先，要先用到的是中介軟體相關(Middleware Related)節點(Production Node)裡的使用者產生器(User Generator)、也就是 `xn::UserGenerator`；建立人體骨架系統的資料來源是 Kinect 體

感應器 3D 深度感測器產生的深度影像串流(Depth Image Stream)，所以必須也建立一個感應器相關(Sensor Related)節點裡的深度產生器(Depth Generator)。使用者產生器(User Generator)的使用方法必須要透過 callback function 的機制，來做事件(event)的處理。在 OpenNI 裡面，要為一個節點(production node)附加上 callback function 時，基本上就必須透過各種節點提供、名稱為 RegisterXXXCallbacks()的成員函式，來註冊該節點的 callback function；如果要用到使用者產生器的節點(xn::UserGenerator)時，就要用到 RegisterUserCallbacks()這個函式；若是要使用到姿勢偵測則會用到 RegisterToPoseCallbacks()這個函式[6][7]。

實驗流程系統架構圖中的第一部分，也就是讀取空間中人體的關節資訊，要得到這些資訊最重要的一件事是先建立起真實世界三度空間中人體的骨架圖。建立人體骨架圖的流程如圖 Figure 3.2.2 所示。

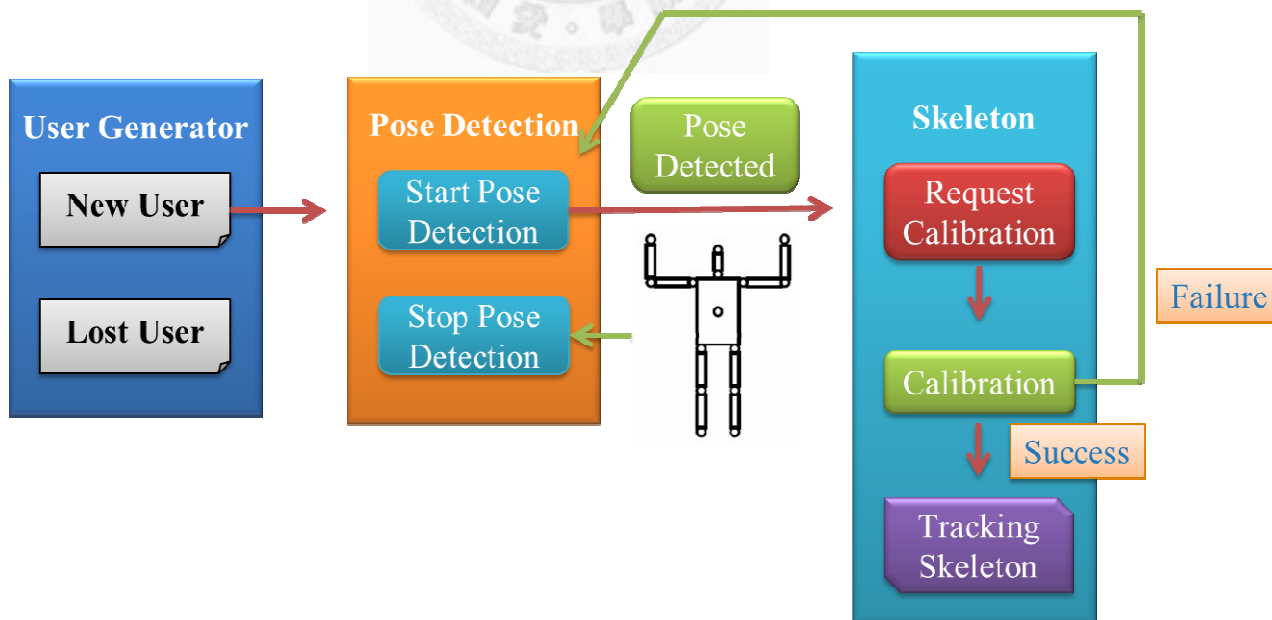


Figure 3.2.2 建立人體骨架流程圖[6]

在建立人體骨架的流程圖中(如圖 Figure 3.2.2 所示)，最左邊的方塊是代表的是使用者產生器(User Generator；`xn::UserGenerator`)，在這方塊裡的「New User」和「Lost User」分別代表了兩個事件的 callback function；而這兩個函示分別會在「畫面內偵測到新的使用者」、「使用者離開可偵測範圍一段時間」時被呼叫。

中間的方塊則是姿勢偵測(Pose Detection)這個能力(capability；`xn::PoseDetectionCapability`)。當 User Generator 偵測到有新的使用者、去呼叫「New User」這個 callback function 時，「New User」的程式會去呼叫 Pose Detection 的「Start Pose Detection」，讓 Pose Detection 開始偵測 NITE 所預先定義好的校正用姿勢：「Psi」（如圖 Figure 3.2.3 所示）。在呼叫「Start Pose Detection」前，Pose Detection 是不會進行姿勢偵測的動作的。

當 Pose Detection 偵測到使用者擺出「Psi」這個校正姿勢後，它就會去呼叫自己的「Pose Detected」這個 callback function、然後進行下一階段的動作；在這個過程中，「Pose Detected」會去做兩件事，一個是去呼叫自己的「Stop Pose Detection」來停止繼續偵測使用者的動作、另一個則是去呼叫 Skeleton 這個 capability (`xn::SkeletonCapability`) 的「Request Calibration」函式，要求 Skeleton 開始進行人體骨架的校正、分析。

在 `xn::SkeletonCapability` 的「Request Calibration」被呼叫後，Skeleton 就會開始進行骨架的校正及分析。當開始進行骨架校正的時候，Skeleton 會去呼叫「Calibration Start」這個 callback function 開始進行人體骨架的校正了；當人體骨架校正結束後，則是會去呼叫「Calibration End」這個 callback function。

不過，當「Calibration End」被呼叫的時候，只代表骨架的校正、辨識的階段

工作結束了，並不代表骨架辨識一定成功，也有可能是會失敗的。如果成功的話，就是要進入下一個階段、呼叫 `xn::SkeletonCapability` 的 `StartTracking()` 函式，讓系統開始去追蹤校正成功的骨架資料；而如果失敗的話，則是要再讓 Pose Detection 重新偵測校正姿勢，等到有偵測到校正姿勢後，再進行下一次的骨架校正。

在骨架校正成功且開始進行追蹤骨架後，之後只要呼叫 `xn::SkeletonCapability` 用來讀取關節資料的函式，(例如: `GetSkeletonJoint()`)，就可以讀取到最新的關節相關資訊，並建立完整的人體骨架資料[6]。

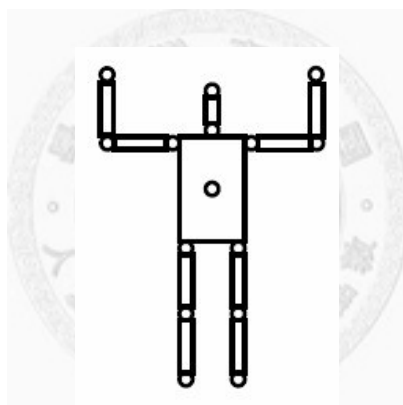


Figure 3.2.3 校正姿勢 Psi [6]

Table 3.2.1 Callback 分析表[6]

節點 (Production Node) 、 能力 (Capability)	Callback Function	型態 (Type)
User Generator	New User	void (XN_CALLBACK_TYPE* UserHandler)( UserGenerator& generator, XnUserID user, void* pCookie )
	Lost User	
Pose Detection Capability	Pose Detected	void (XN_CALLBACK_TYPE* PoseDetection)( PoseDetectionCapability& pose, const XnChar* strPose, XnUserID user, void* pCookie )
Skeleton Capability	Calibration Start	void (XN_CALLBACK_TYPE* CalibrationStart)( SkeletonCapability& skeleton, XnUserID user, void* pCookie )
	Calibration End	void (XN_CALLBACK_TYPE* CalibrationEnd)( SkeletonCapability& skeleton, XnUserID user, XnBool bSuccess, void* pCookie )



### 3.3 手指指尖偵測流程

基於 Kinect 體感應器所提供的三度空間深度影像圖，來偵測空間中手指指尖的位置，其流程主要可以分為以下幾個步驟(如下圖 Figure 3.3.1 所示)。

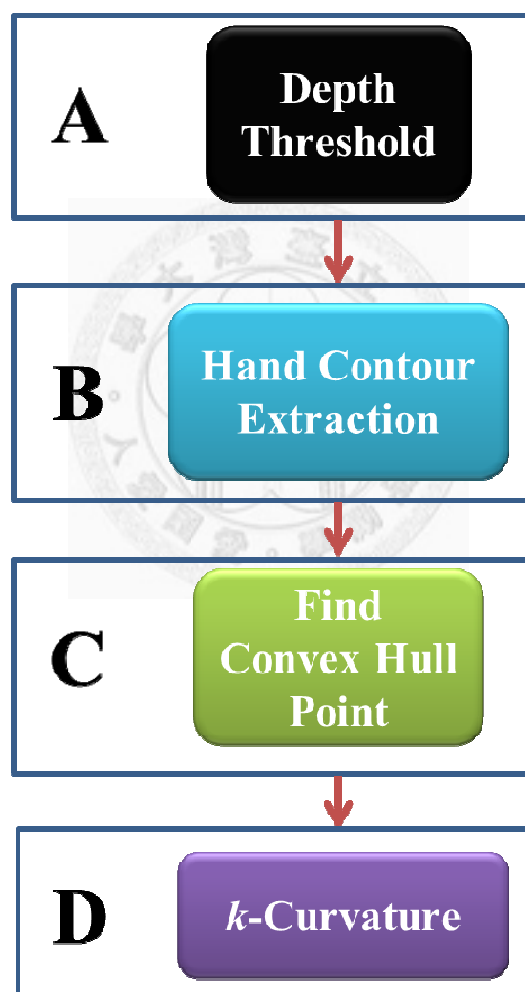


Figure 3.3.1 手指指尖偵測流程圖

A. 從 Kinect 體感應器中得到深度影像資訊，並萃取出手部深度資訊:

在先前的章節已經有提過，我們可以在同一個時間從 Kinect 體感應器裡去擷取深度影像以及彩色影像；彩色影像跟一般的彩色攝影機得到的類似，而深度影像(如下圖 Figure 3.3.2 所示)的產生最主要是利用 Light Coding 的技術來取得；深度影像資訊是拿來重建三度空間立體資訊的重要來源。目前深度影像輸出每個畫素的灰階值可以到達 16 bits，有 65536 個灰階，不過實際上並沒有用到那麼多。



Figure 3.3.2 Kinect 體感應器深度影像

在此步驟，某些特定的深度資訊會被萃取分類出來。為了能夠有效地得到手掌部位的有效資訊，以手部在真實空間座標系的 Z 軸座標為基準，分別選取正負(遠近)10 公分的距離，並且以投影座標系手部位置(x,y)為圓心，120



畫素距離為半徑範圍內的資訊，當成接下來要找出手部邊界圖案的范围。

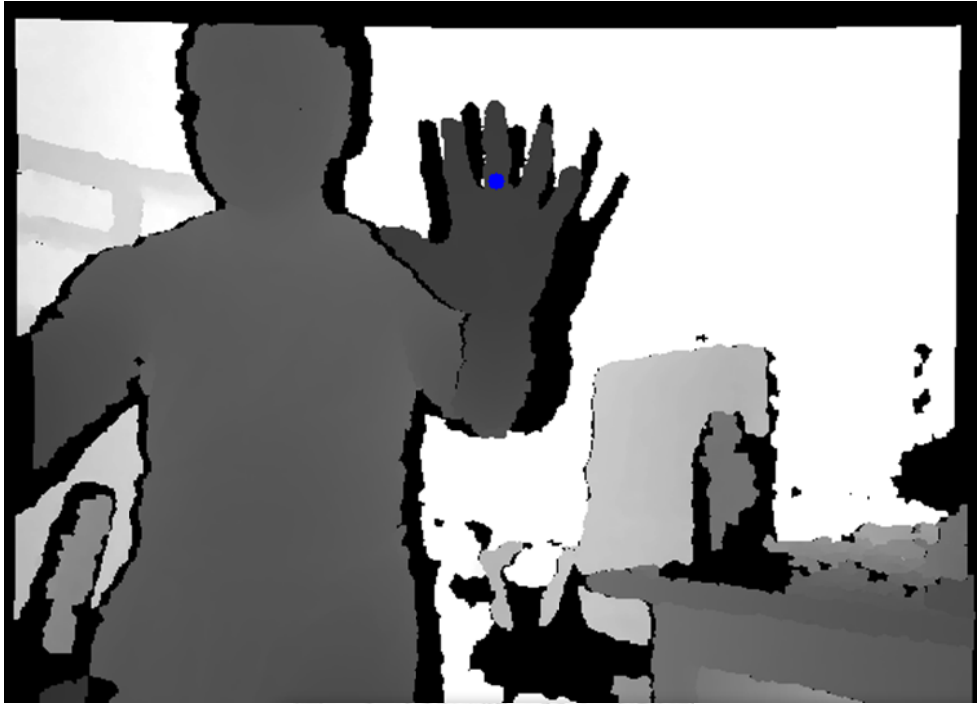


Figure 3.3.3 Kinect 手部位置示意圖

B. 手部輪廓的萃取:

上個步驟裡，已經將預做處理的深度影像資訊範圍分割出來，分割出來的資訊為單通道二維陣列的二元影像資訊，大小為 640\*480、8-bit 的結構。再利用以上資訊來得到該範圍內的手部輪廓(如下圖 Figure 3.3.5 所示)。

