國 立 交 通 大 學

電 機 與 控 制 工 程 學 系

碩 士 論 文

可攜性力感系統於 3D 環境之研發與應用

Development of a Portable Force Reflection System in 3D environment and its Applications

研 究 生：陳李政

指導教授：楊谷洋 博士

中 華 民 國 九 十 六 年 七 月

可攜性力感系統於 3D 環境之研發與應用

Development of a Portable Force Reflection System in 3D environment and its Applications

研 究 生：陳李政　　Student：Li-Cheng Chen

指導教授：楊谷洋　　Advisor：Kuu-Young Young

國 立 交 通 大 學

電 機 與 控 制 工 程 學 系

碩 士 論 文

A Thesis

Submitted to Department of Electrical and Control Engineering

College of Electrical Engineering and Computer Science

National Chiao Tung University

in partial Fulfillment of the Requirements

for the Degree of

Master

in

Computer and Information Science

June 2006

Hsinchu, Taiwan, Republic of China

中華民國九十六年七月

可攜性力感系統於 3D 環境之研發與應用

研究生： 陳李政　　指導教授: 楊谷洋教授

國立交通大學電機與控制工程學系

摘要

由於電腦網路使用的普遍性與方便性，人與人間的互動不再因爲距離而受到限制，也發展出許多的遠端應用。目前的遠端呈現，聲音與影像都已相當普遍且技術上也相當成熟，但對於力感呈現方面，卻相對的少，大多數的力感呈現方式還停留在有向或無向的固定式震動居多。因此本論文主要是建置一個可攜性的力感系統，連結現有的力回饋裝置，將所架構的系統藉由 TCP / IP 網路通訊協定於遠端及近端相互傳遞資訊，並建立一個 3D 的虛擬環境使遠端及近端的力回饋裝置能互動；此外，我們也將討論在所建置的虛擬世界裡，如何對虛擬物體施力與力感的反應等。爲了提供逼真的物理反應，我們引入物理法則。我們成功地發展出互動的遠端操控模擬，並提供使用者真實的感受。

Development of a Portable Force Reflection System in 3D environment and its Applications

Student：Li-Cheng Chen　　Advisor：Dr. Kuu-Young Young

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

Along with the prevalence and convenience of the computer network, human interaction is no longer limited by the distance. Meanwhile, there have been many researches in teleoperation and telepresence. For telepresence, techniques in sound and image are now quite mature. By contrast, the sense of force is not yet well tackled; as most of them are implemented in the forms of fixed patterns. Therefore, in this thesis, we develop a portable force reflection system, using the force feedback joystick and corresponding softwares. We use the Internet as a command transmission medium via the TCP/IP protocol, and build the 3D environment using the virtual reality(VR) technology. Key issues for system implementation are discussed, including system structure, signal flow, and force management. To obtain ware realistic feeling, we intrude physics rules into object formulation. With the proposed system, the operator can manipulate the force feedback joystick to experience quite realistic feeling in a VR environment.

誌　謝

首先感謝我的指導教授----楊谷洋博士，在這兩年的研究期間，由於他熱心的指導，使我的研究工作得以順利完成。同時，感謝口試委員們：宋開泰教授、莊仁輝教授及柯春旭教授撥冗參與論文口試，並給予許多寶貴的指導與建議，使我獲益良多。另外，謝謝許多學長給予我在研究上的討論與建議，還有其他在「人與機器實驗室」的夥伴們：木政、一元、豪宇學長，又勳、哲儒、怡康、博翔同學以及佑綸、宗仁、怡翔學弟們，在這兩年的實驗室研究生活中，由於你們的陪伴讓我的生活更多采多姿。最後要感謝的是我的家人以及我的朋友們，你們的關懷與支持使我能心無旁騖的完成學業。這兩年將會是我這輩子很難忘的一段日子，謝謝大家。

目　　錄

表　目　錄

圖　目　錄

第一章

導論

隨著電腦科技與網際網路的快速進步，各種資訊可藉由電腦網路彼此互相流通交換，舉凡網頁瀏覽(web browsing)或者電子郵件(e-mail)等，幾乎成爲現代人們生活中不可或缺的一部分。在網路多樣化的應用中，網路的遠端應用即是其中一個相當重要的部分，例如爲了讓身在遠端的機器人和人一樣可以適時做出適當的決策， Goertz 與 Thompson 於 1954 年提出遠端操控(teleopration) [17]，將人類與機器人兩者的優點結合，既可讓人類的感測與操控能力延伸到遠端，又可以使操縱者處於安全的地方來進行操作；另外，在 1986 年由 Jaron Lanier (the founder of VPL Corporation) 首先提到的虛擬實境(Virtual Reality , VR) [11]，其發展也從實驗室走入一般大眾的日常生活與工業應用，例如 3D 的線上遊戲、網路博物館、房屋仲介的線上看屋系統、玩具製造公司的建模模擬等。

當進行遠端應用時，需要將各種資訊回傳到主控端給操縱者，像是視覺、聽覺、觸覺，甚至是嗅覺及味覺等感覺資訊，使操縱者可以感受到遠端環境的變化情形，將遙不可見的遠距景物清晰地呈現在使用者的面前，此項技術稱爲遠端呈現 (Telepresence) [5]。遠端呈現這個名詞最早在 1983 年由 Akin 等人 [5] 定義，其目的是用於太空科技上，以幫助太空人在外太空作檢修的工作，其定義如下：“在執行任務端，操縱器 (manipulator) 具有靈活性，能讓使用者完成一般人類的動作。而在控制端，使用者接收足夠感覺資訊，並提供一種彷如實際身在現場的感覺”。而所謂的足夠感覺資訊，就人主要的感官感覺來討論，一般以視覺、聽覺、以及觸覺爲主，所以主要的人機介面裝置也是用來處理影像、聲音、與力量方面的資訊。虛擬實境發展之初以立體影

像 (stereoscopic image) 爲主，接著加入環場音效 (surround sound) 以增加臨場效果；而發展較晚的觸覺回饋，早期應用在遠端操作 (teleoperation) 上，用來復現遠端機器人在順應性工作下的力資訊情況，協助操作員控制遠端機器人完成任務 [8][19]。但爲了讓人們能單純地從獲得刺激到更進一步發展與虛擬的環境互動，以期營造出更具真實的虛擬世界，由於力感是最直接與環境互動的介面，所以觸覺回饋裝置漸漸受到重視，在 1960 年代後即被大量應用於虛擬實境的回授刺激上 [1][10]。而對於加入力覺或觸覺回饋裝置後的典型遠端操作系統，呈現於圖 1-1 中，圖中人類操作者於近端對力回饋搖桿進行操作、搖桿將獲得的資訊交給電腦運算處理，電腦運算後即再將控制命令下達給遠端機械臂的驅動裝置、以驅使機械臂依循控制命令執行動作，而機器臂上的感測裝置不斷地把目前機械臂的位置與感受到的受力傳回電腦處理，電腦將其一部份轉換成視覺資訊顯示於螢幕、另一部分交由力回饋裝置施行力感回饋。

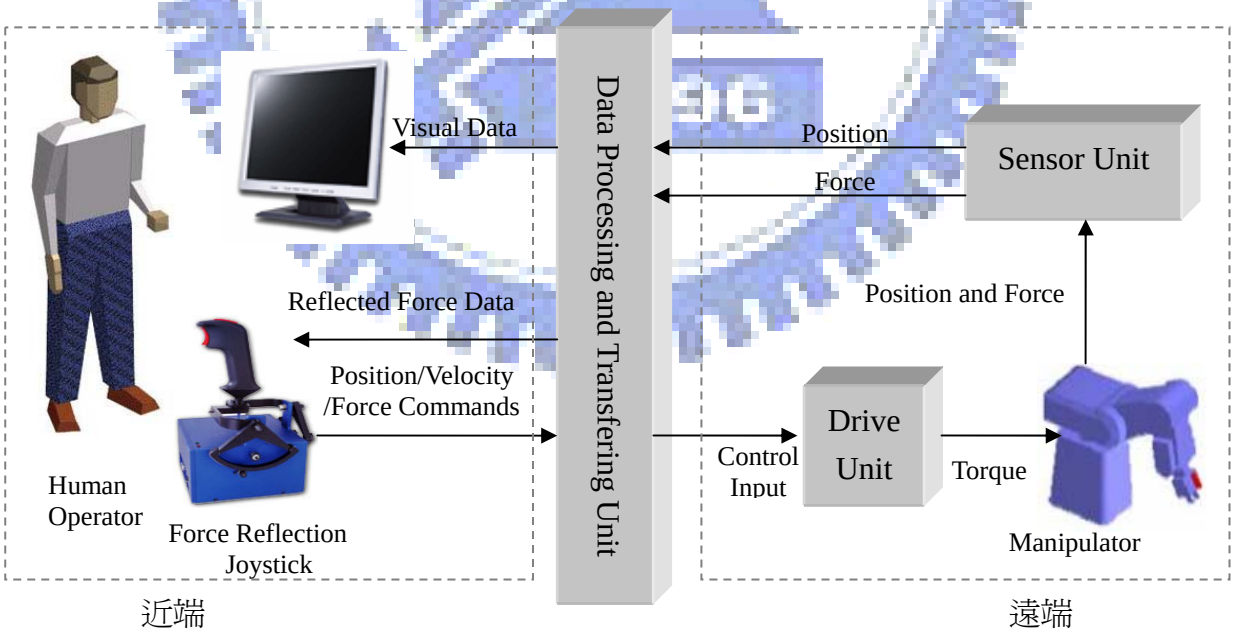


圖 1-1　典型的遠端操作系統示意圖

遠端操作系統與操作者之間的即時互動，必須透過雙向互動的力感裝置方能達到緊密配合的效果，力資訊經由力回饋搖桿的轉換，產生觸感然後傳送到操作者手上。基本上觸覺介面種類繁多，如搖桿、手套以及滑鼠等，彼此間因任務的不同而各有適合的觸覺設備。在目前的遠端呈現，聲音與影像已經發展多時且技術上有很多的突破，然而對於力感的直覺式呈現方面，除了工業界或是需要精密操作的遠端環境外，一般大眾常接觸到的力感、或說是觸感的呈現方式，極大多數仍停留在有方向或無方向的固定式震動居多，其力感、觸感表現大多很粗糙，而原因是因爲取得設備的價格昂貴，使得普及推廣有一定的難度；但隨著科技的不斷進步，各式資訊裝置的成本已逐漸低廉，在預期硬體可以大眾化的情況下，未來力感的精緻化與普及化將會是必然的潮流，像是立體雙聲道幾乎已完全取代了單聲道一樣，例如在 2002 年，英國倫敦大學學院與美國麻省理工學院這兩地的科學家，透過網際網路成功示範的「隔洋握手」(Virtual hands reach across ocean) [22]，在未來相信會是很一般、直覺式的網路互動行爲，所謂的「隔洋握手」，如圖 1-2 所示，即雙方研究員透過小型機器臂 Phantom 隔洋合力把一個電腦驅動的方型盒子抬起來，能夠清楚感受到對方的推拉力道，並從電腦螢幕了解彼此是如何移動方形盒子。但上述的隔洋握手提及到的力資訊與觸覺的流通，是建立在每秒 10Mb 的網路傳輸速度上，所用的是光纖或高速寬頻網路，而非一般家庭用的網際網路，但在可預期的今後，一般家用網路傳輸必定會朝比此更高的傳輸速度邁進。

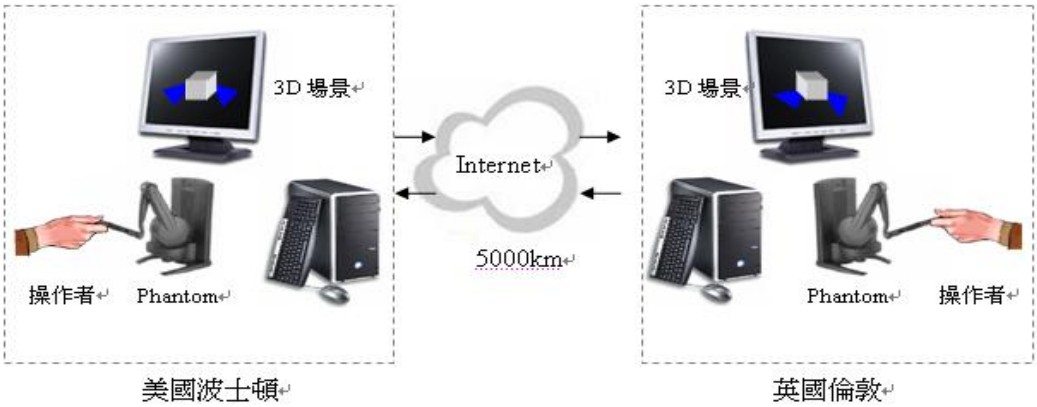


圖 1-2　隔洋握手操作示意圖

如前述的互動行爲，若欲推廣至大眾，勢必要讓軟體與硬體皆可輕易的安裝。故本論文的目的，爲發展一可攜性的力感系統，讓使用者可以輕易地帶著力感裝置至不同的地方安裝，經由電腦網路與在遠方的其他使用者即時互動，並且能操作各自的力回饋裝置，一起在同一個虛擬環境中進行活動。而對於目前已發展的設備，諸如力回饋搖桿 (force-reflection joystick)、感應手套 (Sensing glove)、空間球 (Spaceball)、3D 滑鼠 (3D mouse) 以及感測衣 (Body suit) 等，這些雖都可呈現細膩、正確方向與大小的的觸感、力感，但以可以爲大眾輕易取得、操作易懂、建置容易、且技術較成熟爲前提下，本文中將選用力回饋搖桿作爲發展實現力感的工具。在決定好了力回饋裝置與系統的建置後，我們也著手進行了合作搬運的測試，了解對於合作搬運此一應用上，可能會遇到的問題，並就遇到的問題提出討論且解決之。

本論文將於第二章介紹所建置的可攜性力感系統，說明本實驗所架構的互動式遠端操作系統，包含基本架構與系統組成；第三章依據說明如何利用系統中的力回饋裝置，建置並復現虛擬世界中的力感；第四章則是說明如何在軟體上建構虛擬物體；第五章利用所發展的力回饋系統進行合力搬運實驗；最後，第六章提出結論以及未來發展的方向。

第二章

遠端操作系統

爲了能夠傳達遠距離間人與人觸覺般的力感受，增進兩方間更爲真實的感受，以達到及時互動性的呈現，我們發展了互動式即時力呈現遠端操作系統，藉由網路的主從式(Master-Slave)架構，設計了 Master 系統與 Slave 系統，分別代表了近方與遠方操作系統，兩方操作者只要藉著此遠端操作系統，即可做到簡單的即時力互動，圖 2-1 顯示此遠端操作系統組成方塊圖。本章首先介紹此系統的主從式架構，並且對系統架構概略性的介紹，接下來討論完整的系統架構，並描述相關的軟體設計。

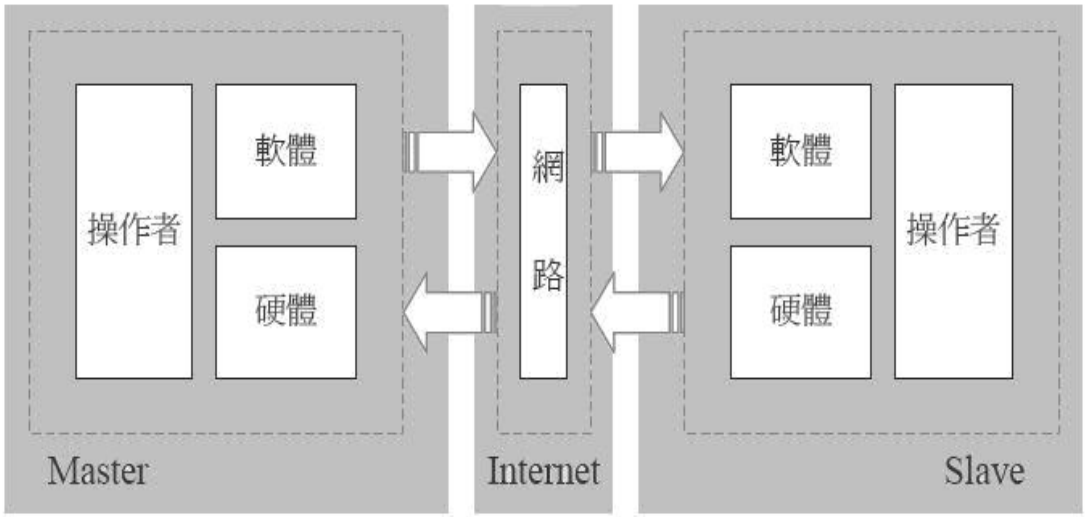


圖 2-1 遠端操作系統方塊圖

2.1 系統架構

在系統的建置中，因爲要連結來自不同地方的力回饋裝置，使其能在相同的模擬環境中，故需透過網路來達成。而對於各種常見的遠端系統，Server-Client架構是最廣泛運用的網路通訊模型，又稱之爲Master-Slave主從式架構，其概念爲Client先對Server端提出要求，而後Server對Client所提出的要求提供適當的服務，接著將結果再傳回Client端；如圖2-2所示，當多個Client連結至一個Server，由Server端進行各Client端提出的運算要求，再將結果送回Client端，此架構的好處是，可以確保Server端所在的模擬環境是針對所有的資訊進行處理，此模擬環境可以維持穩定。其缺點就是當模擬的環境很複雜且Client端眾多時，Server端運算會相當繁重而耗費大量時間造成Client端無法即時反應。如果採取分散式的系統架構，即沒有明顯的Server與Client之分，任一台主機(操作端)都是Server、也同時是Client，則整體系統的設計難度將會提高很多，要達成各操作端的同步性是更加困難。所以爲了模擬環境可以維持穩定我們採用Server-Client主從式架構，而解決即時反應的方式則是將模擬的環境各別由Server端和Client端自行運算，網路也只需傳輸少量的資訊就可達到同步。因此本論文以Server-Client架構爲發展，在Server端及Client端建置虛擬的環境，中間藉由網路連線，使各個Client端操作者可以藉力回饋裝置活動於虛擬環境。而Server與Client端所各自負責的工作，在下面章節將討論。

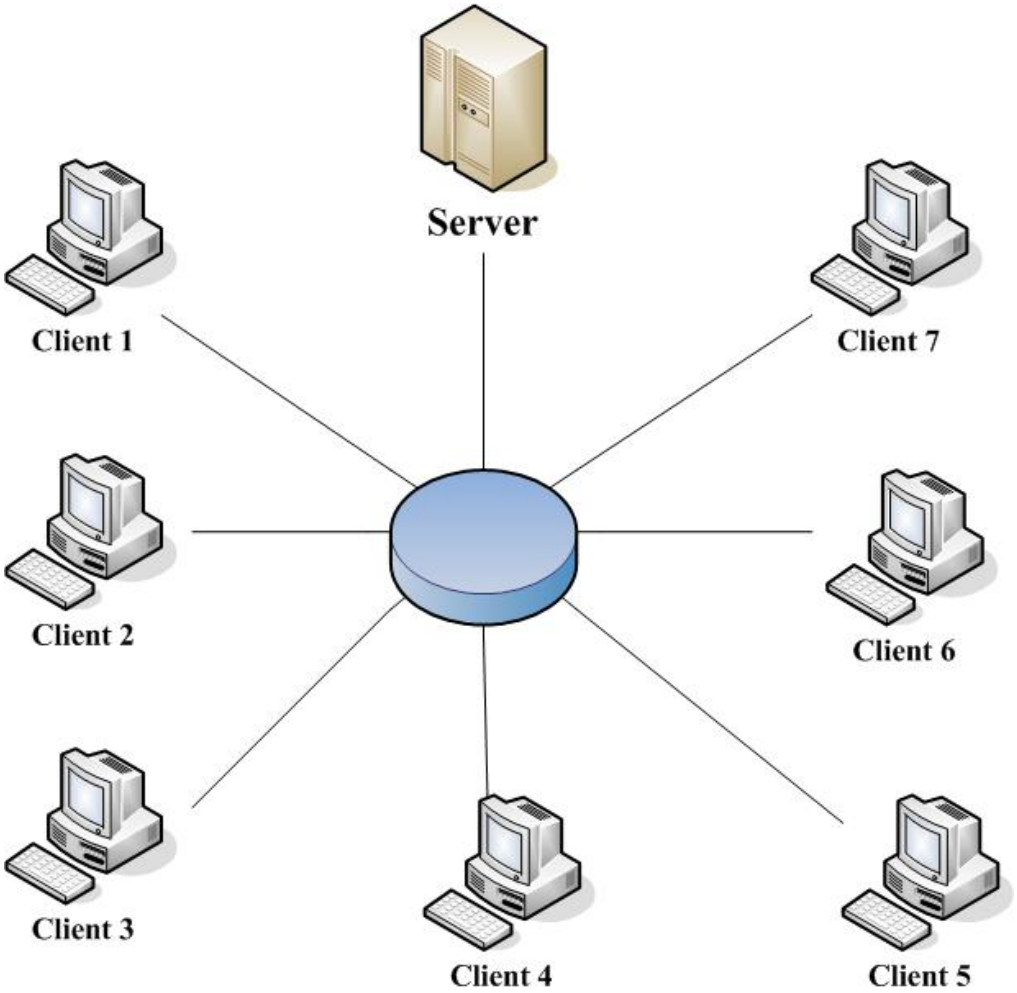


圖2-2　Server-Client 架構圖

2.1.1 Client 端

Client端的操作者對搖桿進行操作，並對Server端提出連線要求，發送Client端目前搖桿的位置資訊至Server端，同樣的 Client 端也會接收到Server端的搖桿位置資訊而在Client端有一對應於Server端搖桿位置的虛擬機械臂(Manipulator)，當Client端與Server端接觸虛擬物件時，會對虛擬物件產生作用力，Client端的操作者會很快感應到物體的回饋力，因爲力的資訊是在各別的主機上計算。我們可在圖2-3看到Client端接收與傳送資訊示意圖

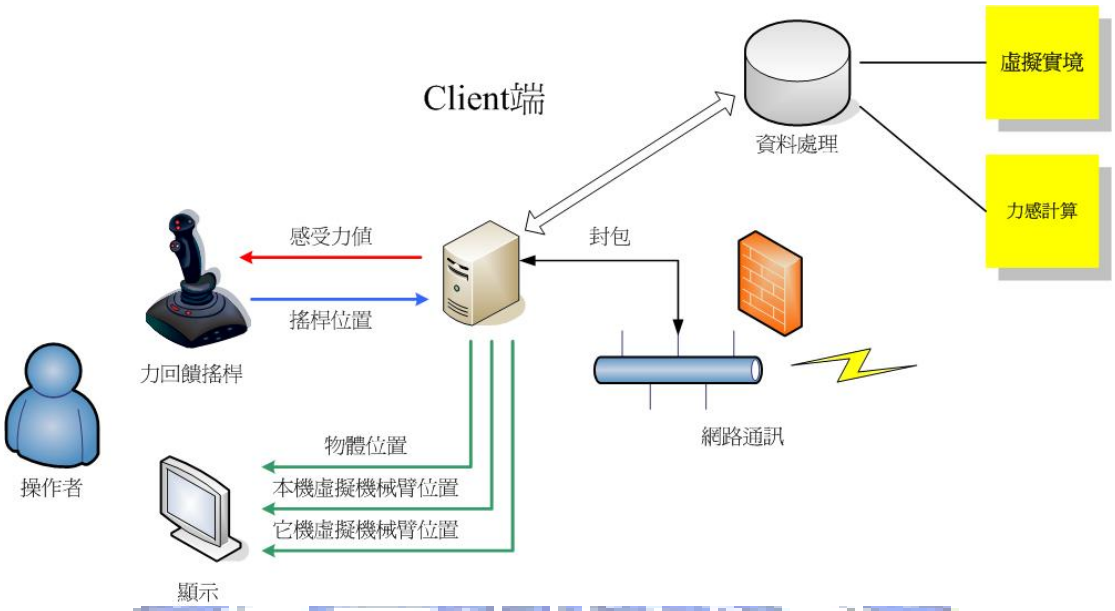


圖2-3　Client 端接收與傳送資訊示意圖

2.1.2 Server 端

在Server端方面，就架構功能而言，負責自身任務要求需要的運算與接受來自Client端的搖桿位置資訊並提供本身搖桿位置資訊至Client端並提供相關資訊讓Sever與Client能同步。而系統中其決策流程如圖2-4所示，依據遠端Client端所傳來的位置資訊與自身搖桿位置進行判斷是否有對模擬物件產生接觸事件，一旦發生接觸時立即進行力資訊運算，得出作用於物體的力資訊與應回饋於Sever端的回饋力，並同時負責計算施與物體之力資訊對物體產生的反應。另外在圖2-5 Server端系統示意圖中，因爲在Server端電腦也裝置了力回饋搖桿，使得Server端電腦本身亦扮演了Client端的角色，在圖中顯示了要接收處理與要發送的資訊。