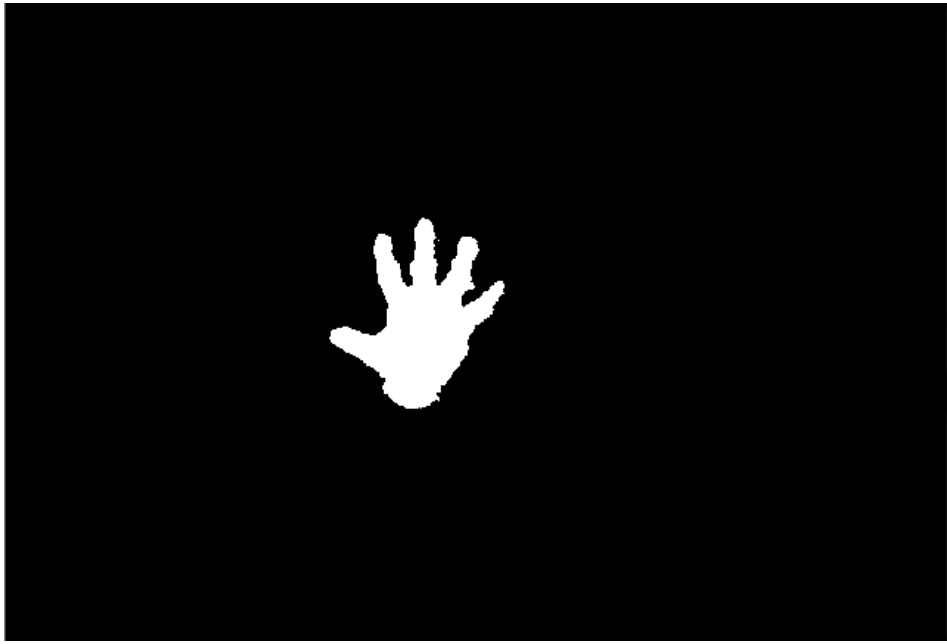


Figure 3.3.4 手部位置深度影像萃取圖

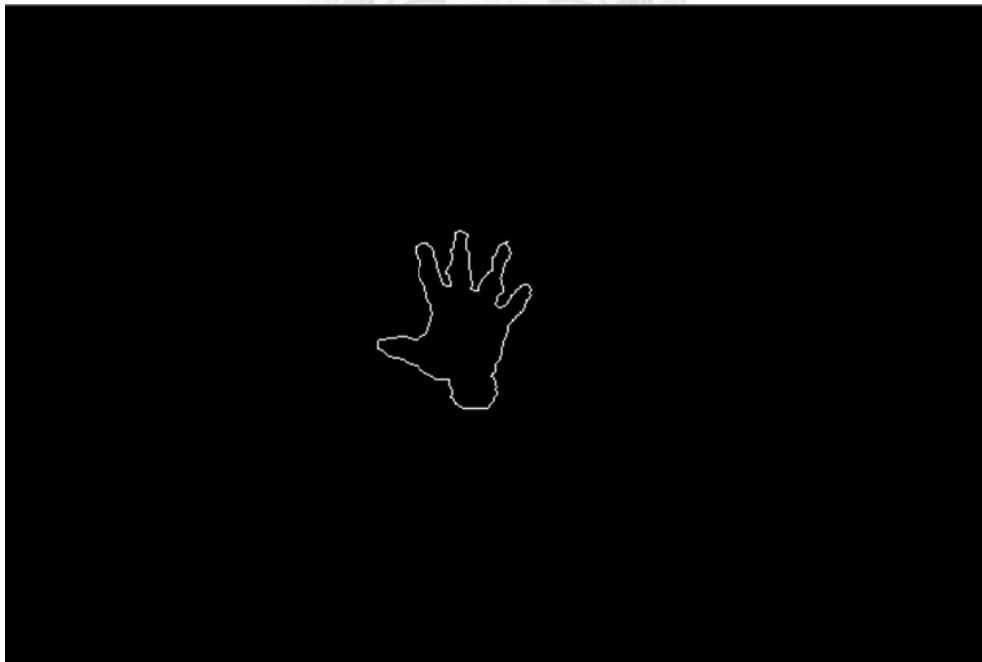


Figure 3.3.5 手部輪廓圖

C. 找出手部輪廓的凸包群集點集合:

在此,利用 OpenCV 裡的 `convexhull` 來找出手部輪廓的凸包群集點集合。

D. 手指指尖的座標位置的偵測

取得凸包群集資訊後,在此凸包的範圍內的二維影像也包含著手部的輪廓。我們希望透過此範圍內的手部輪廓來取的手指指尖的特徵點,因此,利用 *k*-Curvature 演算法來找出指尖的位置,此概念是利用兩向量的內積,去計算出兩向量之間的夾角,藉此找出指尖位置[13],藉由輪廓搜尋結果,得到一個手部輪廓,由一序列的 *x* 與 *y* 座標  $(x_1, y_1), (x_2, y_2), (x_3, y_3), \dots, (x_{n-1}, y_{n-1}), (x_n, y_n)$  所構成,假設其中一點 *A*, 距離 *A* 點距離 *k* 個座標點為 *B* 和 *C* (如下圖 Figure 3.3.6 所示),利用這三點取得兩個向量  $\vec{AB}$ 、 $\vec{AC}$ ,計算這兩向量的內積,可以得到兩向量間的夾角,假設夾角為  $\theta$ ,計算公式如(3.3.1):

$$\theta = \cos^{-1}\left(\frac{\vec{AB} \cdot \vec{AC}}{\|\vec{AB}\| \times \|\vec{AC}\|}\right) \quad (3.3.1)$$

可利用此關係,得到夾角  $\theta$  的最小角度,就可以定義出指尖的位置 (如下圖 Figure 3.3.7 所示)。

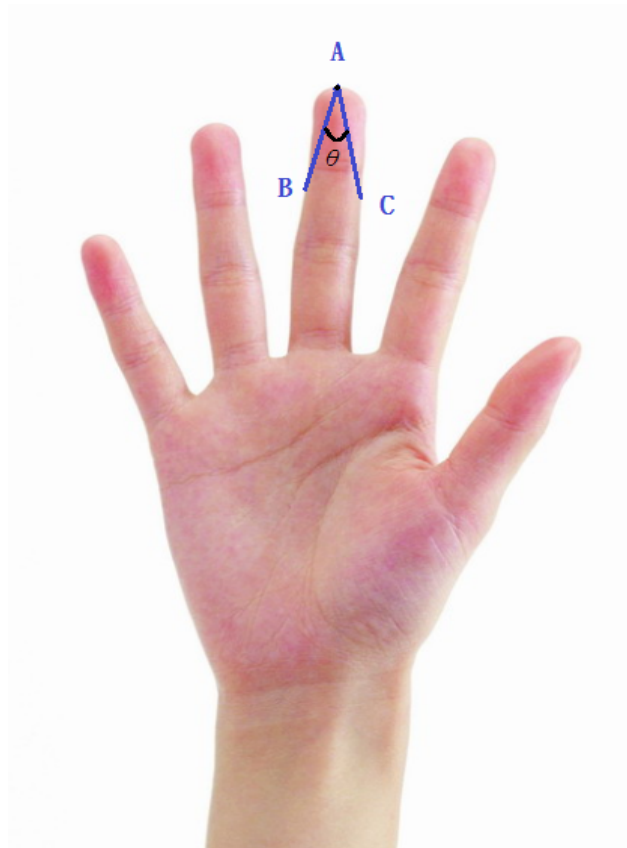


Figure 3.3.6 手指指尖夾角示意圖

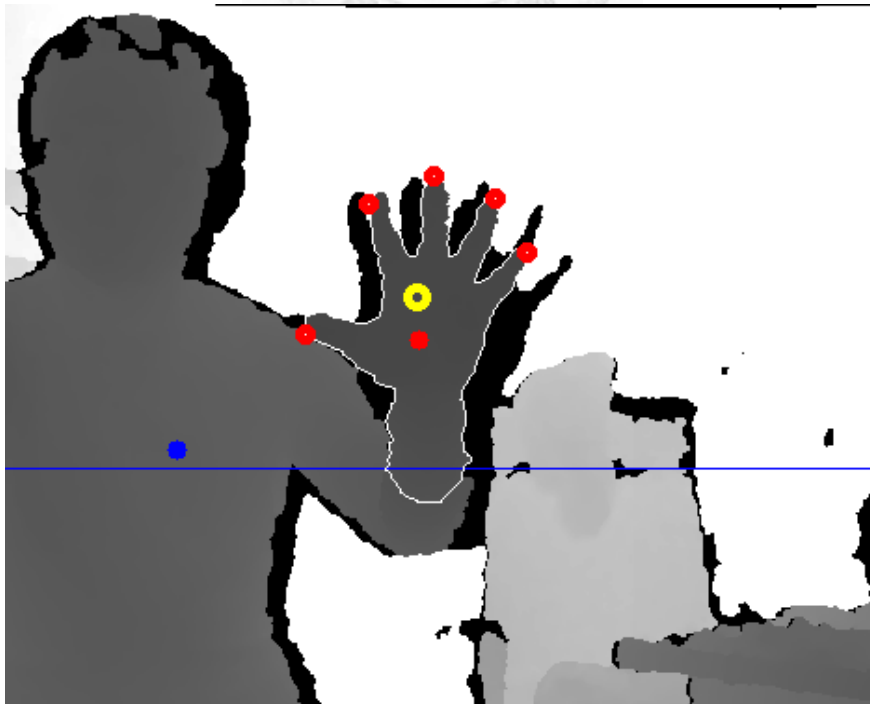


Figure 3.3.7 手指指尖示意圖

## 第四章 驗證過程與實驗結果

### 4.1 Kinect 三度空間座標系信賴度

在本章主要在討論與呈現相關的驗證資料以及測試結果。首先在 4.1 節裡，一開始會將 Kinect 體感應器所建構的真實世界三度空間座標系統做深度(Z 軸)、左右距離(X 軸)、上下距離(Y 軸)的驗證資料完整呈現；第二部分會針對手指指尖偵測結果做相關的驗證；最後藉由三度空間座標位置會導出手指的彎曲角度，在此會將其值做校正，接著會將校正結果的驗證資料在該小節介紹說明。

Kinect 體感應器在產生真實空間座標系統時，主要是先利用 Light Coding 的技術先得到空間中所有可以量測到點的深度值，也就是以感應器為零點，垂直水平射出方向為正 Z 軸的 Z 座標；得知 Z 座標後配合感應器視角的延伸，基本上 Kinect 體感應器在空間中的視野像是一個四角錐狀體(如下圖 Figure 4.1.1)，不同的深度會有不同的視野大小，雖然轉換到投影座標系時都是 640\*480，但在真實世界的座標系每個深度的水平、垂直座標的範圍都不一樣。

在 Kinect 三度空間座標系統信賴度驗證過程，首先，先對 Z 軸座標來做驗證。針對深度資訊準確度分析，在此採用了萊卡公司所出產的手持雷射測距儀為基準，其商品明及型號為 Leica DISTO<sup>TM</sup> D3a BT (如下圖 Figure 4.1.2 所示)。此測距儀常應用在建築、土地、空間面積、傾斜角度等的量測。本身具有藍芽數據傳輸功能且操作簡便。在空間中 Z 軸座標驗證的實驗規劃一開始先固定 Kinect 體感應器座落位置，以及雷射測距儀的位置，並刻劃出與感應器的距離尺線，每 10 公分為一

間距，從 0 到 140 公分(如下圖 Figure 4.1.3 與 Figure 4.1.4 所示)。經 OpenNI 來回報 Kinect 體感應器的深度值，最小可偵測距離約 50 公分，並參考 Kinect 在 OpenNI 下深度資料性質圖(如下圖 Figure 4.1.5 所示)。因此，將待測物從尺標 50 公分開始量測，每 0.5 公分為一個量測間距，擷取 Kinect 及雷射測距儀所回報之 5 筆深度資訊並做其平均輸出，量測深度距離為 50 公分到 130 公分(如下圖 Figure 4.1.6 所示)。

經由 OpenNI 回報的 Kinect 體感應器空間中深度值，以及由雷射測距儀所量測到的真正空間中的深度值，統計分析其 Kinect 體感應器之深度距離的誤差值(4.1.1)，可以發現誤差值會隨著距離增加而提高(如下圖 Figure 4.1.7 所示)。如果將其轉換成誤差率來看時，可以發現從開始量測之 50 公分起算到 130 公分，單位距離內的平均誤差率雖然與距離增加成正比，但在量測範圍內 Kinect 感應器之深度距離的平均誤差率(4.1.2)均不超過 1%(如下圖 Figure 4.1.8 所示)。

$$\text{誤差值} = |\text{Kinect 之深度值(mm)} - \text{雷射測距儀量測之深度值(mm)}| \quad (4.1.1)$$

$$\text{誤差率(\%)} = (\text{誤差值} / \text{雷射測距儀量測之深度值}) * 100 \quad (4.1.2)$$

由以上的 Kinect 感應器 Z 軸深度值的量測結果與雷射測距儀量測距離比較，可以發現在距 Kinect 感應器深度範圍在 50 公分到 130 公分以內，由 Kinect 所量測到深度值之平均誤差率可以控制在 1% 以下。在接下來的 X 軸與 Y 軸距離的驗證，這項數據是一個強而有力的基礎。