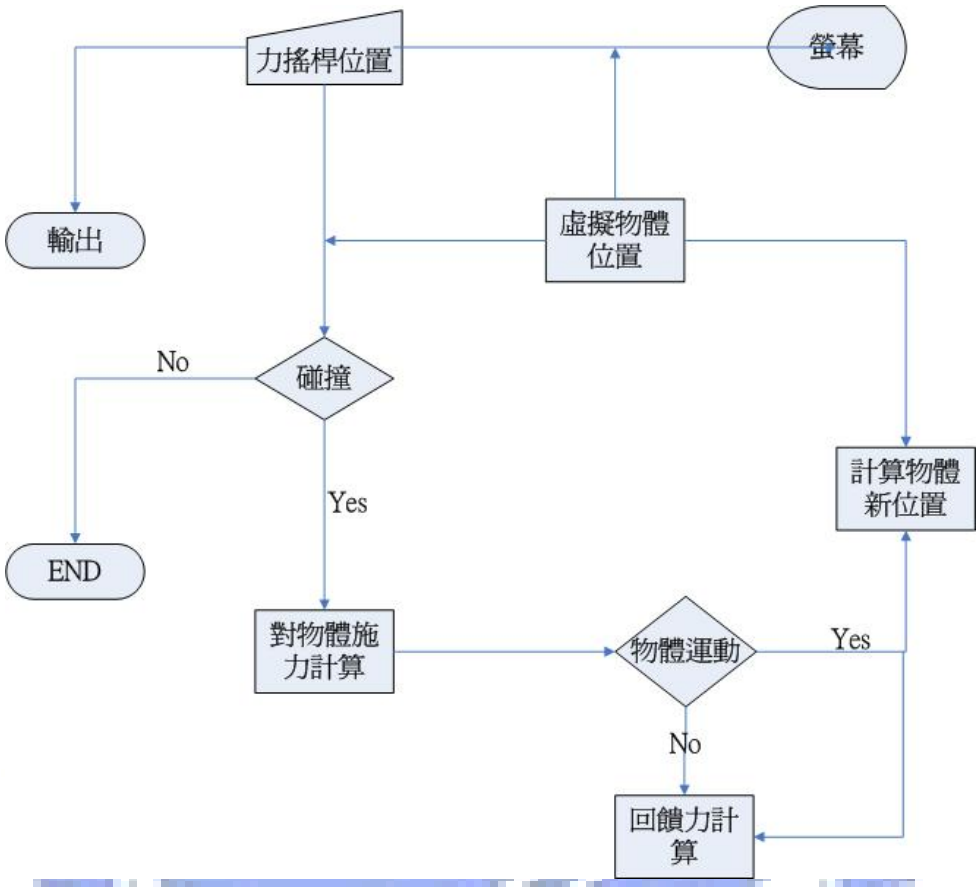


圖2-4　　Server 端系統決策流程圖

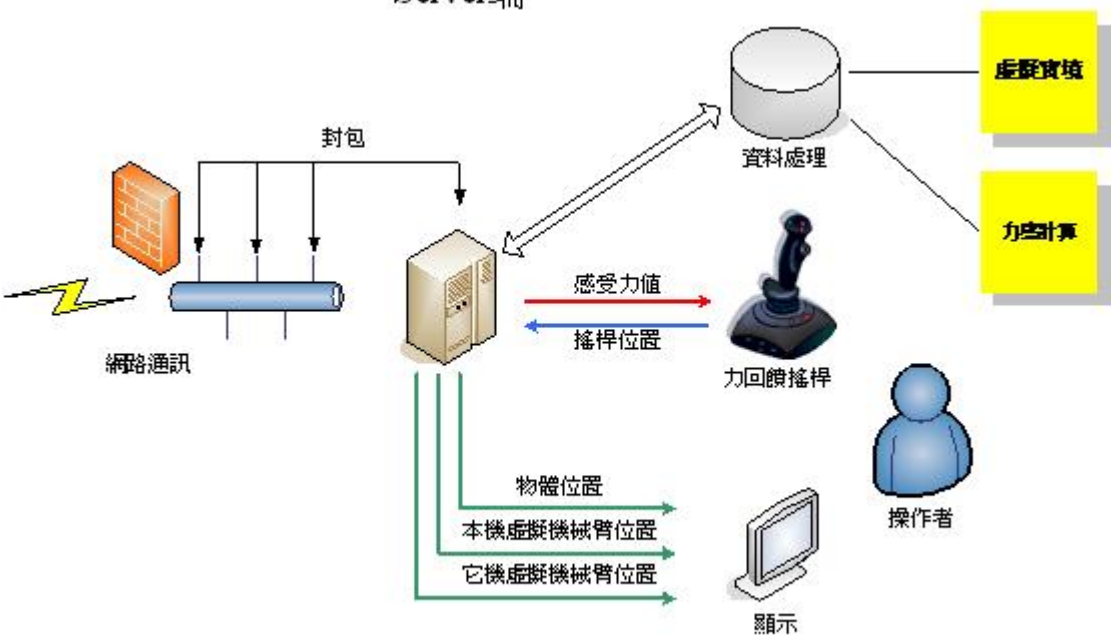


圖2-5　　Server 端系統示意圖

2.2 網路傳輸

本節首先討論一遠端系統在網路上會遇到的訊號延遲問題，接著介紹實作系統網路部份時，所使用的通訊協定，以了解此系統的傳輸方法。

2.2.1 時間延遲

資料在網路上傳輸會有時間延遲的現象，典型通訊延遲主要由三種部份所組成，第一部份爲傳輸時間(transmission time)，第二部份爲傳播延遲(propagation delay)，也就是信號從電路上一點傳播到另一點所需的時間，又稱網路延遲(network latency)，第三部份爲處理延遲(processing delay)，也就是從請求信號發出到請求信號被處理所需的時間，又稱排隊延遲 (queuing delay)，如圖 2-6 所示。

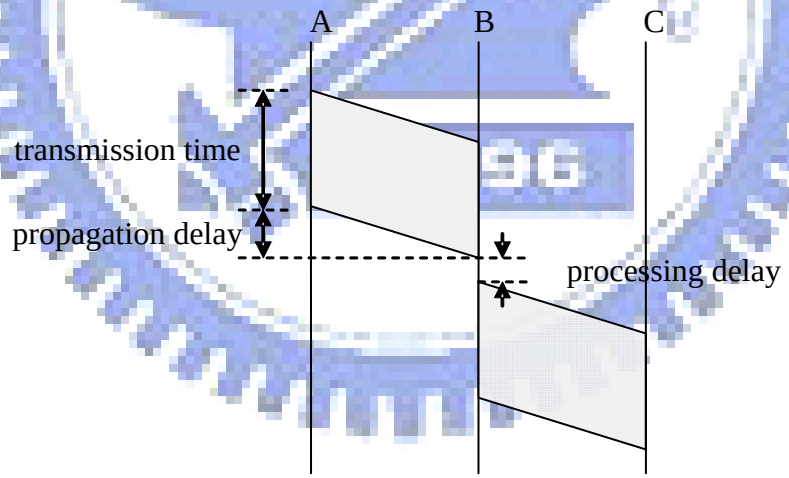


圖 2-6　通訊延遲示意圖

而遠端操作一個完整來回的操作所需花費的時間，包含了以下四個部份，分別爲由遠端系統(Robot)傳回給近端操作者(Operator)的網路通訊延遲 t_1，操作者作決定所造成的時間延遲 t_2，將控制命令傳給遠端系統的網路通訊延遲 t_3，以及遠端系統執行動作所花的時間 t_4，將以上所延遲的時間加起來就是一個迴圈

所需的全部時間，如圖 2-7 所示。

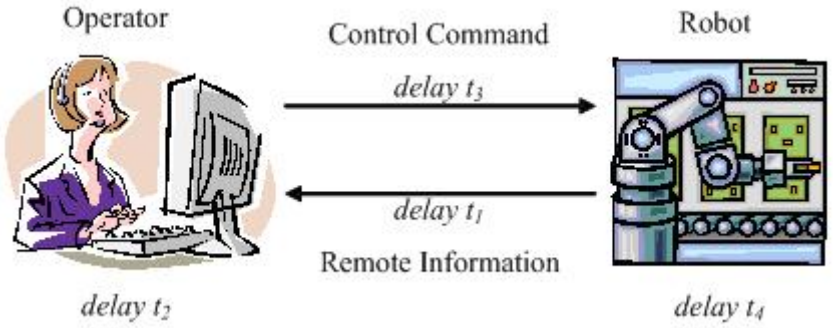


圖 2-7　遠端操作往返一趟所需時間示意圖

典型遠端操控系統雙向訊號流程圖，如圖 2-8 所示，圖中下標 m 代表 Local site，d 代表 delay，h 代表操作者，e 代表遠端環境，x 爲位置資訊，v 爲速度資訊，f 爲力資訊，t 爲時間，$T_1(t)$和 $T_2(t)$則分別代表由 Local site 端傳到 Remote site 端以及 Remote site 端傳到 Local site 端的傳輸時間延遲函式，各訊號之間的關係，如下所示：

$$x_{md}(t) = x_m(t - T_1(t)) \tag{2.1}$$

$$v_{md}(t) = v_m(t - T_1(t)) \tag{2.2}$$

$$f_{hd}(t) = f_h(t - T_1(t)) \tag{2.3}$$

$$f_{ed}(t) = f_e(t - T_1(t)) \tag{2.4}$$

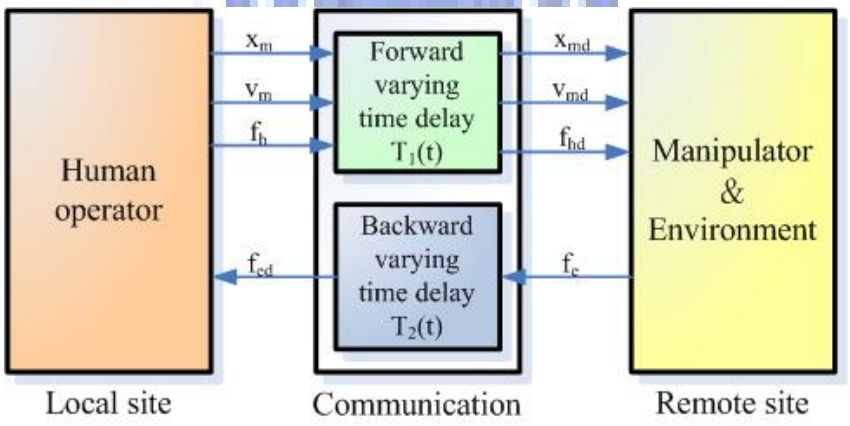


圖 2-8　雙向遠端操作系統訊號流程圖

我們使用的訊號流程首先由 Client 端透過網路傳送 Client 端搖桿位置到 Server 端系統，Server 端也傳送 Server 端搖桿位置到 Client 端，彼此互相計算各別的力反應。另外，因爲網路上存在路徑選擇等問題，並且網路上時間延遲的現象是時變的，最後我們將討論網路上時間延遲對我們系統有何影響。

2.2.2 通訊協定

我們在實作裡使用WinSock來建構網路通訊介面，再藉由TCP/IP通訊協定與Ethernet來達成Server端Client端的連接。而TCP/IP通訊協定並不是只有TCP/IP，實際上它是很多協定的組合，TCP/IP協定家族是由TCP、UDP、IP、ARP、RARP、ICMP等所組成的，以下稍微對IP、TCP與 DP作介紹：

IP (Internet Protocol) / 網際網路協定：

IP位於網際網路層，主要工作是切割封包與選擇封包傳送的路徑，由於IP只負責將資料傳送到目的地，而不作任何錯誤檢查與控制，因此爲非可靠性傳輸，需要靠上層的TCP作偵錯的動作。

TCP (Transmission Control Protocol) / 傳輸控制協定：

TCP位於傳輸層，採用連接導向模式，具有資料接收確認回應與兩端相互維護封包順序號碼之特性，爲可靠性傳輸，適合用在需要可靠傳輸而不希望資料傳輸錯誤的地方。

UDP (User Datagram Protocol) / 使用者資料傳施協定：

UDP 位於傳輸層，採用無連接模式，對於發送端送出的資料封包並不具有順序號碼，接收端也不會有所回應產生，雖然較不可靠，但與 TCP 比較具有較

少的額外負擔(overhead)，此外，UDP還有可作多點投射(multicasting)與廣播(broadcasting)的好處。

2.3 網路連結實作

WinSock是Windows Sockets一詞的縮寫，由Marting Hall在1991年所提出的，並且在1993年，一個公開的視窗網路程式發展介面Windows Sockets 1.1版正式發行。WinSock並不是指認何具有實體的程式或軟體，而是指一套公開在Microsoft Windows下發展網路程式的應用程式介面(Application Programming Interface，API)，一般而言，當我們在寫程式時需要依靠寫好的函式庫以及系統呼叫來達成我們的目的。視窗環境即藉應用程式介面提供了這樣的服務，其定義了若干個函式，包括每個函式的呼叫方式、回傳資料型態、變數個數及函式功能等，程式設計者按照這個規格，使用這些函式以取得系統服務。總之，Winsock應用程式介面即是位於TCP/IP與應用程式間的介面，使得在視窗環境中的網路應用程式與TCP/IP傳輸協定核心間的溝通具有統一的規格，如圖2-9所示。

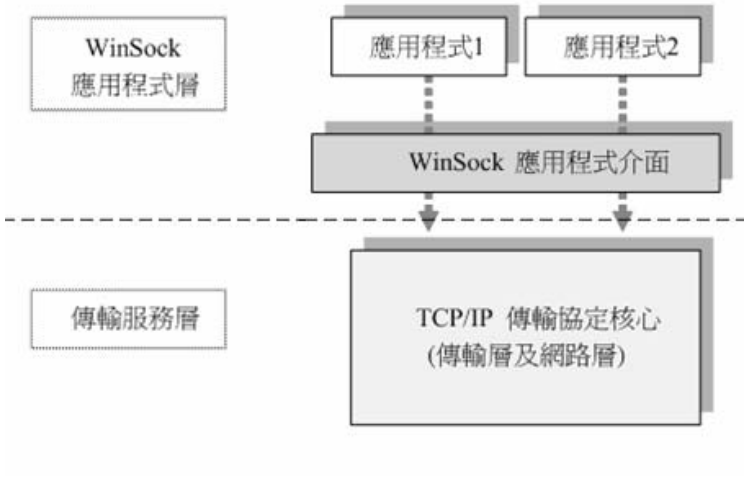


圖2-9　WinSock 應用程式介面

在網路的實作上，我們採用無連接式(UDP)的Server-Client模式，如圖2-10所示，首先呼叫WSAStartup()函式向作業系統要求使用WinSock的動態函式庫，接著在兩端各開啓一個socket，socket可抽象地想像成資料的傳輸口，利用這個傳輸口和遠端的機器互相傳送資料，並指定本地IP位址和port number給未定名的socket，等一切設定指定就緒後，Client端透過socket將資料傳送給Server端，待Server端接收完成後，將所收到的資料進行計算與處理，再透過Server端的socket將資料傳回給Client端，Client端將接收到的資料再次進行計算與處理，如此即完成一次完整的傳送與接收迴圈，之後按照這樣的迴圈反覆下去，直到操作結束，而當要中斷兩端點的連線狀態時，Server端與Client端均需利用closesocket()函式關閉socket，並且呼叫WSACleanup()函式註銷使用WinSock的動態函式庫。

資料在兩端點傳輸時，WinSock的資料輸出入模式可分爲三種。第一種爲阻擋模式(Blocking Mode)，在此種模式下的函式呼叫，會等到該函式作用完成後才返回函式呼叫點，其優點是直接易懂，缺點是易造成系統停擺；第二種爲非阻擋模式(Non-blocking Mode)，在此種模式下的函式呼叫，不論函式執行完成與否均會立刻返回函式呼叫點，其優點是程式不會浪費太多時間在等待，缺點則是需要利用輪詢方式檢查該函式是否執行完成，增加程式的複雜度；第三種爲非同步模式(Asynchronous Mode)，在此種模式下，函式於呼叫後，在尚未執行完成時，即會立刻返回函式呼叫點，讓應用程式繼續執行，當該函式真正執行完成時，系統會向應用程式發出訊息，通知該函式以執行完畢。三種模式各有其優缺點，基於本系統運作機制與同步性的考量，我們採用阻擋模式進行資料的傳輸，以確實掌握程式運作流程。