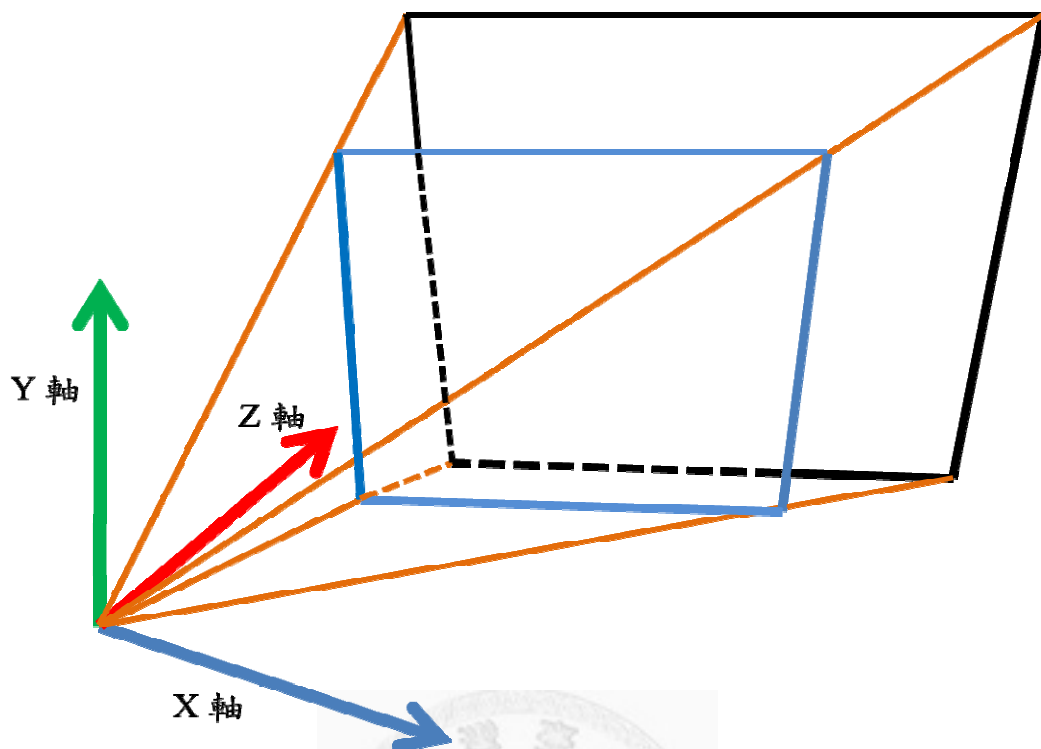


Figure 4.1.1 Kinect 體感應器真實世界視野範圍



Figure 4.1.2 Leica DISTO™ D3a BT 產品圖

(圖取自: <http://ptd.leica-geosystems.com/en/index.htm>, 萊卡公司官網)

Table 4.1.1 Leica DISTOTM D3a BT 基本技術參數[12]

技術參數	D3a BT
測量精度	$\pm 1.0 \text{ mm}$
測量範圍	0.05 m ~ 100 m
傾斜傳感器中量測角度	$\pm 45^\circ$
面積 / 體積計算	有
重量 (含電池)	149 g

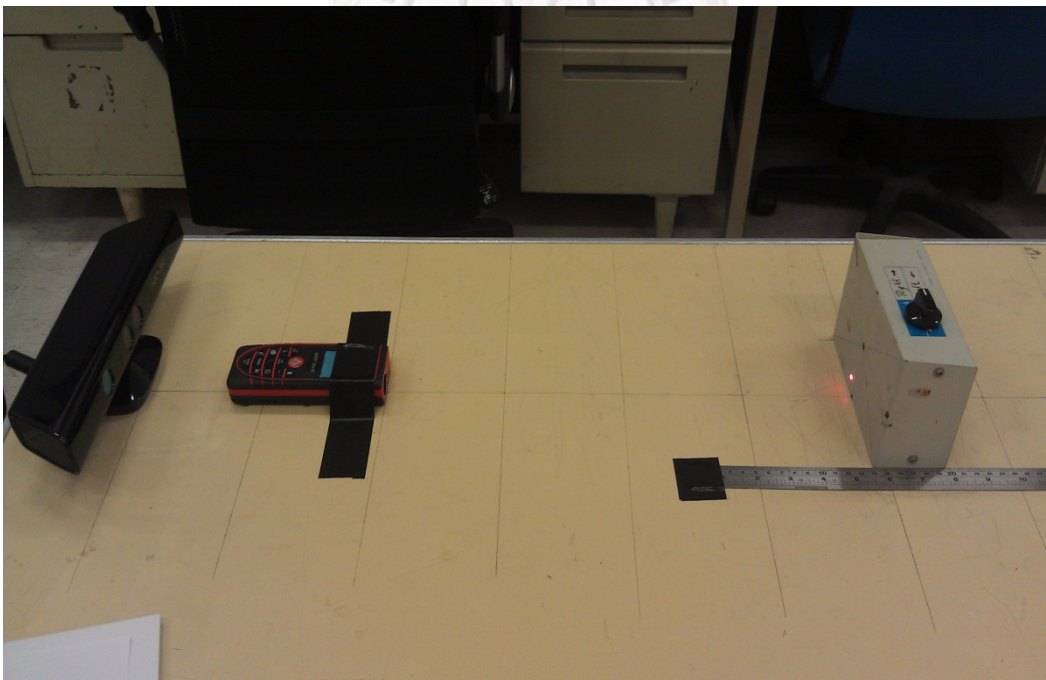


Figure 4.1.3 深度驗證實驗側視圖





Figure 4.1.4 深度驗證實驗俯視圖

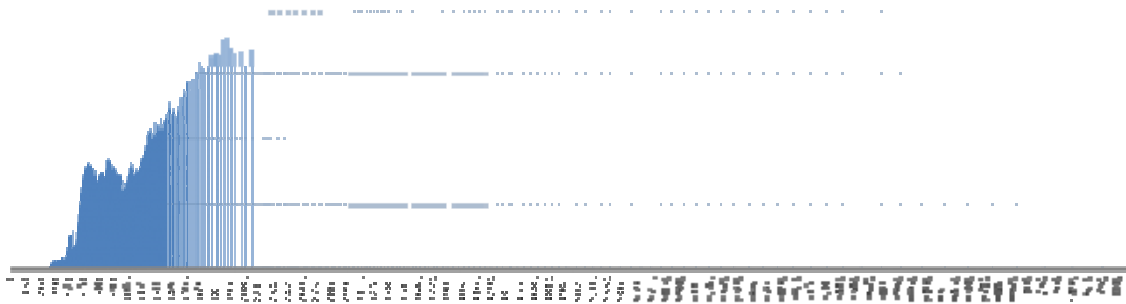


Figure 4.1.5 Kinect 在 OpenNI 下深度資料性質圖[6]

(上圖是透過 Kinect 連續錄下 50 的畫面的深度分布統計資料，橫軸是深度(mm)、  
縱軸則是該深度所抓到的點的數量)

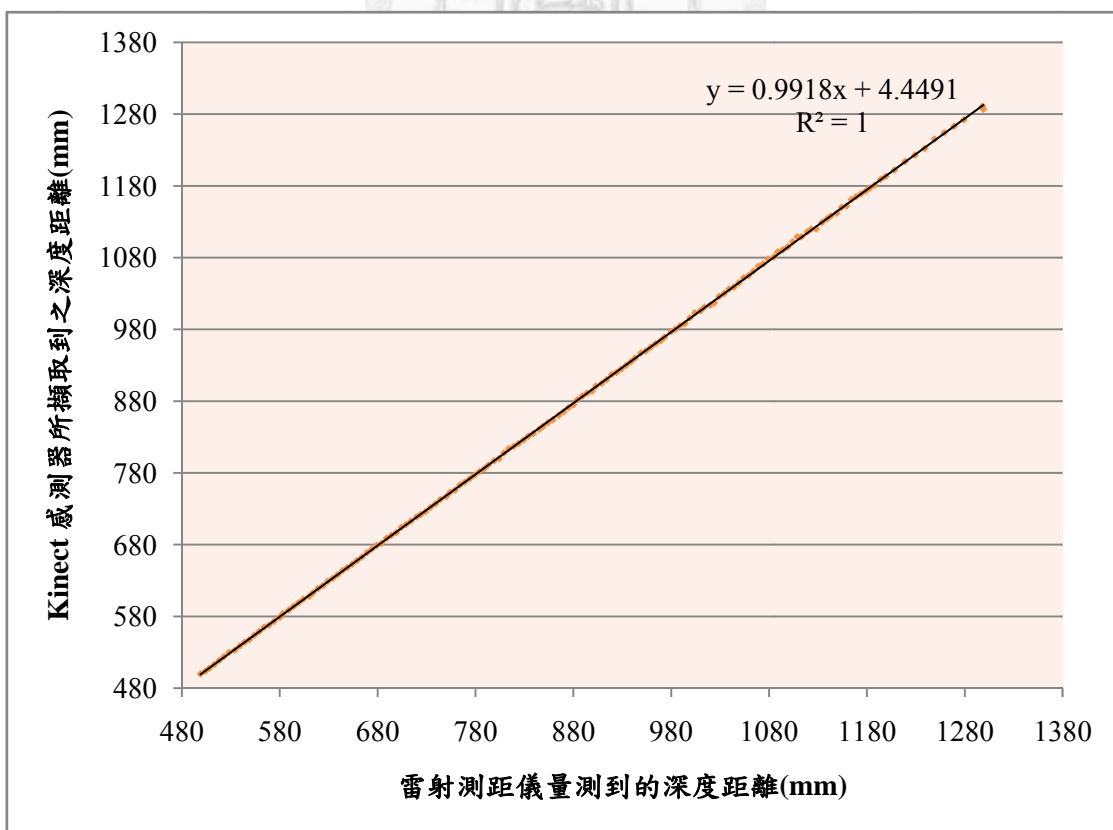


Figure 4.1.6 雷射測距儀與 Kinect 感應器所量測到深度距離對應圖

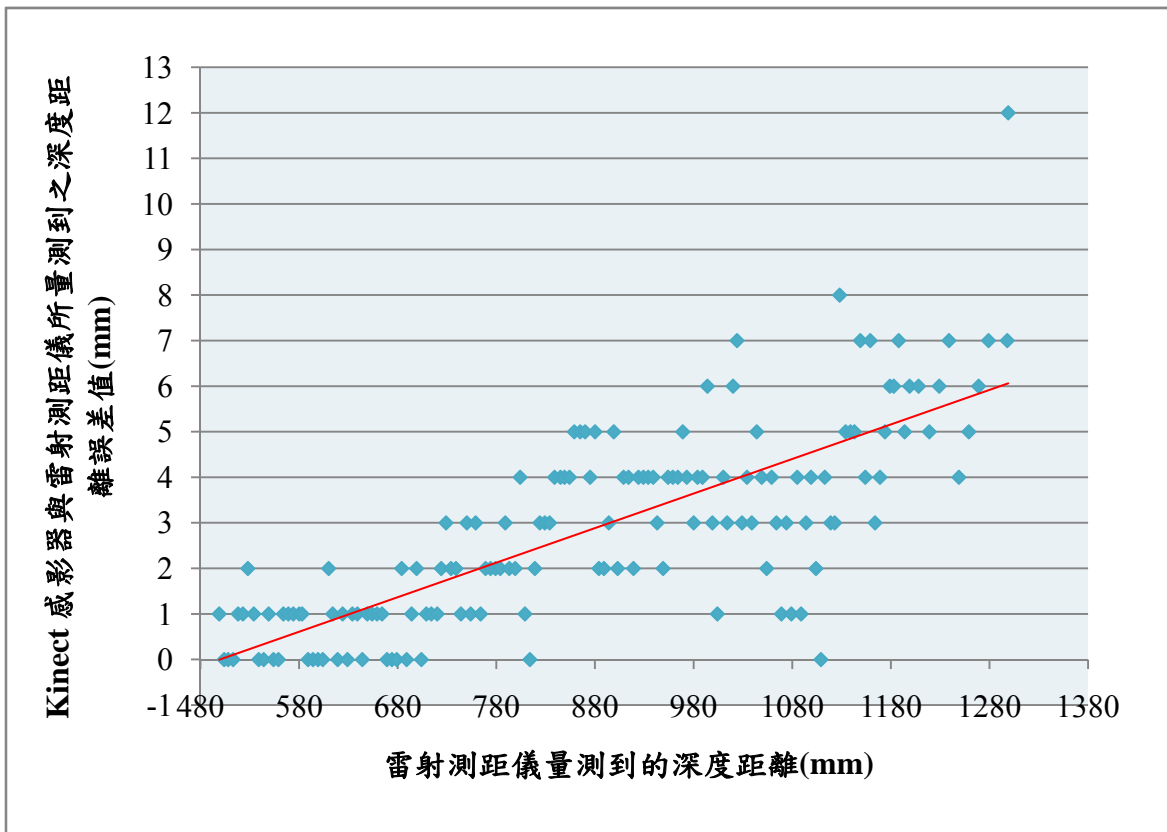


Figure 4.1.7 Kinect 體感應器深度距離誤差值示意圖

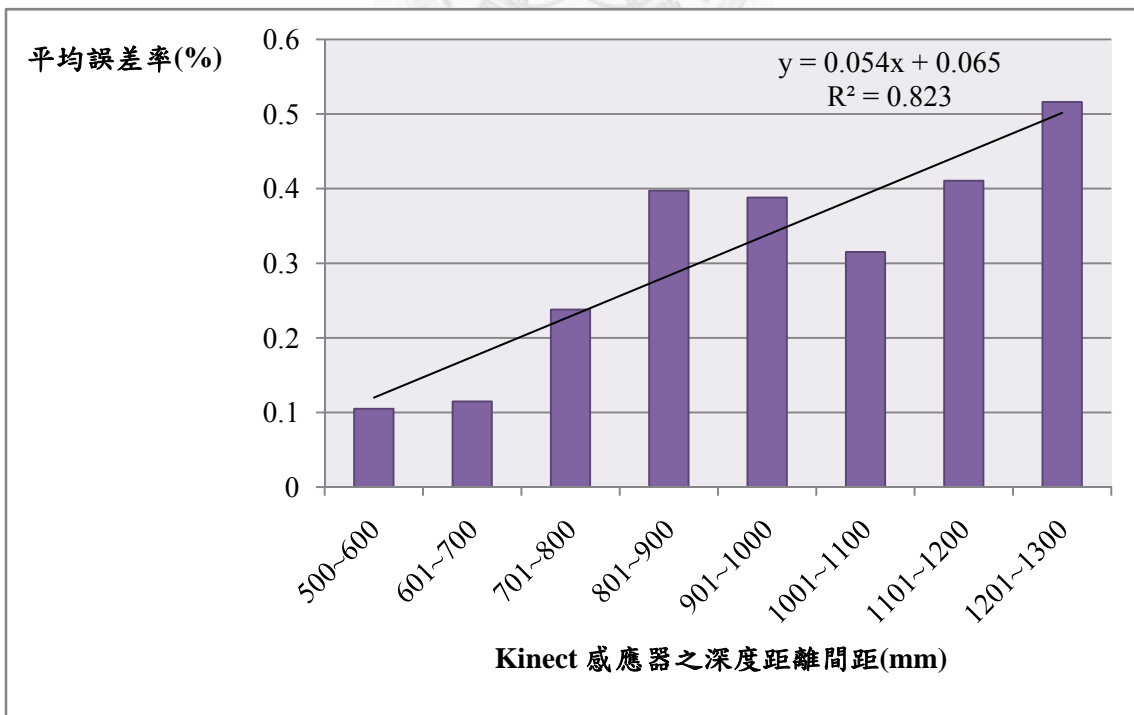


Figure 4.1.8 體感應器單位距離之深度值平均誤差率

X、Y 軸座標信賴度的驗證，在此是採用水平距離以及垂直距離量測資料來做為驗證之依據。本實驗先選取一個已知水平距離(146mm)及垂直距離(102mm)的量測物(如下圖 Figure 4.1.9 所示)，在選取量測物時必須考量到該量測物的量側面是否反光或太過光滑，因為以上兩種情況均會讓 Kinect 體感應器偵測不到任何的資訊。量測範圍從距離 Kinect 感應器 50 公分到 130 公分，每一公分收取一份由 Kinect 量測到水平及垂直距離資料；取得距離的方法為：利用投影座標系所產生的 2D 影像，利用滑鼠點選欲輸出了水平、垂直距離的範圍，投影座標轉換至真實座標後將其輸出並做平均分析(如下圖 Figure 4.1.10 到 Figure 4.1.15 所示)。由以上資料，可以推論 Kinect 體感應器在無人體自動追焦的情況下，對於水平距離以及垂直距離的量測誤差和平均誤差率，在 70 公分至 110 公分這 40 公分的範圍內最低。