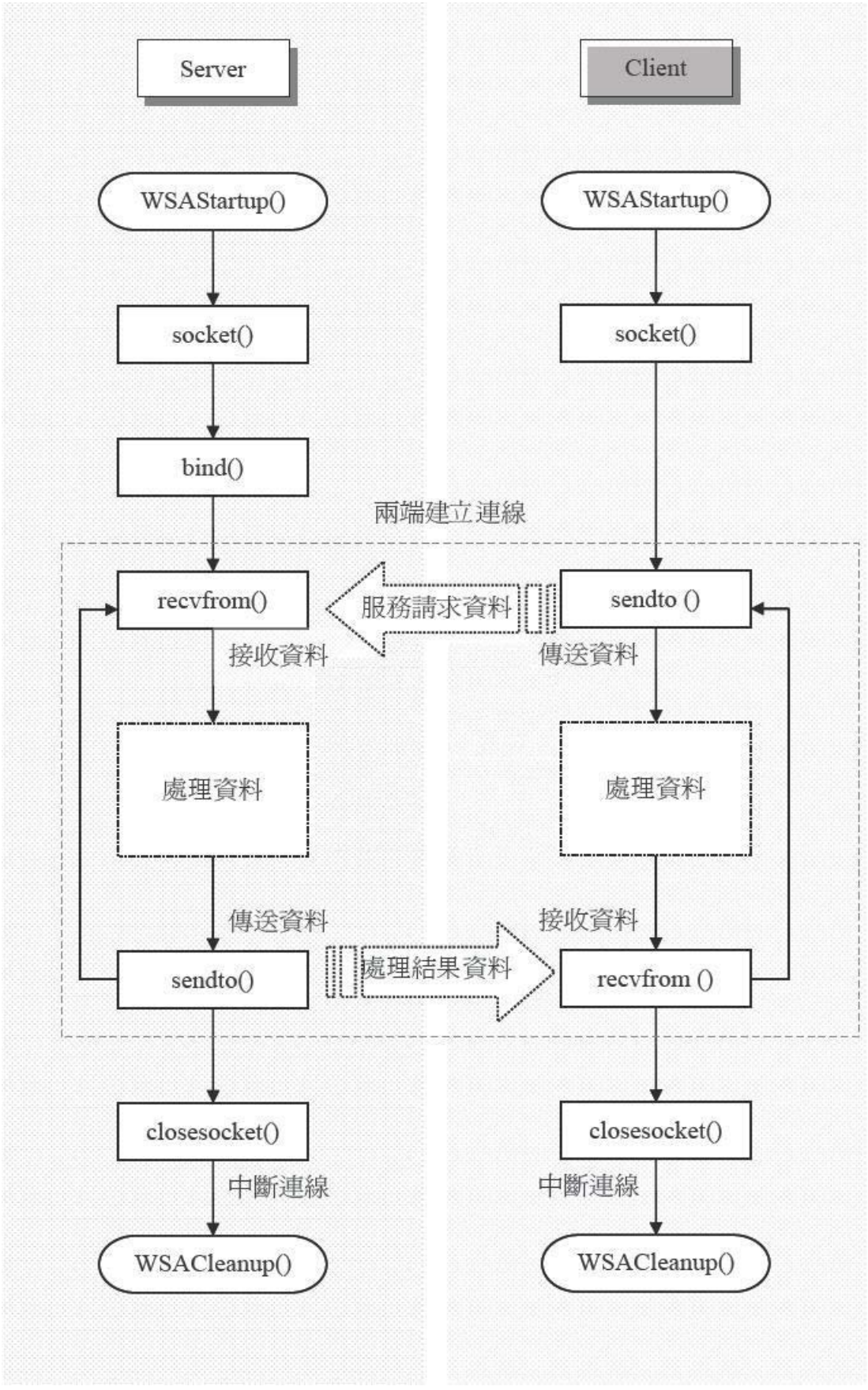


圖 2-10　無連接式(UDP)的 Client-Server 程式流程圖

2.4 多執行緒

力的更新頻率與視覺的更新頻率有所不同，而如果將力的更新頻率與視覺的更新頻率調爲相同，即將視覺更新頻率拉高到與觸覺相同，則會因圖形處理耗去電腦硬體的大量計算，使得電腦執行效能大幅下降；所以不把電腦的圖形處理和觸覺處理在同一個執行緒裡頭運行，而以多執行緒 (multithread) 的方式進行，將兩者分別獨立運作。

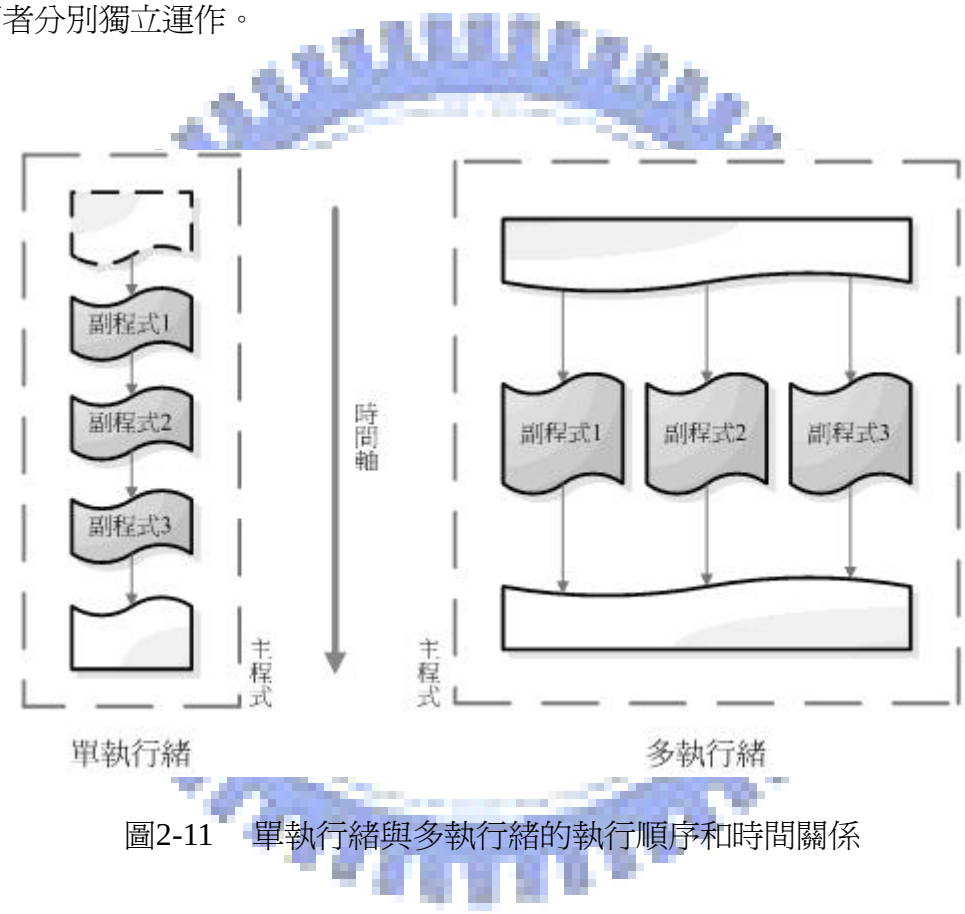


圖2-11 單執行緒與多執行緒的執行順序和時間關係

多執行緒能夠同時提供多個執行緒 (thread) 的功能，通常執行緒所指的是在一個程式中的副程式，多重執行緒將一個程式分割成幾個小模組，每個模組都在一個特定的執行緒當中執行，這就宛如平行處理一般，但是卻利用了多工的能力，表面上使用者並未感覺到多重管線的運作方式，但是實際上程式執行的效率卻提高了。舉例來說，單執行緒就好像大型量販店只有一個結帳出口，如果顧客採買的東西只有一點點，那麼就可以很快速的結帳，但如果採買的東西很多，則

結帳的時間就可能很久，而其它欲結帳的顧客就必須等待，而多執行緒就好像有很多的結帳出口，就算一個出口停頓了很久，也不會影響到其它的出口，如此可大幅的提升運作效益，圖2-11顯示一般程式設計 (單執行緒) 對照多執行緒的執行順序和時間關係。此外，利用多執行緒的技術來設計程式，不但可提升效率，更可以簡化程式架構，在本論文中，即採用雙執行緒，整個系統的流程如圖2-12所示 (在此以 Server 端爲例)，區分爲兩個運作迴圈，分別處理力回饋與視覺呈現，利用此雙執行緒來作爲程式內部的運作方式。

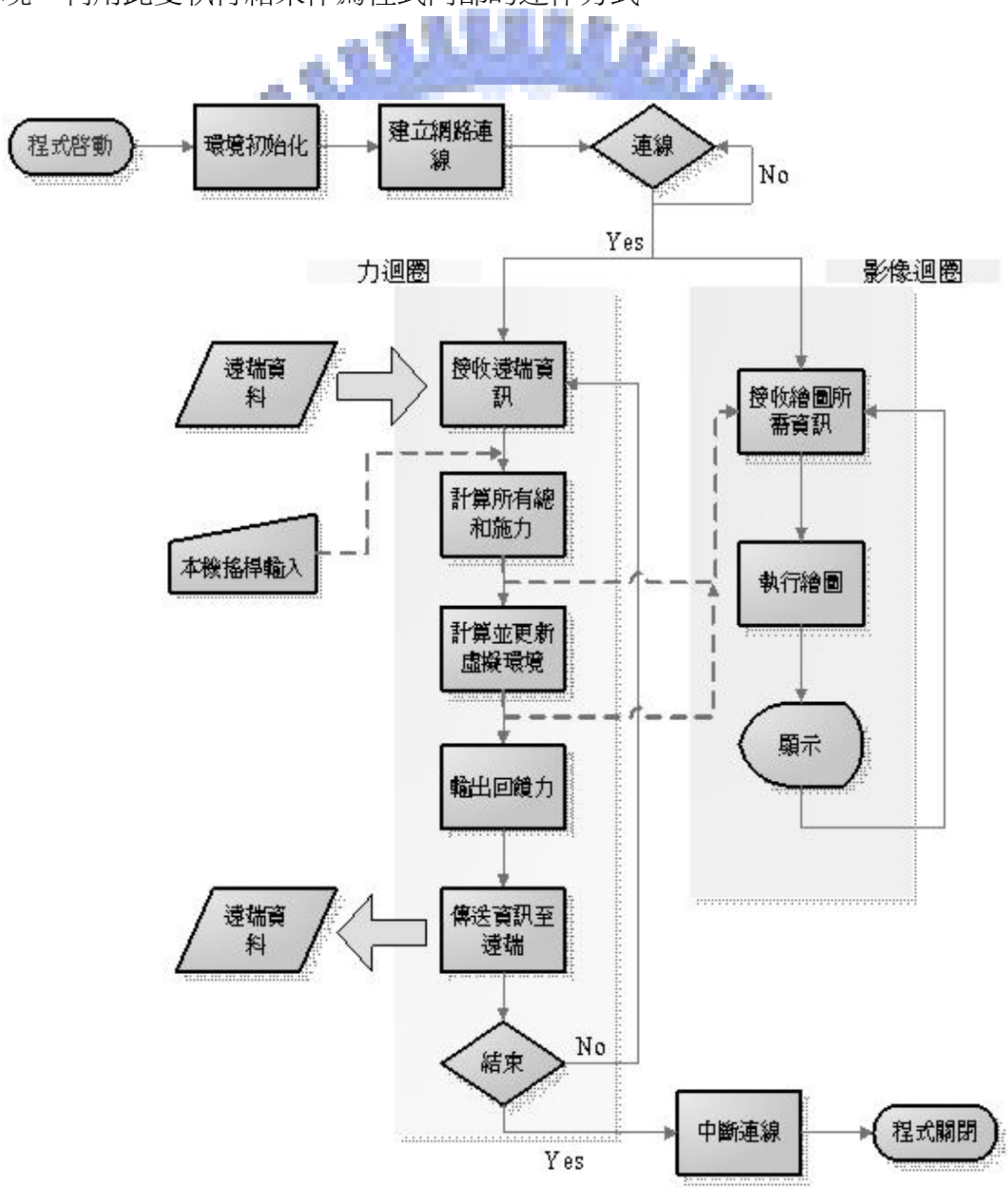


圖2-12　Server 端雙執行緒程式流程圖

第三章

力感系統實現

在第二章所提計劃建構的系統中，有一個不可或缺的環節，亦即操作者用來進行操控的的力回饋裝置，本章首先介紹一個多人的互動式遠端力感系統，根據力的合作、對抗的數種可能性，探討其互動關係，以建立力感系統的基本架構；接著討論如何使用力回饋裝置以達到力感復現，再決定所要使用的力感系統模型，最後我們將討論如何建立我們的可攜性的力感系統。

3.1 力互動模式

力的互動方式大致可以分爲兩大類：合作與對抗。合作就像是雙手合力一起搬運某個物件，其好處是減輕單手施力的負擔，爲的是更易於完成工作；對抗就好比兩個人在比腕力，爲的可能是樂趣，也有可能是爲了阻止對方的危險、不適當地動作。合作的議題一直是被廣泛討論的，像是以一機器手臂控制多個機械臂進行合作搬運的控制方法 [9]，或像是常用在人與機器手臂的互動上的 Impedance control [22] 與 Compliance control [17]等，每當有不同的任務需求，就會發展出不同的控制方式因應。在以”合作”爲主要目的的前提下，當然希望彼此的施力不會產生互相對抗的情況，因此前面提到的控制方式，其作法大多是盡量去削減可能會受到的對抗力量；但在直覺情形下，如果兩個人位於一物體的不同側，其中一人施力於物體使其推向另一人，另一側的人應很自然地感受到物體被推過來的壓力，而如果經過前面的控制方式修改，可能另一側人感受到的推力會不自然，因此在建立系統時，無論是爲了以後的發展性，還是現階段的通用性，