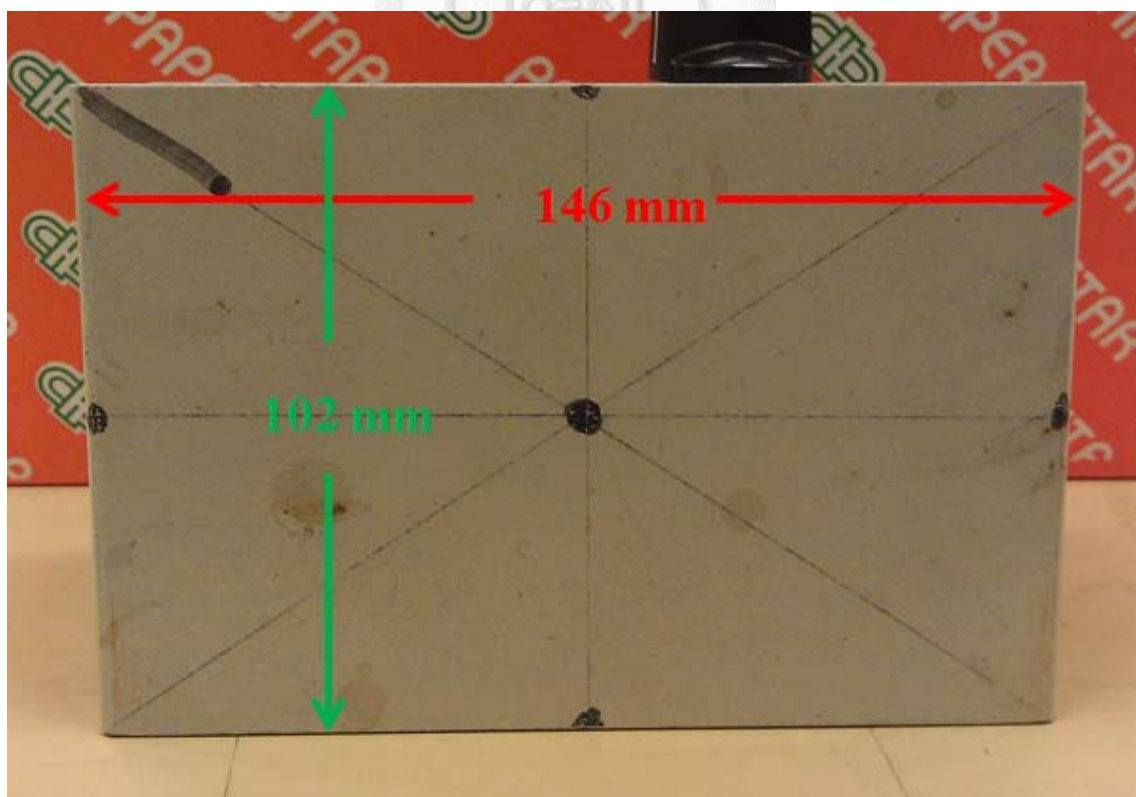


Figure 4.1.9 量測物

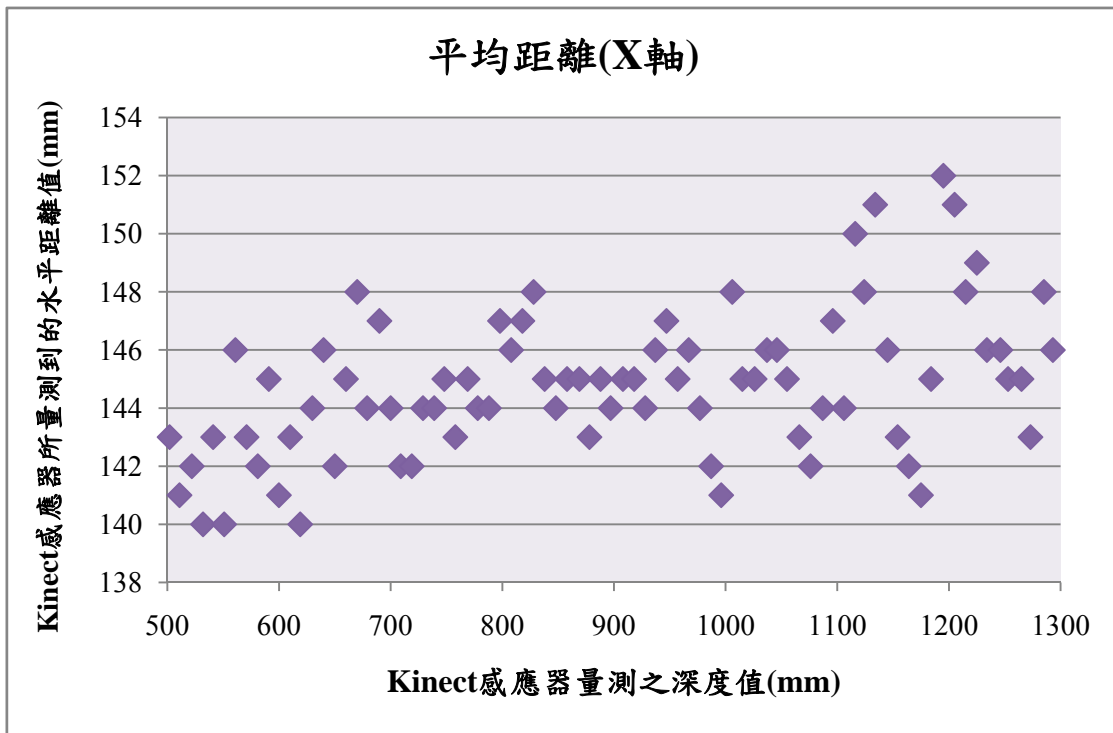


Figure 4.1.10 不同深度水平平均距離量測結果

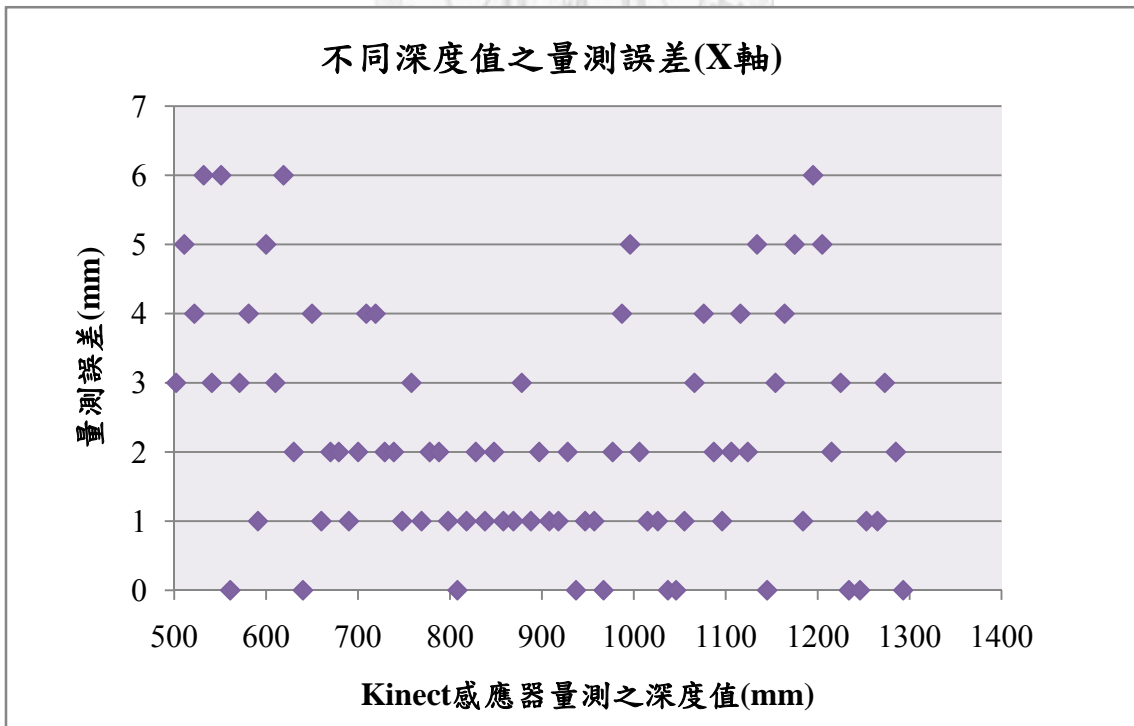


Figure 4.1.11 不同深度水平距離量測誤差結果

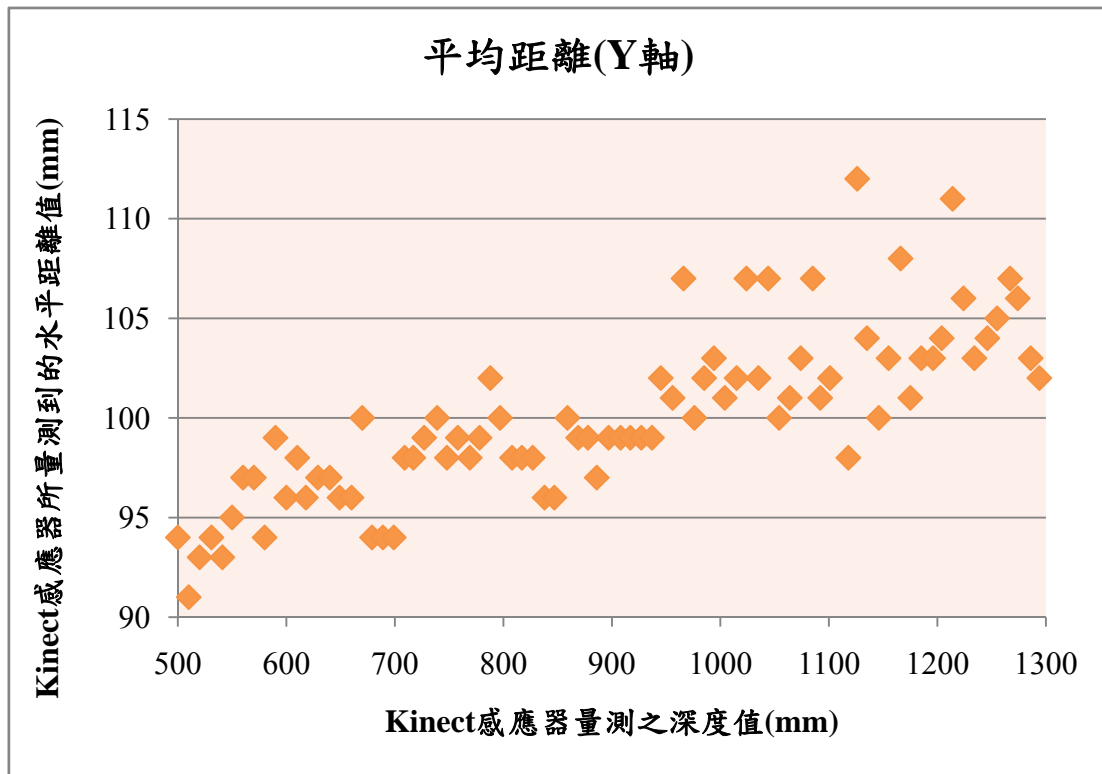


Figure 4.1.12 不同深度垂直平均距離量測結果

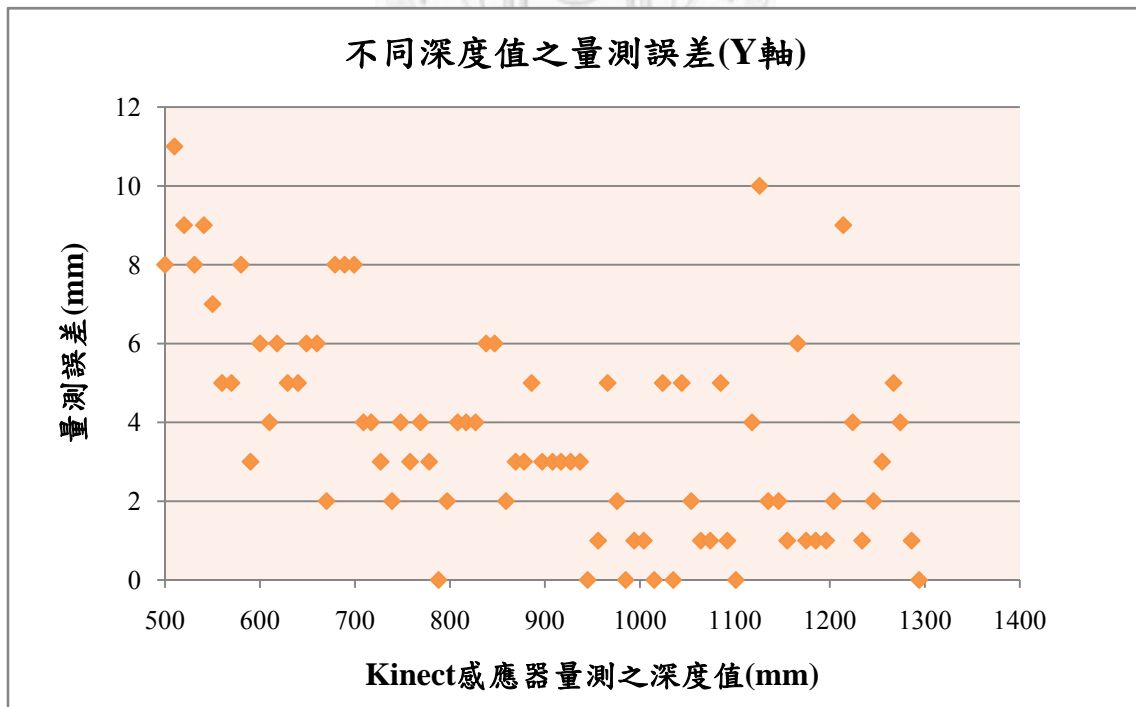


Figure 4.1.13 不同深度垂直距離量測誤差結果

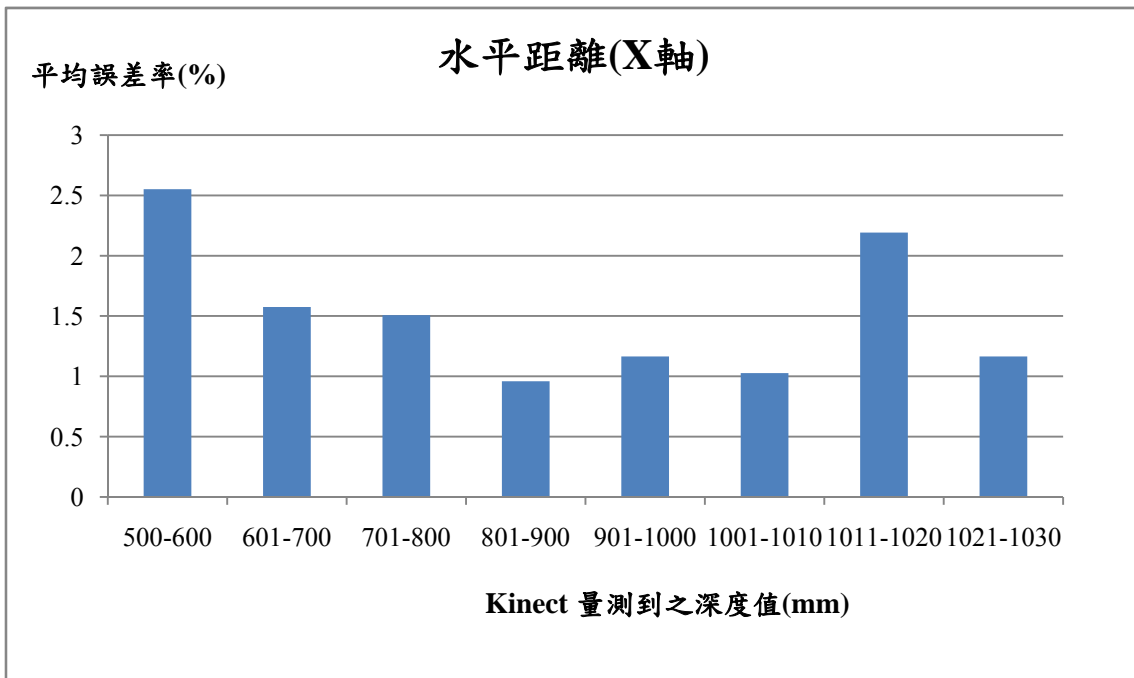


Figure 4.1.14 不同深度水平距離平均誤差率

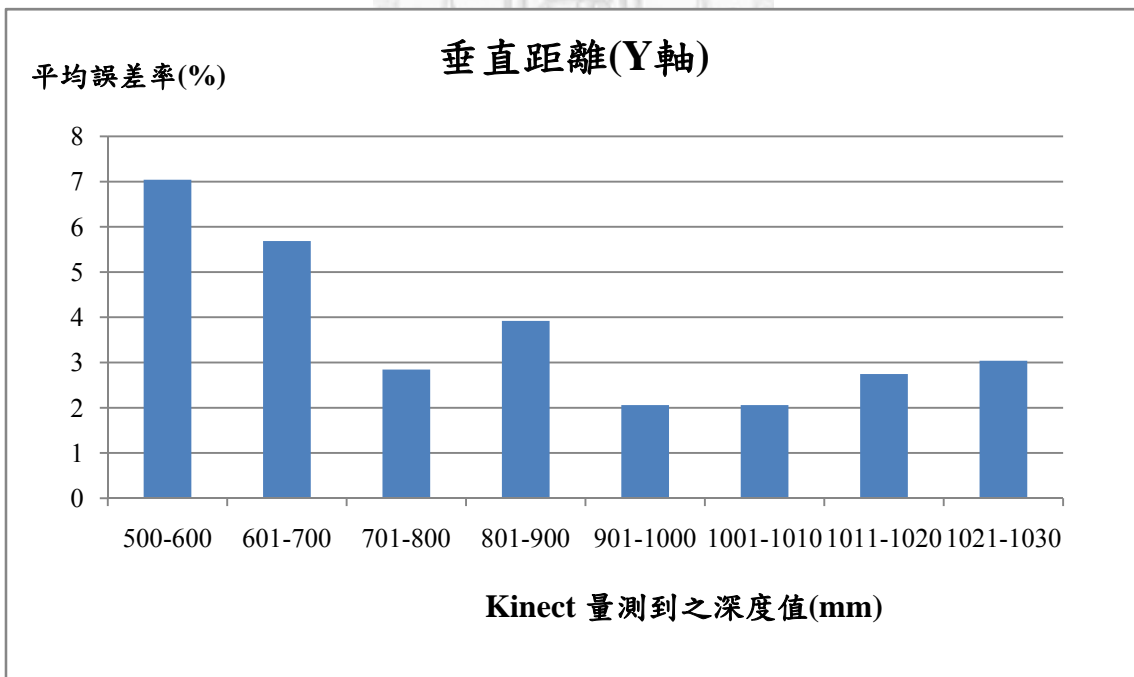


Figure 4.1.15 不同深度垂直距離平均誤差率

## 4.2 手指指尖偵測結果

### 4.2.1 不同深度範圍偵測結果

在本章節會利用自製的假手模型(如下圖 Figure 4.2.1 所示)，在不同的深度間距偵測五個手指指尖的位置(如下表 Table 4.2.1 所示)，在此量測的深度範圍為 50 公分到 130 公分之間。



Figure 4.2.1 假手模型



Table 4.2.1 不同深度間距指尖偵測圖




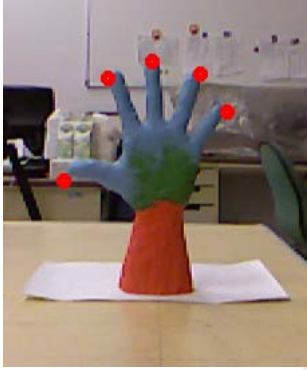




		
50~60 公分	60~70 公分	70~80 公分
		
80~90 公分	90~100 公分	100~110 公分
		
110~120 公分	120~130 公分	

Table 4.2.2 手指指尖偵測研究設備及開發環境

CPU	Intel® Core™ i5-2400 CPU @ 3.10GHz
RAM	16 GB
顯示卡	AMD Radeon HD 6700 Series
實做軟體	Microsoft Visual Studio 2010 OpenCV2.3.1
實驗平台	Window 7
攝影機	XBOX 360 Kinect Sensor



## 4.2.2 偵測穩定度分析

在機率統計中，平均絕對值誤差率(Mean Absolute Percentage Error, MAPE)為 Lewis 在 1982 年提出，客觀評估績效的統計量。是利用實際值( $y_t$ )與估算值也就是預測值( $\hat{y}_t$ )之間的差直取絕對值後，在求其平均數，係用來計算實際值與估算值的差異，用以評估量測模型的好壞(4.2.1)。由此公式可以推得，MAPE 越小表示實際值的量測能力越好。本實驗中為了提高取樣樣本佔母體百分之九十九的信賴區間，且量測誤差在 1 mm 以內，根據需最少量測樣本數的公式(4.2.2)，得知最少需取樣約 405 次，本實驗取樣數為 1000 次。可以用來評估量測模型好壞的還有均方誤差(Mean Squared Error, MSE)(4.2.3)，在此章節將會呈現這兩種評估方式的結果。

$$MAPE = \frac{1}{N} \sum_{i=1}^N \left| \frac{e_t}{y_t} \right| \times 100\% \quad (4.2.1)$$

N: 樣本數； $e_t$ (誤差):  $y_t - \hat{y}_t$

$$N = \left( \frac{C \times STD}{minError} \right)^2 \quad (4.2.2)$$

$C = 2.58$ ，代表參數估計達 0.01 顯著水準，這關係表示每一百次就會有 99 次會成立，也就是 99% 的信賴區間； $STD = 7.8$ ，代表本實驗量測到的手指指尖座標位置最大標準差(mm)； $minError = 1$ ，表示誤差要在 1 mm 以內。

$$MSE = \frac{1}{N} \sum_{i=1}^N e_t^2 \quad (4.2.3)$$

N: 樣本數； $e_t$ (誤差):  $y_t - \hat{y}_t$

本實驗中首先假設量測到數據的平均值為估算值，也就是先將其平均值視為真正值，再利用量測到的實際值與其比較，計算平均絕對值誤差率(MAPE)與均方誤差(MSE)來當作其手指指尖偵測穩定度分析。以 MAPE 做為評估模型好壞績效指標，只要是因為 MAPE 為相對數值，不受實際值與預測值單位的影響，能客觀地獲得其相對差異。若值越接近於 0，表示估計效果較佳，在此可視為量測結果較穩定。Lewis 針對 MAPE 大小提出說明(如下表 Table 4.2.3 所示)。

Table 4.2.3 MAPE 評估標準表[26]

MAPE(%)	說明
<10	高度精準的預測
10 ~ 20	優良的預測
20 ~ 50	合理的預測
>50	不準確的預測

以下圖 Figure 4.2.2 ~ Figure 4.2.6，是針對每一個手指指尖所量測到的空間座標軸(X,Y,Z)分別計算出在不同深度區間的 MAPE。由這些圖可以發現，除了中指指尖部分所量測之 X 座標以及 Y 座標的 MAPE 有超過 10 以外，其他均小於 10；雖然中指指尖部分較不穩定但其 MAPE 也都小於 50，所以都是合理的範圍內。

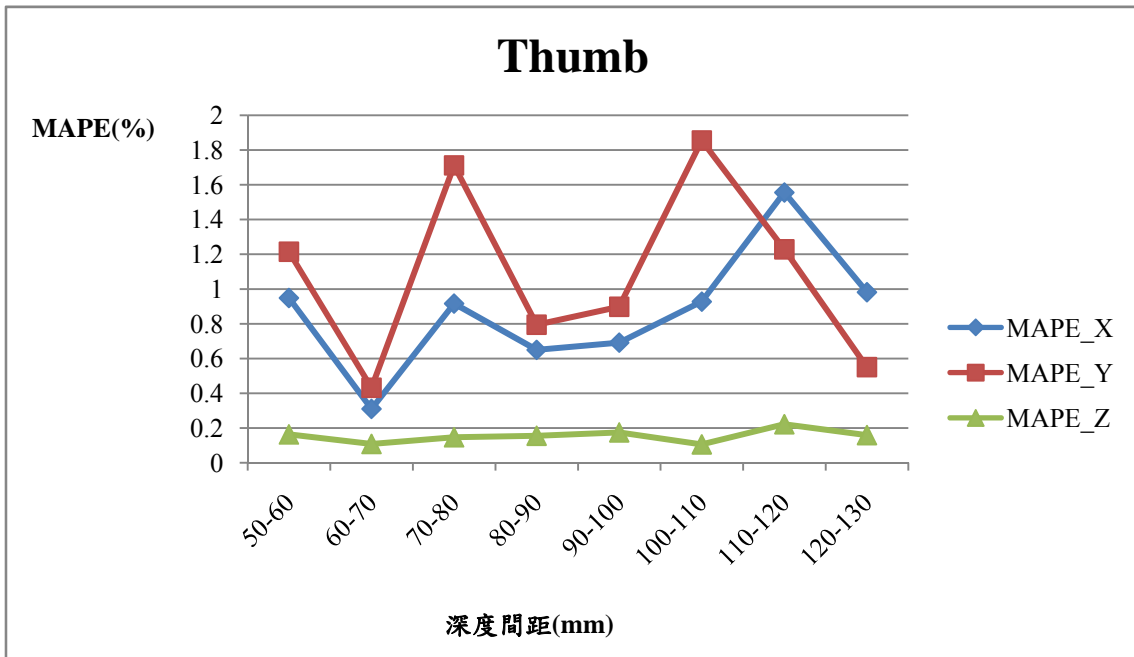


Figure 4.2.2 大拇指 X、Y、Z 座標在不同深度距離之 MAPE

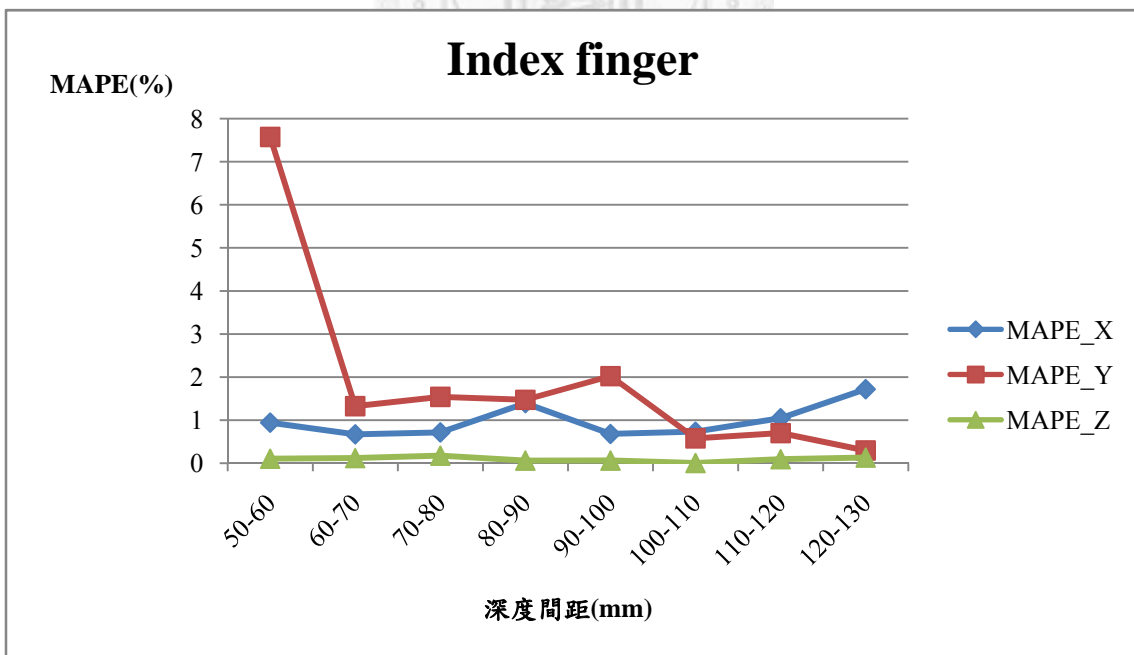


Figure 4.2.3 食指 X、Y、Z 座標在不同深度距離之 MAPE

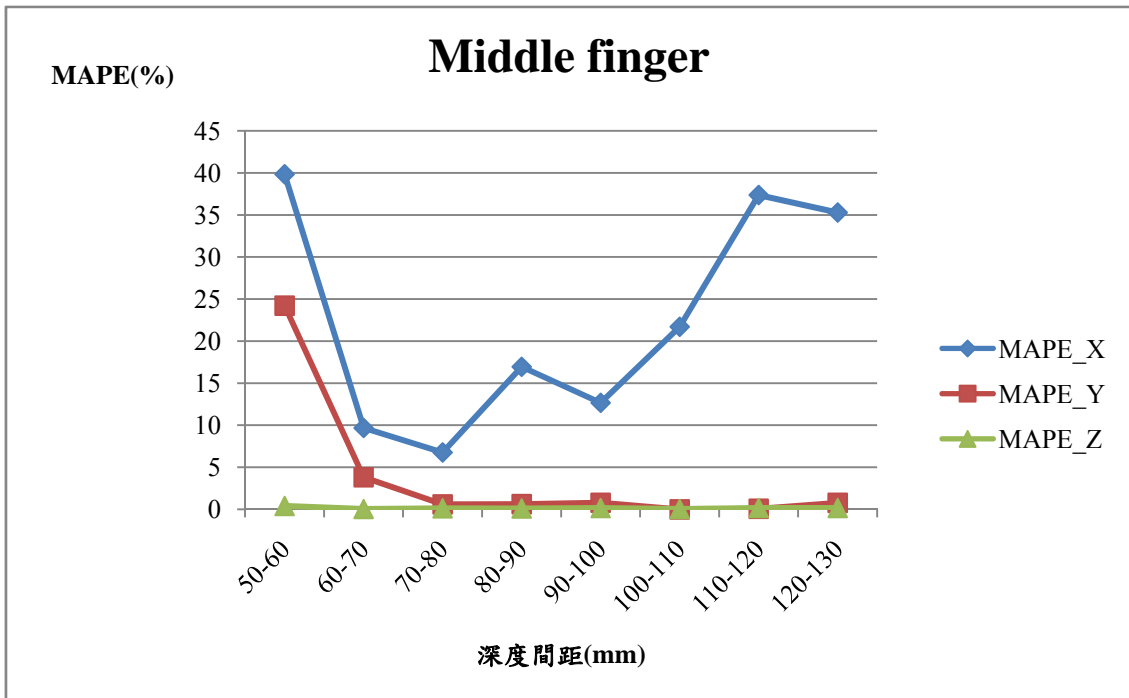


Figure 4.2.4 中指 X、Y、Z 座標在不同深度距離之 MAPE

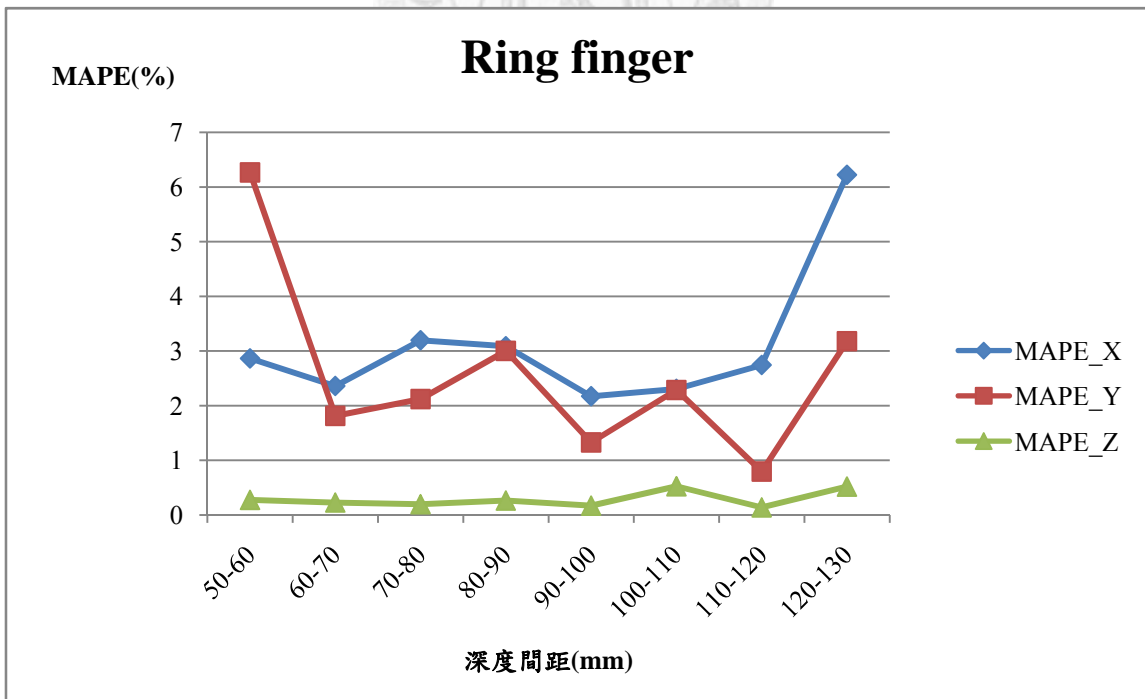


Figure 4.2.5 無名指 X、Y、Z 座標在不同深度距離之 MAPE

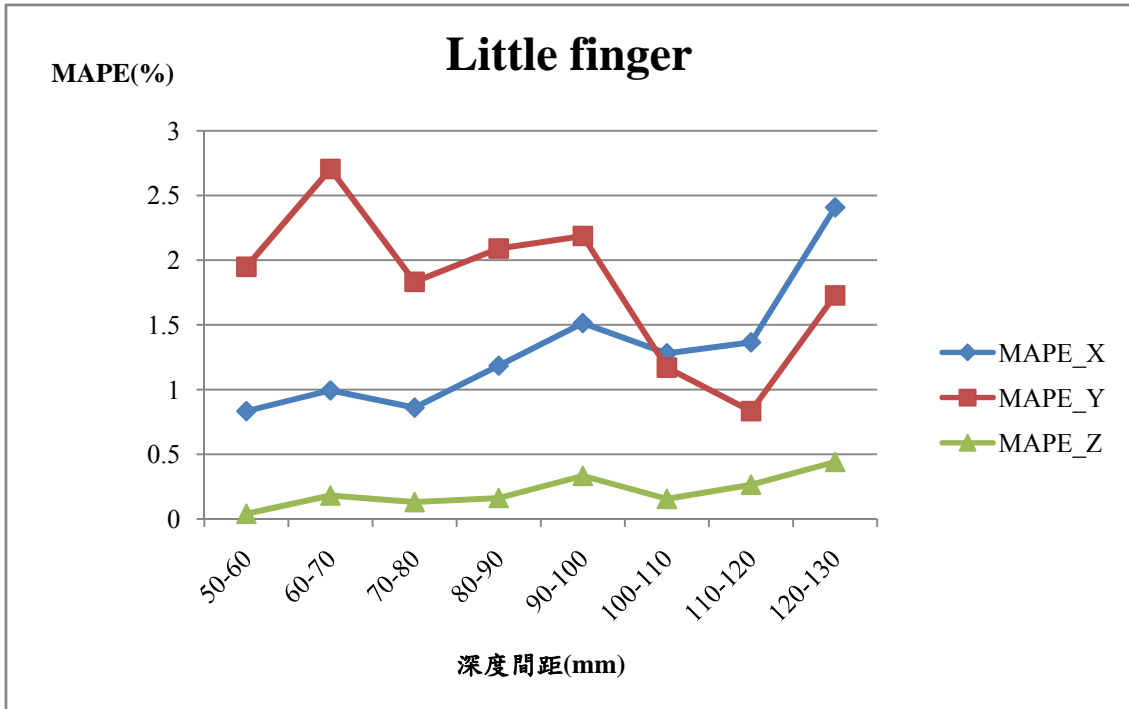


Figure 4.2.6 小拇指 X、Y、Z 座標在不同深度距離之 MAPE

在此假定每個指尖偵測到的平均值為真實座標位置，因此，接下來本實驗會利用均方差(MSE)來做為所有實驗數據的有效性(Efficiency)分析，如下圖(Figure 4.2.7 ~ Figure 4.2.11 所示)。最後計算量測到的每個指尖座標與平均值座標的平均誤差距離(如下圖 Figure 4.2.12 所示)。由以上這些資料可以得知，當深度距離提高時，不論是 MSE 或是平均誤差距離也都會隨著深度增加而呈現提高的趨勢，也可以將其解釋為當深度距離漸漸增加時，其系統量測的有效性會漸漸下降。