應該將模擬的環境依自然界的物理法則建置，如此則可以隨意切換進行力的對抗或合作。

3.2 力回饋裝置

計劃建構的力回饋系統其運作的流程如圖 3-1 所示，系統接受來自使用者的施力，經力回饋系統將運動指令送至虛擬場景單元，虛擬場景單元將場景資訊送至力資訊運算單元，力資訊運算單元將控制訊號送至力回饋裝置，再回傳給使用者。

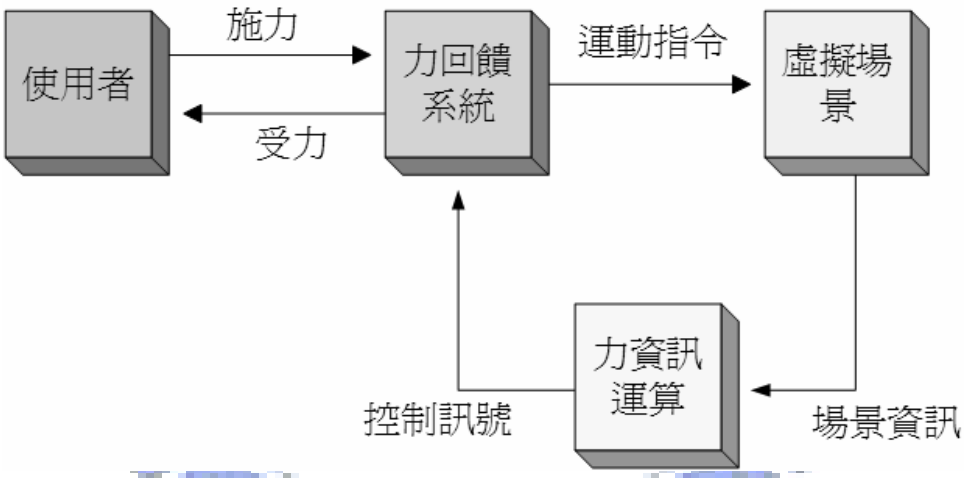


圖 3-1　力感操控系統控制流程

而對於直接和使用者進行互動的力回饋裝置，其基本組成如圖 3-2 所示，在力回饋裝置內有著不斷感測目前資訊的感測元件，還有驅使裝置本體 (如搖桿本體) 將力量傳回操作者以達成力回饋功能的動力元件。

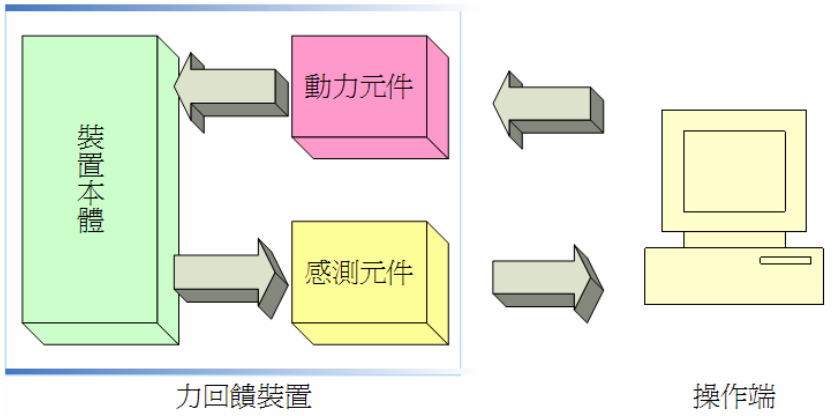


圖 3-2　力回饋裝置示意圖

多種的力回饋裝置以及其感測裝置，大致可分爲主動式和被動式兩種，兩者皆是當環境有反應力回傳時，會把力感回饋給操作者感受，所以就回饋力感的功能性而言，兩者無異，其最大的不同，是在於施力方面，主動式力回饋裝置隨時偵知操作者肢體的出力情況，而被動式裝置卻僅在與環境的物體接觸後，才會在該環境中產生施力，換句話說，主動式力回饋裝置能主動感知操作者施力大小和方向，就算施力者的施力肢體處於自由運動狀態、或是與環境處於靜止對抗狀態(比如推物體推不動，但其實手還是在輸出推力的情況)，皆能知道施力者的施力狀況，而要達到如此的隨時主動探知，其裝置上的力感測元件必定要隨時與施力的肢體保持接觸才行，整體的概念是利用現今發展的肌電訊號直接量測肌肉出力大小，再搭配專執力回饋的力回饋裝置來構成整套系統。而至於被動式的力裝置，相較於主動式裝置的複雜性，大多只含有位置的感測裝置，只能感知裝置的輸出位置，或是移動速度，而不能隨時感知施力者究竟出了多大的力量，也因此僅在操作者接觸遠端物體後，才能決定究竟施了多少力在遠方環境，但如果在被動式的裝置上加上力感測裝置 (force sensor)，也可以量測到施力靜止對抗狀態時的出力資訊。整體來說，被動式裝置因爲建置較爲容易，且精確度容易控制，如果再搭配力感測裝置、或是加速度感測裝置就可以滿足大部分的使用需求。而本論文以我們實驗室現有的兩個被動式力回饋裝置做爲發展工具。

使用的兩個裝置分別是 Immersion 公司的 Laparoscopic Impulse Engine 與 Impulse Engine 2000，如圖 3-3 所示，其中 Laparoscopic Impulse Engine 是五個自由度的操控器，而 Impulse Engine 2000 則是兩個自由度的操控器。而其硬體規格與一般目前世面上的低價力搖桿的比較表列於表 3-1，由於 Laparoscopic Impulse Engine 的架構與自由度較不同，以不同方式列於表 3-1 之右。

(a)Laparoscopic Impulse Engine　　(b)Impulse Engine 2000

圖 3-3 力回饋裝置. (a)Laparoscopic Impulse Engine 和(b) Impulse Engine 2000

項　　目		Impulse Engine 2000	市　售 Microsoft & Logitech	Laparoscopic Impulse Engine
活動範圍 (θ)	X 軸	65 *deg*	45 ~ 55 *deg*	自由度 5 (可動部份) 3 (力回饋部分)
	Y 軸	65 *deg*	35 ~ 50 *deg*	
總高度(L3)		28 *cm*	18 ~ 30 *cm*	工作空間 5” x 9” x 9” (12.7 x 22.8 x 22.8cm)
握把長度(L3-L1)		18.5 *cm*	13 ~ 20 *cm*	搖桿精確度 0.0009” (1100 dpi) (0.023 mm)
擺動距離(R)		18 *cm*	10 ~ 16 *cm*	最大額定輸出力矩 2lbs. (8.9 N)
最大額定輸出力矩		8.9 *N*	not available	摩擦力 ＜0.5 oz. (0.14N)
搖桿精確度		0.0009” (1100 dpi) (0.023 mm)	not available	工作頻寬 650Hz (線性軸) 120 Hz (旋轉軸)
電腦連接介面		dedicated card	USB、game port	
定位方式		free	fixed	
工作頻寬		650 *Hz*	not available	
控制方式		PC	μ-processor	
價位		180000 *NT*	1000 ~ 5000 *NT*	

表 3-1　力回饋搖桿規格比較

3.3 觸覺頻率

在討論如何將力感系統實現前，我們先討論關於人的特性以及力感與觸感兩者的不同，以利於建立力感模擬系統時可以知道力回饋裝置所需要的工作頻寬，如此所復現的力感，在感覺上才會連續。當人類與物體發生碰觸時，首先遍及皮膚表面的感覺受體會感受到接觸力，藉由神經系統將此感覺資訊傳回給大腦，讓大腦得知碰觸的實際情況後，人腦可對肌肉下達指令以作出適當的反應。而人體感覺器官接收外界刺激的能力，在視覺方面頻率約爲 50 Hz [20]，而 [18] 中提到能讓使用者感到互動流暢的 3D 電腦動畫最低標準爲 15fps(frames per second)，電影般的程度約爲 24 fps，電腦動畫標準門檻值爲 30 fps，但這些值在某些場合還是不夠的，在多數軍用或商用的飛行駕駛模擬應用程式中，要達到真正的流暢需達到 60 fps，一但超過了 60 fps 以上，肉眼已經很難分辨出之間的差異；而人耳朵聽到的頻率約在 20 – 20k Hz 之間 [20]。

人的觸覺 (haptics) 大概可以分爲力感 (force) 及觸感 (touch/tactile) 兩個部分，力感部分的感覺器官 (receptor) 較低頻約爲 20 – 30 Hz，位在人體內較深處，而力感對力解析度約爲 0.5 N [6]；觸感部分的感覺器官較爲高頻約 320 Hz [20]，位於接近皮膚表面；一般而言，力感感覺器官處理低頻且高振幅的力資訊，而觸感感覺器官處理高頻低振幅的觸感資訊。

頻寬及解析度等參數值除因各文獻量測的對象及方法會有些差異外，還會隨使用者運動的方式不同而有所改變，比如說人的觸感系統在精神集中的情形下，可以感覺得到的振動有時可以高達 1k Hz [10]，甚至可高達 10k Hz [7]，皆遠比先前提及的 320 Hz 要高出很多，很明顯地這些參數會隨應用環境而有所變動，所以在發展觸覺回饋 (haptic feedback) 裝置時，要依其應用場所需的規格來設

計，例如醫生開刀所需的觸感回饋 (touch feedback) 裝置需要高達 1k Hz 才能提供足夠的觸感 [4]，隨操作者的手越快速移動則需要更高的更新頻率 。故在大多數的狀況下，力更新頻率遠較影像的更新頻率 30 – 60 Hz 高出很多，此時若觸覺和影像使用相同的更新頻率，則在復現力感上會有不真實與不連續感。歸納視覺與觸覺所需的更新頻率至表 3-2，依此數據在建置系統時，我們將系統的視覺更新頻率訂在 30fps 以上，力感更新頻率 30Hz 以上。

項目	電影播放	電腦動畫	飛行模擬
視覺更新頻率	24fps	30fps	60fps

項目	力感	觸感
觸覺更新頻率	30Hz	1k~10k Hz

表 3-2　系統模擬時視覺/觸覺所需相關參數

3.4　力感復現

對於力感復現，應該先了解搖桿的系統模型，才能知道如何決定搖桿的施力大小，而介紹完搖桿系統模型後，我們將在第四章接著討論如何建立虛擬場景，並在模擬環境建造一件虛擬物使搖桿能對該物進行接觸。

3.4.1　搖桿系統模型

操作者所感受到的力量主要是靠力回饋搖桿馬達輸出的機械功率所賦予，不

同的力覺感受係由不同的力量大小、方向與頻率等因素組合而成，當使用者操縱搖桿時，施予搖桿力量的手會與搖桿產生互動，其中包含位置、速度、加速度與力量彼此間的動態關係，可以以阻抗來表示， Hogan 在 1987 年提出阻抗控制理論 (Impedance Control Theory) [14][15]，用來處理機器手臂碰觸物體表面所發生的狀況，我們以此爲基礎，將其概念應用在力回饋搖桿與遠端環境兩者雙向的互動力感關係上。

首先定義在操作者所處的真實世界中，搖桿系統本身的運動性質，令系統本身的彈性係數爲 k，阻尼係數爲 b，質量係數爲 m，搖桿的位置爲 x，速度爲 v，加速度爲 a，系統的摩擦阻力爲 f_{fr}，使用者施加的力量爲 F_h，馬達出力爲 F_m；接著相對應於真實世界的搖桿，定義用在電腦模擬的虛擬世界中，代表虛擬搖桿的物理性質，令模擬物件的彈性係數爲 K，阻尼係數爲 B，質量係數爲 M，搖桿的位置訊號偵測值爲 X，速度爲 V，加速度爲 A，虛擬的摩擦阻力爲 F_{fr}，使用者感受的力量與施加的力量同爲 F_h，根據 [11][12] 中的系統模型，在真實操作端中的搖桿存在以下的阻抗關係式：

$$F_h + F_m = ma + bv + kx + f_{fr}\frac{v}{\|v\|} \qquad (3.1)$$

而在虛擬世界裡我們期待能達成類似(3.1)式的虛擬阻抗關係式：

$$F_h = MA + BV + KX + F_{fr}\frac{V}{\|V\|} \qquad (3.2)$$

圖 3-4 即是顯示此兩種關係的示意圖，左邊是真實世界操作端搖桿的系統組抗圖，可以看出當施力於搖桿上時，所施的力量與代表搖桿本體的物體質量 m 間會有一彈性彈簧與阻尼模擬接觸時的情形；而在電腦端運行的虛擬世界中，即圖右邊，也有相當的關係存在於施力者和虛擬端的搖桿質量之間。假

設量測到的物理量與實際值相等，即：$x = X, v = V, a = A$，結合 (3.1) 和 (3.2) 兩式，可得到馬達輸出力的方程式：

$$F_m = (m - M)a + (b - B)v + (k - K)x + f_{fr}\frac{v}{\|v\|} - F_{fr}\frac{V}{\|V\|} \qquad (3.3)$$

其中 m、 b、 k 和 f_{fr} 爲系統常數，理論上先輸入需要模擬的虛擬質量 M，虛擬阻尼係數 B，虛擬彈性係數 K，以及虛擬摩擦力 F_{fr}，只要能夠量測到位置 x，速度 v，以及加速度 a，就可藉由不同的阻抗元件及其組合，調變馬達的輸出力 F_m，模擬各種物理元件。