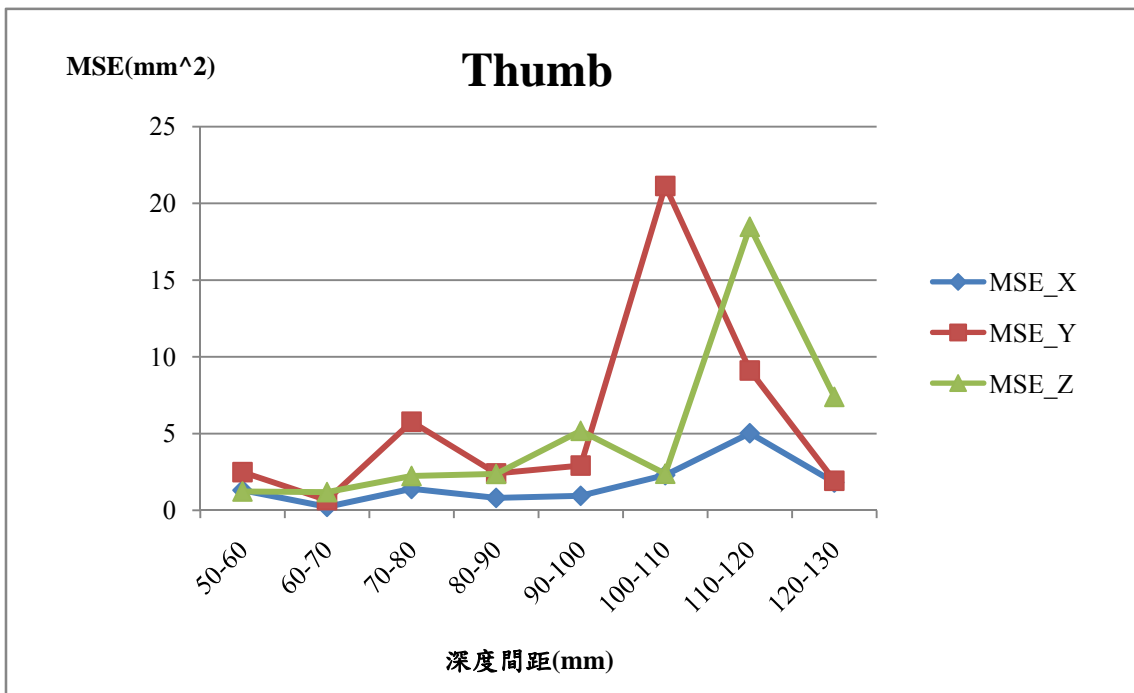


Figure 4.2.7 大拇指 X、Y、Z 座標在不同深度距離之 MSE

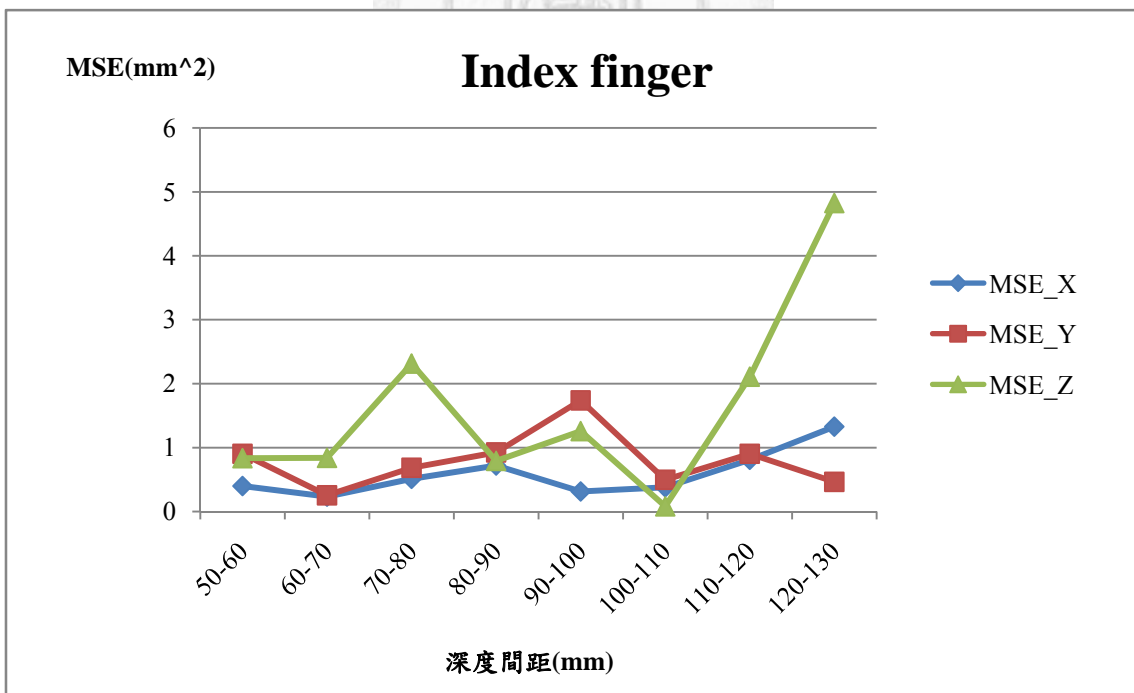


Figure 4.2.8 食指 X、Y、Z 座標在不同深度距離之 MSE



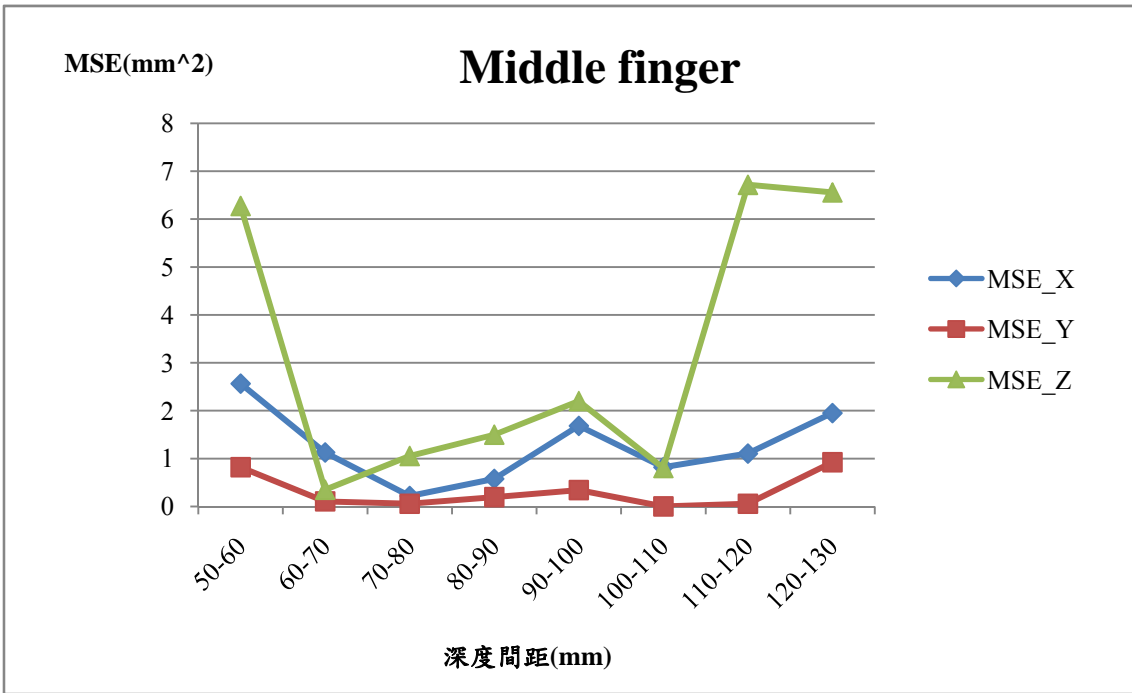


Figure 4.2.9 中指 X、Y、Z 座標在不同深度距離之 MSE

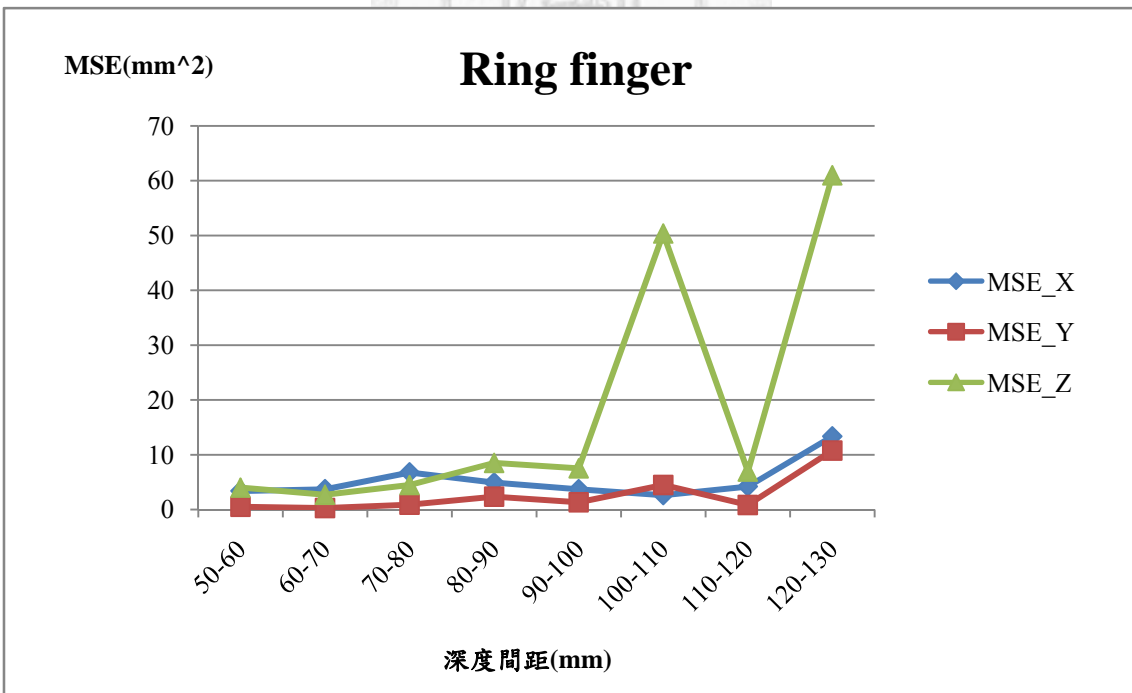


Figure 4.2.10 無名指 X、Y、Z 座標在不同深度距離之 MSE

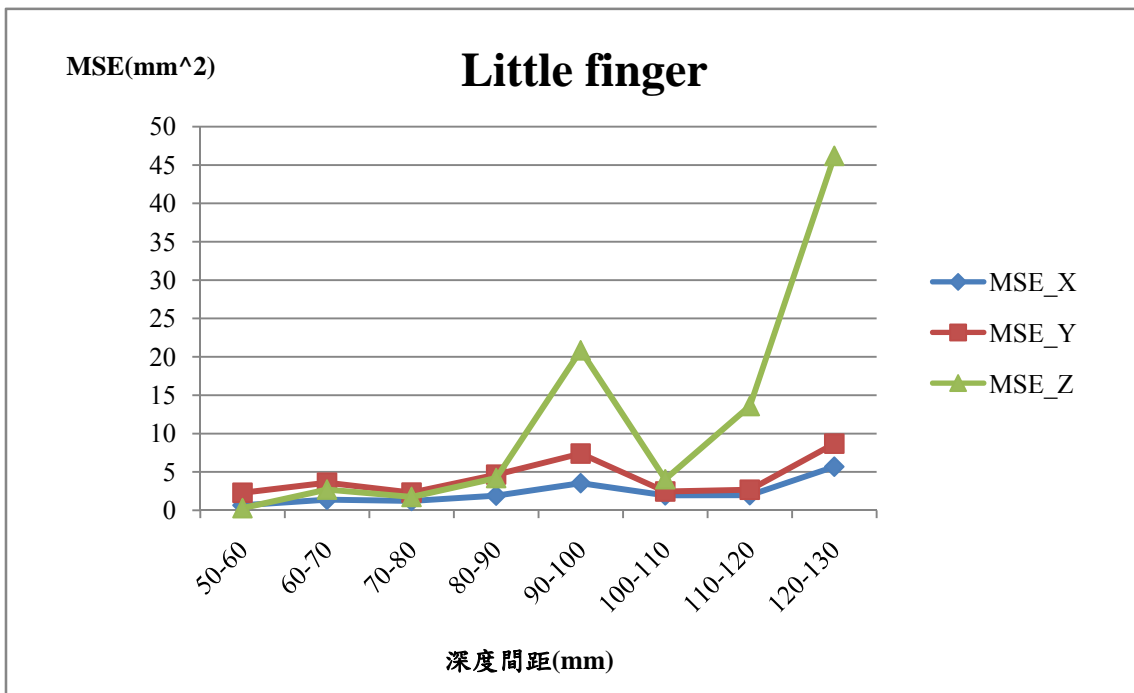


Figure 4.2.11 無名指 X、Y、Z 座標在不同深度距離之 MSE

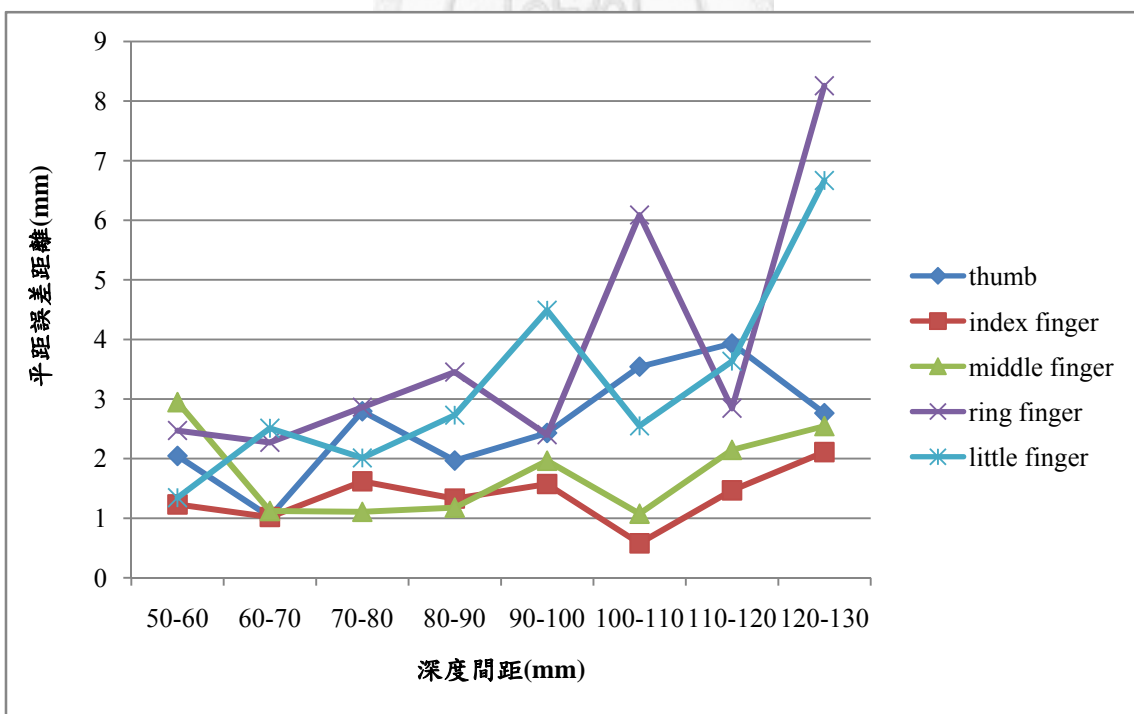


Figure 4.2.12 量測座標與平均值座標之平均誤差距離

### 4.3 手指指尖偵測之手指彎曲角度極限

在量測手指最大彎曲角度的實驗裡，利用自製的假手模型(如下圖 Figure 4.3.1 所示)，其中手指是可拆卸式，將其假手平面與 Kinect 體感應器的 Z 軸方向垂直，分別在不同深度範圍內量測每個手指在手指指尖偵測上所能彎曲的最大約略角度(如下表 Table 4.3.1 所示)。由量測數據顯示，大拇指部分在 50-130cm 的量測深度範圍內，其最大能偵測到的彎曲角度約為 40-45 度；食指約是 30-38 度、中指約在 30-40 度、無名指約在 28-30 度、小拇指約是 40-43 度之間。