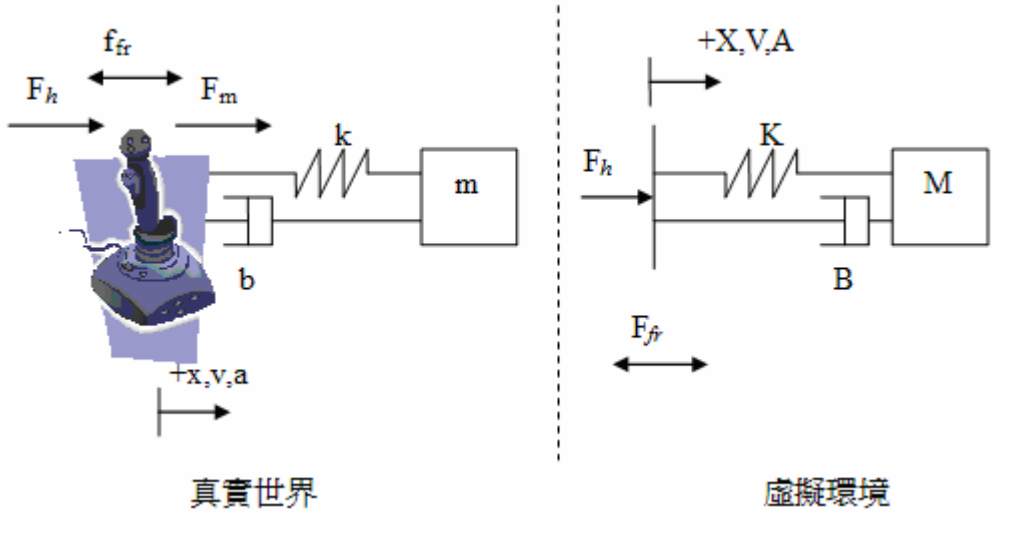


圖 3-4　真實世界和虛擬環境之系統阻抗圖

另外，若加速度不易量得 (需要高精度的加速規)，也可利用加裝在搖桿上的力感測器，直接量得的搖桿與手之間的作用力 F*h*，則馬達出力會有如下關係式：

$$F_m = \left\{\left(\frac{m}{M} - 1\right)F_h + bv + kx + f_{fr}\frac{v}{\|v\|}\right\} - \left(\frac{m}{M}\right)\left(Bv + Kx + F_{fr}\frac{v}{\|v\|}\right) \qquad (3.4)$$

若是在模擬中，欲直接以真實端之搖桿去感受模擬物體的特性，可將模擬物件的質量直接設定成搖桿系統的自然等效質量 (M＝m)，代入(3.4)式：

$$F_m = \left\{ bv + kx + f_{fr} \frac{v}{\|v\|} \right\} - \left\{ Bv + Kx + F_{fr} \frac{v}{\|v\|} \right\} \tag{3.5}$$

則力感測值並不會影響馬達的力量輸出，只單純的分成真實世界的物理力量與虛擬世界的虛擬力需求的差值。值得一提的是，假設把(3.5)式後半部設爲零，相當於希望搖桿模擬處在一完全不受力的虛擬環境之情況，使用者將感受到搖桿平順無阻力的移動，此時馬達需額外出力以克服搖桿本身阻力，其出力爲

$$F_m = \left\{ bv + kx + f_{fr} \frac{v}{\|v\|} \right\} \tag{3.6}$$

但因爲在此我們所使用的力回饋裝置，其摩擦力並不高，阻尼也不明顯，所以可以忽略掉(3.5)式前半部，即只要專注於(3.5)式後半部，即可模擬出接觸的物理元件性質。

3.5 可攜性力感系統

最近幾年因電腦網路的快速發展，人們對遠端呈現的要求也越來越高，單純的影像和聲音已不能滿足現在的人還需要加上力感，但現在大多數的力感呈現方式還停留在有向或無向的固定式震動居多，所以本節希望能建置一個可攜性的力感系統，連結現有的力搖桿裝置並能透過 TCP/IP 網路通訊協定，使遠端及近端的系統能互相溝通，網路的部份已於第二章提及，本節僅討論如何達成可攜性。

什麼叫可攜性?可攜性這個詞在一般的文宣廣告非常常見，一般都認爲可攜性就是體積不大可以帶著隨處跑，但這樣的定義太模糊了所以在這節中我們定義一下，本篇論文所認爲的可攜性，如下所示：

1. 可攜性裝置能隨意的裝在當前的任何電腦上使用，並不受限於電腦裝置。
2. 可攜性裝置並不止限定在硬體上，也包含軟體。
3. 符合上述任一樣皆可稱爲可攜性。

我們以大家常用的滑鼠爲例，在早期的時候滑鼠並不是跟現在一樣，是列爲電腦的基本配備，在當時要使用滑鼠必須去買一塊滑鼠的I/O卡再將滑鼠接在上面，當時的滑鼠並不具備可攜性，因爲只有滑鼠沒有I/O卡根本不能動作。但是當滑鼠加上I/O卡時就可在當時的任何電腦上運作，只要再配合不同的作業系統(DOS、Linux、Mac OS)給予相對應的驅動程式即可。由這個例子中可以看到一個可以在電腦上使用的可攜性裝置除了硬體外還需軟體的配合，除非這個裝置已列爲基本配備相關的驅動程式皆已內建在作業系統中。

我們使用的力搖桿裝置，在目前來說並不是一個被列爲電腦基本配備的裝置，所以我們要讓這力搖桿裝置能有可攜性的方法就是從軟體下手。我們使用 C++ 的程式語言來編寫我們的程式碼，並將搖桿的驅動程式寫在我們的系統中，這樣只要帶著我們的程式即可在任何電腦上使用。而且在我們所建立的系統中，網路通訊使用 WinSock 的語法，3D圖形則使用 OpenGL ，以上這兩種程式語言加上C++都是目前最廣泛使用的程式語言，即使在不同的作業平台上也可以很容易的編譯出我們需要的程式碼，從而建置一個可攜性的力感系統。

第四章

軟體設計

根據第二章所建構的力回饋系統以及第三章中的力感系統，在本章我們將進行系統的環境設定。爲了能增加遠端呈現(telepresence)的效能，增強系統的整體表現性，我們利用OpenGL來建立互動式場景，以增加互動時的可見性與真實性。我們以合作搬運爲例，如圖4-1所示，其中爲了讓場景中的物體，在接受到外力時，能有接近現實生活物體應有的表現，我們將在這一章先對虛擬場景做簡單的介紹，接著再對如何建立3D物體的模擬做詳細的說明。

圖 4-1　合作搬運示意圖

4.1 虛擬場景

OpenGL是在1990年代初期由SGI公司的IRIX GL標準所演化而來的，目前則當作Unix、Linux、Mac OS與Windows作業系統的2D與3D模型建造，許多遊戲，如Quake，都用到這類技術。SGI控制了OpenGL API的授權，而OpenGL Architecture Review Board (ARB) 是一個獨立委員會，負責這個標準的維護與改

進，並訂定詳細的計畫。OpenGL 1.0版於1992年七月一日發行，最新的是OpenGL 1.4版。此外，爲了因應OpenGL在Windows遊戲上越來越常被使用，微軟另外開發了一套DirectX，專供Windows平台遊戲使用。不過DirectX 的遊戲並不容易移植至其他非Windows 平台使用，所以OpenGL依然會是可攜性與開放標準的良好選擇。當然，市面上的繪圖軟體或函式庫體種類繁多，如OpenGL、WorldToolKit與DirectX等各有其優缺點，而OpenGL因具有可靠性、可攜性以及發展性，是一文件完備的應用程式設計介面，基於這些優點及跨平台的考量，我們採用OpenGL作爲本系統的繪圖函式庫。

實作上，OpenGL 是一個標準的程式庫，我們使用 VC++ 語言在 Windows XP 平台上進行撰寫，並且呼叫 OpenGL 的程式庫。在大多數的平台上，OpenGL 程式庫都伴隨有 OpenGL(utility library) 簡稱 GLU，檔名爲 glu32.dll，此工具程式庫是一組專司經常性工作的函數，如特殊的矩陣運算，或支援各種常用的曲線或曲面。爲了在 Microsoft 的工具中建立使用 OpenGL 的程式和呼叫 DLL 中的函數，必須匯入 opengl32.lib 以及 glu32.lib 這兩個程式庫，並且將程式與匯入程式庫相連結。另外，我們使用了 GLUT (OpenGL utility toolkit) 工具集程式庫，來增加設計時的方便性。圖 4-2 介紹了本研究中關於軟體繪圖程式部份，而在建構虛擬場景時我們利用了以下幾種函式，首先呼叫 glutInitdisplayMode()函式，告訴 GLUT 程式庫建立視窗時要採用何種模式，在此我們採用雙緩衝區視窗以及 RGBA 色彩模式；接著呼叫 glutCreateWindow()函式，用來在螢幕上建立場景視窗；glutReshapeFunc()函式，用來處理當視窗大小改變時新的設定值；glutTimerFunc()函式，用來建立繪圖場景的無窮迴圈，產生具有連續動畫的效果；glutDisplayFunc()函式，告訴 GLUT 當要繪圖時必須呼叫那個函式；最後呼叫 glutMainLoop()函式，用來啓動 GLUT 主要的處理迴圈，包含按鍵、滑鼠、計時器、繪圖與其他視窗訊息，直到程式結束時才會返回。

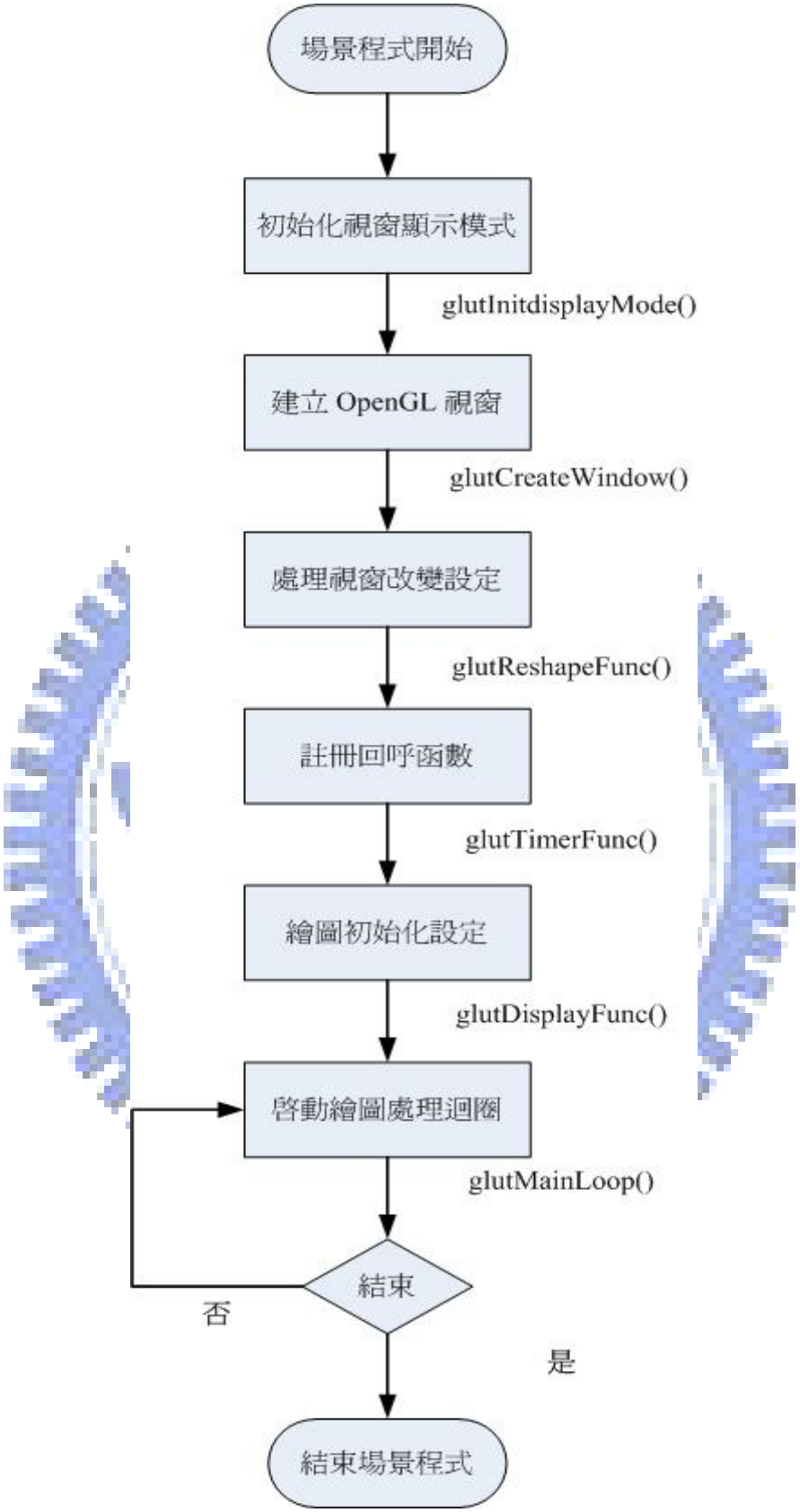

場景程式開始
初始化視窗顯示模式
glutInitdisplayMode()
建立 OpenGL 視窗
glutCreateWindow()
處理視窗改變設定
glutReshapeFunc()
註冊回呼函數
glutTimerFunc()
繪圖初始化設定
glutDisplayFunc()
啓動繪圖處理迴圈
glutMainLoop()
結束
否
是
結束場景程式

圖 4-2 場景程式流程圖

4.2 3D 物體模擬

在 3D 物體的模擬過程中，主要分成幾個部份：

1. 3D 效果呈現：因爲電腦螢幕是二維的平面，我們必須進行轉換，將三維物體的幾何描述轉換成螢幕上的平面影像。
2. 物理模型的建立：在 3D 物體模擬的過程中，我們所要模擬的是現實生活中物體的物理現象，所以需要應用到物理學的原理，使模擬能更真實。
3. 數值積分：因爲我們使用物理模型、運動方程式和微分方程式來對模擬物體進行動作或狀態的運算，所以需要一個數值積分的方法來執行以上的動作。
4. 碰撞偵測和碰撞反應：在一具多物體的 3D 環境中執行碰撞偵測和碰撞反應是非常重要的，但在本篇論文的實驗中主要是進行合作搬運的問題，物件是單一的所以碰撞偵測和碰撞反應主要使用在周圍的虛擬牆壁上，當虛擬物接近虛擬牆壁時就會發生反應。

4.2.1 3D 效果呈現

3D 的物件具有三個測量的維度:寬度、高度以及深度。3D 電腦繪圖實際上也是電腦螢幕上的二維影像，只不過是營造出三維景深的幻覺罷了。首先我們使用透視(Perspective)來呈現 3D 的效果，如圖 4-3 所示，由九條線段及三條虛線所組成的三維立方體，以線條間的角度營造出有景深的感覺。

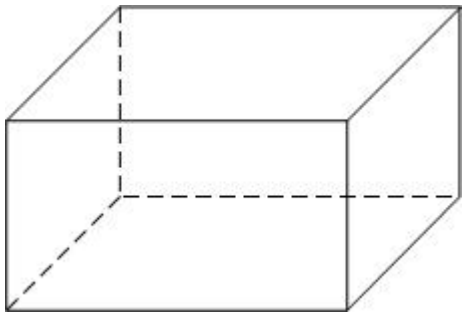


圖 4-3 三維立方體

在三維的空間中，我們使用笛卡兒座標(Cartesian Coordinates)系統來表示物體的位置，現在我們要將三維的笛卡兒座標轉換成可繪製在螢幕上的二維座標，使用的方法是投射(Projections)，如圖 4-4 所示，投射的方法主要有兩種，第一種是正交投射(Orthographic Projections)[7]，我們指定一個立方或長方體爲範圍來進行這種投射，立方或長方體範圍外的任何事物都不會畫出來，而且，所有具有相同維度的物體不管遠近看起來大小都相同，如圖 4-5 所示，這種投射常用在建築設計，CAD(電腦輔助設計)或 2D 圖形上。

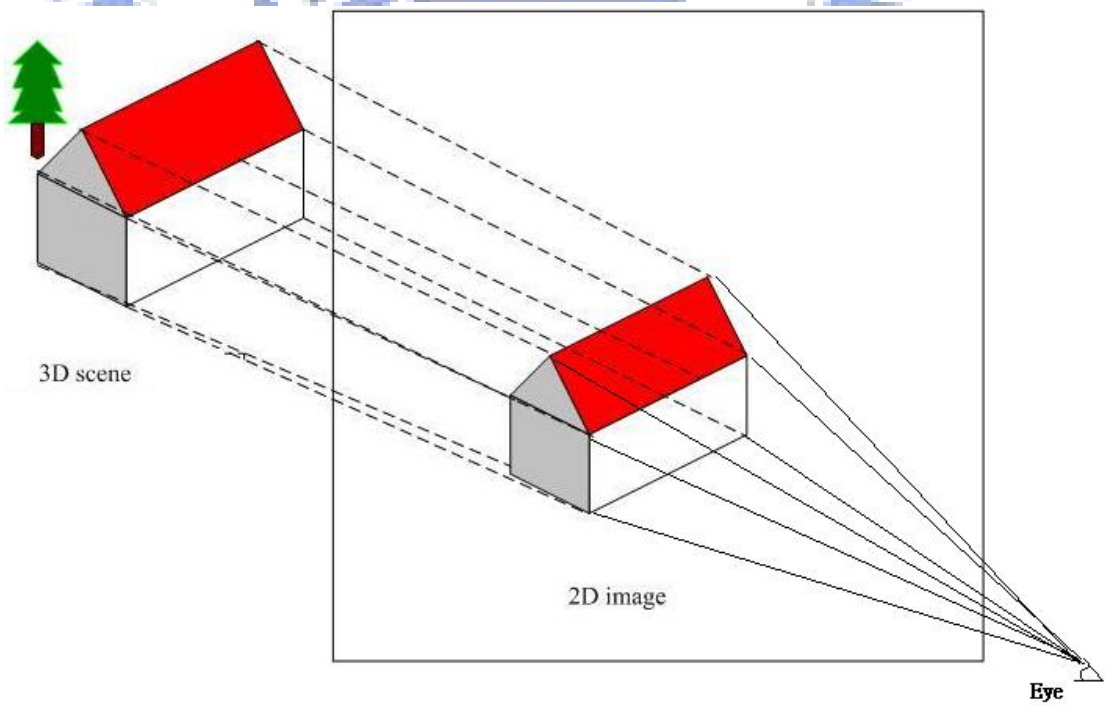


圖 4-4　3D 影像投射到 2D 平面

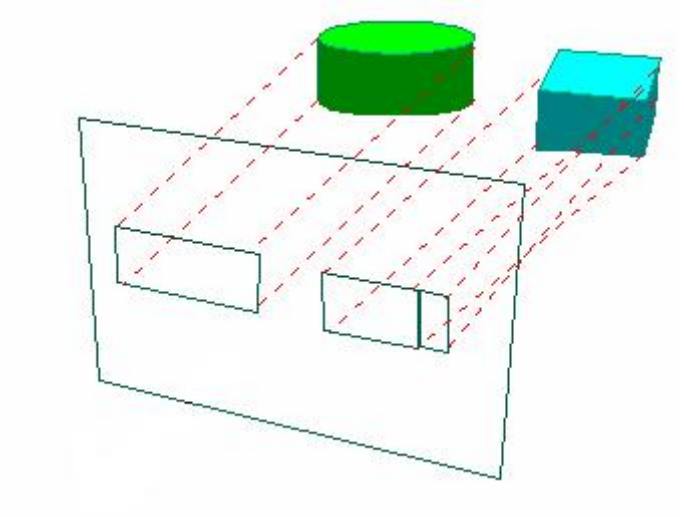


圖 4-5 正交投射

第二種是透視投射(Perspective Projections)法[7]，也是我們實驗所使用的方法，這種投射會使遠方的物體看起來比近端物體更小，如圖 4-6 所示，靠近觀察區域前方的物體看起來大小與實際大小較相近，但靠近末端的物體投射後縮小許多，這種投射方式能爲模擬 3D 的畫面帶來較佳的真實感。而 3D 轉 2D 的座標點轉換程序，如圖 4-7 所示，其中 Modelview 矩陣是用來轉換到眼睛座標，因原始座標可能經過翻轉。接下來眼睛座標會被乘上投射矩陣，產生裁減座標(clipcoordinates)，這會抹去不需出現在觀景窗中的資料，如物體的背面，再來經過透視投射的處理並正規化然後將所得結果給 viewport 以映射到 2D 平面上。

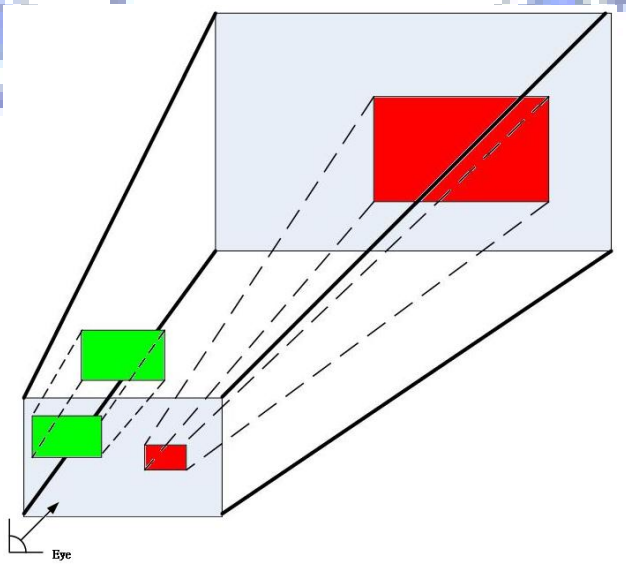


圖 4-6 透視投射

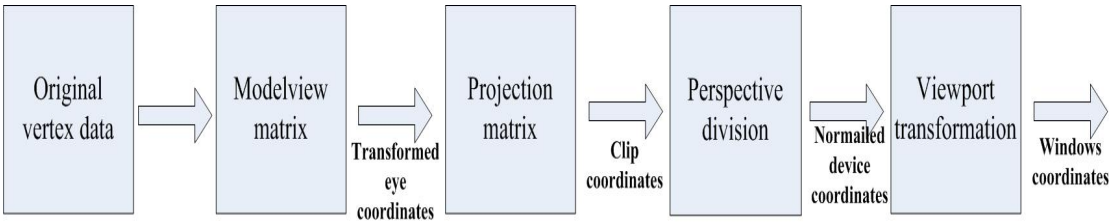


圖 4-7 座標點轉換程序

4.2.2 3D 物體模擬

在 3D 物體的模擬研究中，爲了讓模擬結果能接近現實，最好的方法就是使用物理學的角度來分析物體的行爲反應。在此我們使用了物理學中的運動學和力學等理論[2][3]，接下來我們將介紹在實驗中應用到虛擬物上的物理學理論。

A. 物體作用力

事實上，除了對物體外部施力(接觸力)而造成物體運動外，還有別的力也會影響物體本身的運動，我們稱爲「場力」(field force)。這種力不需實際接觸物體，便能產生作用力在該物體上，如重力、黏滯力等，說明如下：

1. 重力：對於所有在重力場中的質點，都會受到重力的作用，根據牛頓運動定律，重力的大小爲 $F_g = m \cdot g$ ，其中，g 爲重力加速度，m 爲質點質量， F_g 爲重力。
2. 黏滯阻力:黏滯阻力的大小爲， $F_c = -k_d \cdot v$ ，其中， k_d 爲黏滯係數， v 爲質點之速度， F_c 爲黏滯阻力。黏滯阻力的作用在於抵抗質點之運動，便得質點會慢慢停止。適當的阻力有助於加強系統的數值穩定，過量的阻力會造成質點不易運動。

B. 彈簧與阻尼

彈簧是一種結構上的單元，當它連接兩個物體時，會有量值相同而方向相反的兩個力作用在兩個物體上，根據虎克定律，彈簧的公式爲：

$$F_s = k_s(L - r) \tag{4.1}$$

其中的 F_s 是彈力，k_s 是彈力常數，L 是彈簧伸長或壓縮後的長度，而 r 是彈簧靜止時的長度。阻尼(damper)常常與彈簧合併使用，阻尼的作用與黏滯阻力相同，皆爲抵消速度。阻尼所發出的力，與兩連接物體之間的相對速度和阻尼常數 k_d 成正比，相對速度和阻尼力(dampingforce)的關係式如下：

$$F_d = k_d(v_1 - v_2) \tag{4.2}$$

F_d 是阻尼常數和兩連接物體上連接點相對速度的函數，阻尼力的公制單位是 N，而速度的單位是 m/s，k_d 的單位是 kg/s。 彈簧與阻尼通常組合爲單一的彈簧/阻尼單元，如圖 4-8 所示，因此用一個公式就能計算合併後的力。用向量表示時，連接兩物體的彈簧/阻尼單元的公式如下：

$$F_1 = -\{k_s(L - r) + k_d[(v_1 - v_2) \bullet l]/L\}l/L \tag{4.3}$$

在此，F_1 表示物體 1 所受的力，由於物體 2 所受的力與物體 1 大小相同方向相反，因此作用在物體 2 上的作用力爲 $F_2 = -F_1$，其中 l 爲彈簧/阻尼的長度向量，亦即物體 1 與物體 2 間的距離，L 爲 l 的大小，r 爲彈簧/阻尼的原始長度。

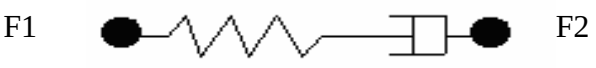


圖 4-8 彈簧/阻尼單元

C. 物體運動學

在物理的運動學中，一般是將粒子和剛體分開來討論，剛體實際上就是一個粒子系統，粒子之間保持固定距離且無相對的移動與轉動，換句話說，剛體移動

時其外形不變，或者外形的變化小到(不重要)可以忽視不計。就剛體而言，其大小與方位是很重要的，且必須同時考慮物體的線性運動及角運動；另一方面，粒子是具有質量的物體，其大小在研究的問題中是可忽略或不重要的，例如，對於拋射體或長程火砲的路徑，分析其軌道時大可忽略其大小。當討論粒子時，其線運動是重要的，而粒子本身的角運動則否。

另外在實驗中我們使用彈簧/阻尼單元來模擬多粒子系統，彈簧可以提供結構力或是附著力，使得粒子間可以緊密地結合(或保持一定的距離)，而阻尼可以使得連接的物體不至於看起來太有彈性。從前面的公式中我們可計算出作用在物體上的力，再依據牛頓運動定律，即可得到加速度、速度與位置：

$$\begin{cases} a(t) = \dfrac{F(t)}{m} \\ v(t) = v_0 + \int a(t)dt \\ p(t) = p_0 + \int v(t)dt \end{cases} \tag{4.4}$$

在(4.4)式中可看出聯立方程式是以連續時間的積分表示，因此，當我們在電腦上作模擬時，必須以數值積分的方法求得上述聯立方程式的解。而我們在實驗中所使用的數值積分方法是尤拉法(Euler's method)，之所以使用尤拉法的原因在於它的計算簡單，尤拉法爲泰勒級數中捨去第一階導數之後的多項式數列，而得到：

$$x(t+\Delta t) = x(t) + (\Delta t)\cdot \dot{x}(t) \tag{4.5}$$

由於尤拉方程式只取泰勒級數的前兩項，第三項以後的部分都被省略掉了，被捨去的部分稱之爲截斷誤差，又稱爲高階項。而把它們捨掉產生了第一階近似值，此估計值的原理就是數列越長，各項的值就越小而對估計值的影響也越小。因爲(Δt)是一個很小的數目，所以$(\Delta t)^2$的值就更小了，依此類推。也因爲出現在分子，所以當數列越高階，值也就越小。