Figure 4.3.1 手指彎曲角度測試假手示意圖

Table 4.3.1 假手在不同深度間距下的最大彎曲角度(°)約略值

最大 彎曲角度 深度間距	手指				
	thumb	index finger	middle finger	ring finger	little finger
50-60cm	40	30	35	30	42
60-70cm	40	35	39	30	42
70-80cm	40	38	40	28	40
80-90cm	40	38	38	30	40
90-100cm	45	32	38	28	43
100-110cm	45	36	38	30	42
110-120cm	45	35	30	29	40
120-130cm	43	32	35	28	40



## 第五章 討論與未來展望

### 5.1 討論

本研究的系統架構主要是利用微軟的 Kinect 體感應器為影像擷取系統，利用此感應器中 Light Coding Technology 所建構出來的深度影像為基礎，將其應用到真實空間中的座標系統量測及驗證，並利用深度影像資訊來擷取空間中手指指尖的座標位置。本論文主要是來探討 Kinect 體感應器在真實空間中作為手部活動擷取系統的可行性分析，希望將來可以將其推廣應用在手部復健醫學上。主要的驗證評估方式可分為以下幾類：

壹、Kinect 體感應器所建構之真實三度空間座標系的信賴度驗證分析，在此實驗中，

我們主要先針對 Kinect 體感應器的深度資訊做完整的量測；因為此感應器所建立的真實三維座標系統是以深度座標軸(Z 軸)為出發，進而建立縱座標系(Y 軸)及橫坐標系(X 軸)。從結果(圖 Figure 4.1.7 和 Figure 4.1.8 所示)可以得知 Kinect 體感應器的深度量測物差會隨深度距離變長而變大，探討其平均誤差率發現在本實驗的量測範圍內(50cm ~ 130cm)，其值雖然也是與深度距離呈正相關，但是均在 1% 以內。由以上數據可推知，在本實驗的量測距離內，Kinect 的深度資訊是在可以信賴的範圍之內。

貳、做完深度座標系(Z 軸)的驗證後，接下來就是 X、Y 座標系；在本論文中是利用量測水平以及垂直距離來做為驗證的依據。如結果(如圖 Figure 4.1.14 與 Figure 4.1.15 所示)得知，水平距離的量測平均誤差率在量測距離範圍內都可

以控制在 3%以下；垂直距離的平均量測誤差率在一開始的 50-60cm 與 60-70cm 內的誤差率比較高外，其餘都可以維持在 5%以內，會造成誤差如此龐大的原因可能是垂直距離的量測上出現的量測的不易判斷性，那是因為垂直距離的量測有一端是與桌面 90 垂直相交，有可能在那相交處會產生深度座標資訊的無訊號...等等。

參、手指指尖偵測演算法的偵測穩定度分析。由量測結果(表 Table 4.2.1 所示)可以發現在假手指尖的偵測上，在不同的深度距離，其偵測的結果均是可以接受的範圍。因此，接下來在不同的深度範圍裡讓系統連續地偵測其指尖座標 1000 次，並做穩定度分析。在此用來分析的工具為平均絕對值誤差率(Mean Absolute Percentage Error, MAPE)與均方誤差(Mean Squared Error, MSE)。首先，假設量測設具的平均值為真實座標，再利用以上兩種評估模式來評估系統、演算法量測的穩定度。由 MAPE 評估標準(如表 Table 4.2.3 所示)得知，除了中指指尖的數據在 50-60cm 以及 110-130cm 這區段間數值偏高，但還屬合理偵測範圍內(MAPE<50)，其他均是高度精準的預測(MAPE<10)。均方差(MSE)來做為量測數據的有效性(Efficiency)分析(如圖 Figure 4.2.7 ~ Figure 4.2.11 所示)。平均誤差距離(如圖 Figure 4.2.12 所示)。由以上這些資料可以得知，當深度距離提高時，不論是 MSE 或是平均誤差距離也都會隨著深度增加而呈現提高的趨勢，也可以將其解釋為當深度距離漸漸增加時，其系統量測的有效性會漸漸下降。

肆、手指指尖偵測之手指彎曲角度極限。由量測數據顯示，所有手指在 50-130cm 的量測範圍內，其最大能偵測到的彎曲角度大拇指約為 40-45 度、食指約是 30-38 度、中指約在 30-40 度、無名指約在 28-30 度、小拇指約是 40-43 度之間。以上訊息建議，當利用此偵測模式時，若需要將其手指彎曲程度的資訊

往下利用延伸，則受試者的手指擺放位置，以及所能彎曲的角度範圍勢必會受到相當程度的限制。

根據以上四個測試驗證所得的結果，在此做個總結。第一、Kinect 體感應器雖然在深度距離上的深度距離平均誤差率(如圖 Figure 4.1.8 所示)皆可以在 1% 以內，但是其隨著深度距離的提高，其真正的誤差值也是呈線性正比增加(如圖 Figure 4.1.7 所示)。Kinect 體感應器在建立真實三度空間座標時，是以 Z 軸座標為基準在建立起 X、Y 軸座標，所以當深度誤差值提高時，勢必也會影響到 X、Y 軸的準確性。如果遇偵測的目標是相對大的部位，如：大關節、elbow 關節、手腕關節等等，這些或許不會造成太大的影響，但若是像本次實驗的目標是細小的目標物(手指指尖)其影響會相對大許多。第二、在本實驗的手指指尖方式，所能偵測到的範圍因為光學原理而有所限制，所以才導致其最大的偵測角度只能大約在 30-45 度左右。



## 5.2 未來展望

科技日新月異，我相信在未來的 Kinect 體感應器的空間座標的準確度也會慢慢提升，所以接下來相關 Kinect 感應器在醫學上的應用是可以被期待的。目前本研究的驗證只在靜態假手的驗證，希望接下來也可以完成空間中手指指尖位置的動態驗證。在進行動態指尖偵測之前，可以再套用不同的手指指尖真的的演算法或是開發新的偵測模式來達到不受限制及更精準的結果。因為本實驗的隱含目的是希望利用 Kinect 體感應器來進行手部復健的輔具，所以在接下來也可以將 Kinect 感應器所量測到的空間座標資訊，與 Gold Standard (VICON System)來進行校正比對，增加其偵測的信賴度。





## 參考文獻

- [1] Thomas J. Lord, Yu li, Diana M. Keefe, Nikolay Stoykov, Derek Kamper, “Development of a haptic keypad for training finger individuation after stroke,” 2011 International Conference on Virtual Rehabilitation (ICVR), Zurich, Switzerland, June 2011, pp. 1-2.
- [2] Sriram Sanka, Prashanth G. Reddy, Amber Alt, Ann Reinthal, Nigamanth Sridhar, “Utilization of a wrist-mounted accelerometer to count movement repetitions,” 2012 Fourth International Conference on Communication Systems and Networks (COMSNETS), Bangalore, India, January 2012, pp. 1-6.
- [3] 蔡爭岳, “以類神經網路及 3D 資料建立靜態台灣手勢辨識系統”, 國立台灣科技大學工業管理系博士論文, 2007
- [4] sugizo, “身體就是感應器，微軟 Kinect 是怎麼做到的? Web Site,” April 2012. [Online]. Available: <http://www.techbang.com/posts/2936-get-to-know-how-it-works-kinect>
- [5] Jgospel, “微軟官方揭密 Kinect 工作原理 Web Site,” June 2012. [Online]. Available: [http://jgospel.net/news/tech/微軟官方揭秘\\_kinect\\_工作原理.\\_gc28095.aspx?location=TW](http://jgospel.net/news/tech/微軟官方揭秘_kinect_工作原理._gc28095.aspx?location=TW)
- [6] Heresy, “OpenNI / Kinect 相關文章目錄,” April 2012. [Online]. Available: [http://kheresy.wordpress.com/index\\_of\\_openni\\_and\\_kinect/](http://kheresy.wordpress.com/index_of_openni_and_kinect/)

- [7] OpenNI, “OpenNI User Guide,” April 2011. [Online].  
Available: <http://www.openni.org/documentation>
- [8] PrimeSensor, “PrimeSensor Nite Web Site,” April 2012. [Online].  
Available: <http://www.primesense.com/?p=515>
- [9] Wikipedia, “OpenCV,” April 2012. [Online].  
Available: <http://en.wikipedia.org/wiki/OpenCV>
- [10] Wikipedia, “OpenGL,” May 2012. [Online].  
Available: <http://en.wikipedia.org/wiki/OpenGL>
- [11] Wikipedia, “Kinect,” April 2012. [Online].  
Available: <http://en.wikipedia.org/wiki/Kinect>
- [12] 宏遠儀器, “Leica DISTO™ D3a BT,” June 2012. [Online].  
Available: <http://www.control-signal.com.tw/product.php>
- [13] J. Segen, S.Kumar, “Human-Computer Interaction Using Gesture Recognition and 3D Hand Tracking,” 1998 International Conference on Image Processing (ICIP), Chicago, USA, October 1998, vol. 3, pp. 188-192.

- [14] Stephan Rogge, Philipp Amtsfeld, Christian Hentschel, Dieter Bestle, Marcus Meyer, "Using gestures to interactively modify turbine blades in a virtual environment," 2012 IEEE International Conference on Emerging Signal Processing Applications (ESPA), Las Vegas, USA, January 2012, pp. 155-158.
- [15] K. K. Biswas, Saurav Kumar Basu, "Gesture recognition using Microsoft Kinect," 2011 5th International Conference on Automation, Robotics and Applications (ICARA), Wellington, New Zealand, December 2012, pp. 100-103.
- [16] Jagdish L. Raheja, Ankit Chaudhary, Kunal Singal, "Tracking of fingertips and centres of palm using Kinect," 2011 Third International Conference on Computational Intelligence, Modelling and Simulation (CIMSIM), Langkawi, Malaysia, September 2011, pp. 248-252.
- [17] Nadia Hocine, Abdelkader Gouaïch, "Therapeutic games' difficulty adaptation: an approach based on player's ability and motivation," 2011 16th International Conference on Computer Games (CGAMES), Kentucky, USA, July 2011, pp. 257-261.
- [18] Gwyn N Lewis, Claire Woods, Juliet A Rosie, Kathryn M McPherson, "Virtual reality games for rehabilitation: perspectives from the users and new directions," 2011 International Conference on Virtual Rehabilitation (ICVR), Zurich, Switzerland, June 2011, pp. 1-2.

- [19] Eduardo Souza Santos, Edgard A. Lamounier, Alexandre Cardoso, “Interaction in augmented reality environments using Kinect,” 2011 XIII Symposium on Virtual Reality (SVR), Uberlandia, Brazil, May 2011, pp. 112-121.
- [20] 何正宇, 王志龍, 盧玉強, 孫淑芬, 張照宏, 蔡欣宜, “以 Wii™ 建構虛擬實境輔助慢性中風患者復健訓練之療效評估,” 台灣復健醫誌, 38(1), pp. 11-18, 2010.
- [21] Satoshi Ito, Haruhisa Kawasaki, Yasuhiko Ishigure, Masatoshi Natsume, Tetsuya Mouri, Yutaka Nishimoto, “A design of fine motion assist equipment for disabled hand in robotic rehabilitation system,” Journal of The Franklin Institute, vol. 348, pp. 79-89, February 2011.
- [22] Sergei V Adamovich, Gerard G Fluets, Abraham Mathai, Qinyin Qiu, Jefferey Lewis, Alma S Merians, “Design of a complex virtual reality simulation to train finger motion for persons with hemiparesis: a proof of concept study,” Journal of NeuroEngineering and Rehabilitation, pp. 6-28, July 2009.
- [23] OpenGL, “OpenGL Web Site,” April 2012. [Online].  
Available: <http://www.opengl.org/>
- [24] Microsoft, “Microsoft Kinect Web Site,” April 2012. [Online].  
Available: <http://www.xbox.com/en-US/kinect>
- [25] OpenKinect, “OpenKinect Community Site,” April 2012. [Online].  
Available: <http://openkinect.org/>

- [26] 黃志偉, “高速公路肇事處理時間預測之研究-應用類神經網路分析”, 國立中央大學土木工程研究所碩士論文, 2002
- [27] Valentino Frati, Domenico Prattichizzo, “Using Kinect for hand tracking and rendering in wearable haptics,” 2011 IEEE World Haptics Conference (WHC), Istanbul, Turkey, June 2011, pp. 1054-1061.
- [28] Kouros Khoshelham, Sander Oude Elberink, “Accuracy and resolution of Kinect depth data for indoor mapping application,” Sensors, no. 2, pp. 1437-1454, February 2012.