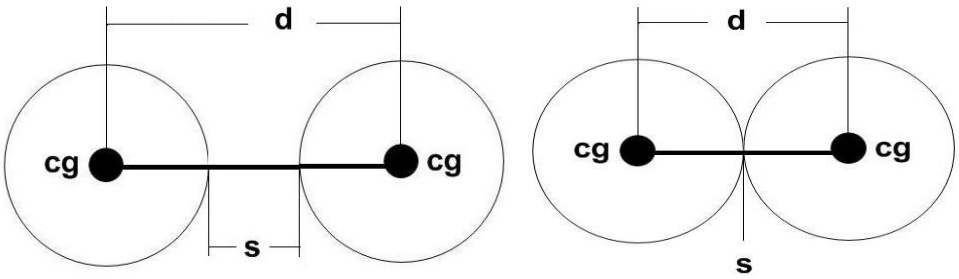
4.2.3 碰撞偵測和碰撞反應

首先我們先將碰撞偵測與碰撞反應作個區別，碰撞偵測是計算幾何的問題，這關係到兩個物體間是否產生碰撞以及產生碰撞的位置。而碰撞反應則是物理的問題，這關係到兩物體碰撞後的運動情形。

A. 碰撞偵測

在兩個物體還未互相靠近的情況下，意味著物體間並未產生接觸。而當兩物體間的距離小於容許值時，就視爲產生接觸的情況。若物體因接觸重疊而移動到對方的內部，則視爲產生穿透的現象。在碰撞偵測的程式中，當發現到物體間的距離小於容許值而構成碰撞的條件時，必須再檢查物體間的速度，以判斷兩物體是靠近中或是遠離中，總而言之，碰撞發生的條件爲：

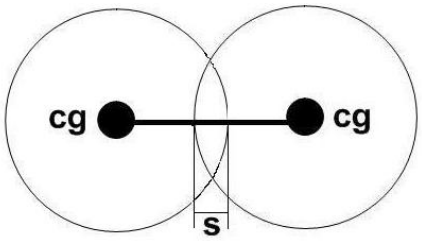
1. 物體間的距離是否小於容許值，如圖 4-9 所示，其中 cg 爲物體重心，d 爲兩物體重心的距離，s 爲兩物體間的距離。
2. 相對法線速度的方向是否向著物體，用以判斷是否物體是互相靠近中，如圖 4-10 所示。



(a) 無接觸　　(b) 接觸

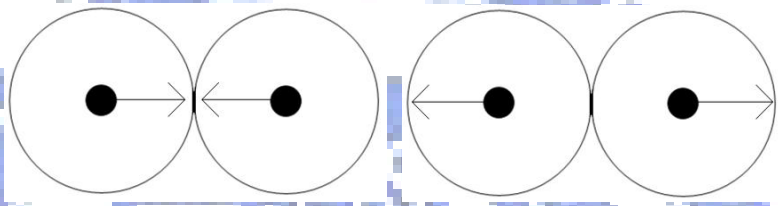
圖 4-9 檢查物體間的距離是否小於容許值：(a) S>0 無接觸，(b) S=0 接觸，

和 (c) S<0 穿透(cont.)



(c) 穿透

圖 4-9 檢查物體間的距離是否小於容許值：(a) S>0 無接觸，(b) S=0 接觸，和 (c) S<0 穿透



(a) 產生碰撞　　(b) 無碰撞

圖 4-10 檢查相對法線速度的方向是否向著物體：(a)產生碰撞 和(b)無碰撞

檢查是否產生穿透現象是很重要的工作之一，如果物體相互重疊，會使得模擬結果看起來很不真實。當發生穿透的時候，必須將時間倒退一個時間間隔，並減少時間間隔在計算一次，直到沒有穿透的情形發生或剛好要碰撞的情形爲止。在我們所進行的合作搬運實驗中，因爲虛擬物體只有一件，我們所要注意的碰撞偵測就是不要讓虛擬物體超過我們所定的工作範圍。

B. 碰撞反應

當我們偵測到發生碰撞時，接下來就是被碰撞的兩個物體應該發生怎樣的碰撞反應，以我們的實驗爲例，由於虛擬物是單一的，會發生碰撞反應只有在 3D

虛擬物超出我們所定的工作範圍。因爲我們在工作範圍的周圍設一道透明的虛擬牆，讓虛擬物體一接近虛擬牆就將它彈回去，讓虛擬物體能一直在工作範圍內。所以在這我們就要考慮一粒子與虛擬牆的碰撞，如圖 4-11 所示，其中$\vec{v}$爲粒子的速度向量，$\vec{N}$ 爲虛擬牆法向量，我們可將$\vec{v}$分成兩個分量，一個是與虛擬牆垂直的分量$\vec{v_n}$，另一個則是與虛擬牆平行的分量$\vec{v_t}$，其值如下：

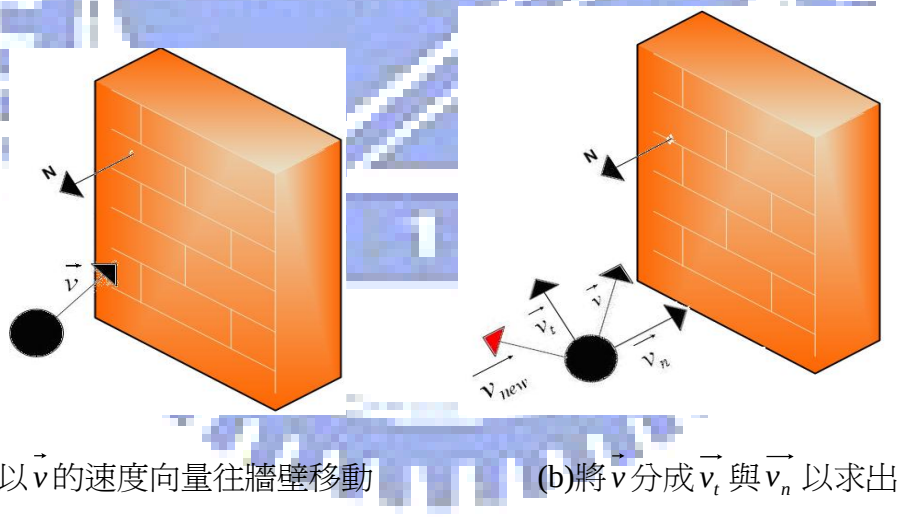
$$\vec{v_n} = \left(\vec{v} \bullet \vec{N}\right)\vec{N} \tag{4.6}$$

$$\vec{v_t} = \vec{v} - \vec{v_n} \tag{4.7}$$

碰撞後粒子的速度爲:

$$\overrightarrow{v_{new}} = \vec{v_t} - k_r \cdot \vec{v_n} \tag{4.8}$$

其中，k_r爲恢復係數(restitution coefficient)。



(a)粒子以$\vec{v}$的速度向量往牆壁移動　　(b)將$\vec{v}$分成$\vec{v_t}$與$\vec{v_n}$以求出$\overrightarrow{v_{new}}$

圖 4-11 粒子與虛擬牆的碰撞：(a) 粒子以$\vec{v}$的速度向量往牆壁移動 和(b)將$\vec{v}$分成$\vec{v_t}$與$\vec{v_n}$以求出$\overrightarrow{v_{new}}$

第五章

實驗

前幾章介紹了遠端操作系統的軟硬體架構，以及所需的相關理論與方法後，在本章我們將以合作搬運的方式進行測試，以了解其在合作搬運中對於力合作與力對抗的資訊是否能執行無誤，因此我們透過網路連結置於不同處的力回饋裝置，在同一場景中進行對虛擬物件的施力測試，藉此印證合作搬運的可行性。本章首先說明實驗的設計方式，以了解實驗的進行，接著對建置的合作搬運開始進行試驗；爲了進一步了解網路延遲對力回饋系統可能造成的影響，我們也進行了實驗室的網路環境與一般家用網路環境的比對，以確實掌握力資訊的流通是否連續無誤，實驗設備如表 5-1~5-2 所示，圖 5-1(a)爲 Server 端操作系統實體圖，圖 5-1(b)則爲 Client 端操作系統實體圖。

Server	
處理器	AMD Sempron 2800+
記憶體	DDR 400 1G
力搖桿	Impulse Engine 2000
作業系統	Windows XP SP2
程式語言	Visual C++

表 5-1　Master 端系統規格

Client	
處理器	AMD k7 1.7Ghz
記憶體	DDR 400 512M
力搖桿	Laparoscopic Impulse Engine
作業系統	Windows XP SP2
程式語言	Visual C++

表 5-2　Client 端系統規格

(a)Server 端

(b)Client 端

圖 5-1 操作系統實體圖：(a)Server 端 和(b)Client 端

5.1 實驗設計

實驗進行的環境，是以一台電腦連接力回饋裝置 Laparoscopic Impulse Engine 作爲 Client 端，另一台電腦連接力回饋裝置 Impulse Engine 2000 作爲 Server 端，藉由網路連結，在同一個 3D 場景中， Server 端操作者及 Client 端操作者對 3D 場景中的虛擬物體進行合力搬運，因本實驗中的 3D 虛擬物是一個軟性的物體，所以當 Server 端操作者及 Client 端操作者對 3D 場景中的虛擬物體進行合力搬運會造成物體的形變其中虛擬物彈性與阻尼係數爲 10 和 5。

合力搬運大致可以分爲兩大類：合作與對抗，在合力搬運遠端操作的實驗中，我們設計了三種狀況來表現合力搬運的合作與對抗，如下所述：

1. Server 端操作者與 Client 端操作者對虛擬物朝同樣方向施力。
2. Server 端操作者與 Client 端操作者對虛擬物朝相反方向施力。
3. Server 端操作者與 Client 端操作者任意一端固定不動，另一端隨意拉扯。

第一種及第二種情形分別用來表示合力搬運的合作與對抗，在第一個及第二個實驗中我們可以知道 Server 與 Client 兩端的操作者都同時在移動，所以第三個實驗我們將一端固定不動，另一端移動，來看它們彼此間的影響，我們可以從以上這幾種狀況驗證本系統在合力搬運中呈現的性能。

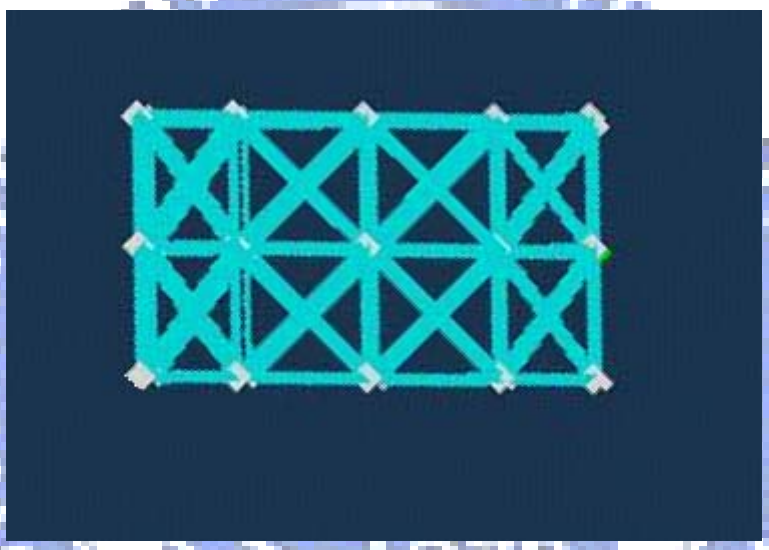
在上述三種實驗中，我們令近端爲 Server 端，遠端爲 Client 端，並配合場景模擬對操作者規劃了三個實驗操作步驟，其步驟內容如下所述：

1. 首先將 Server 端與 Client 端先固定在所要施力的節點上，此時兩端操作者握住搖桿但尚未施力。
2. 接著 Server 端與 Client 端再依據所要表現的狀態來移動並出力。
3. 直到 Server 端與 Client 端移動虛擬物到目的地即停止。

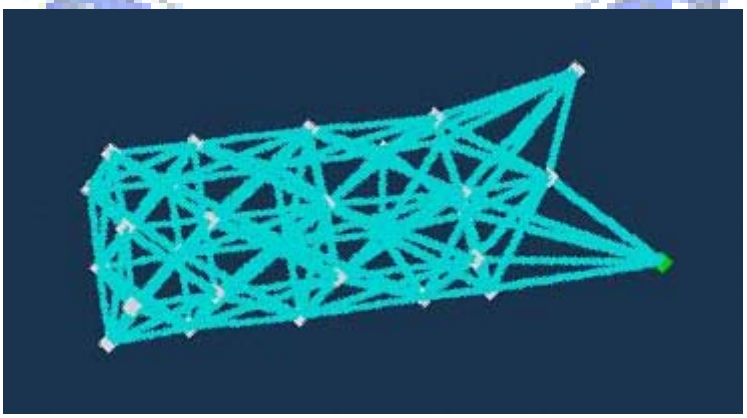
5.2 實驗結果與討論

A. 第一種情形

Server 端操作者及 Client 端操作者對虛擬物朝同樣方向施力，如圖 5-2 所示，從 5-2 圖中可以很明顯看出 Server 端與 Client 端合力搬運時虛擬物的形變，在圖 5-3 中從 X 軸及 Y 軸的走向可以看到 Server 端與 Client 端確實朝相同方向移動，並且從圖 5-4 可以清楚知道 Server 端與 Client 端在移動時所受到力的變化。

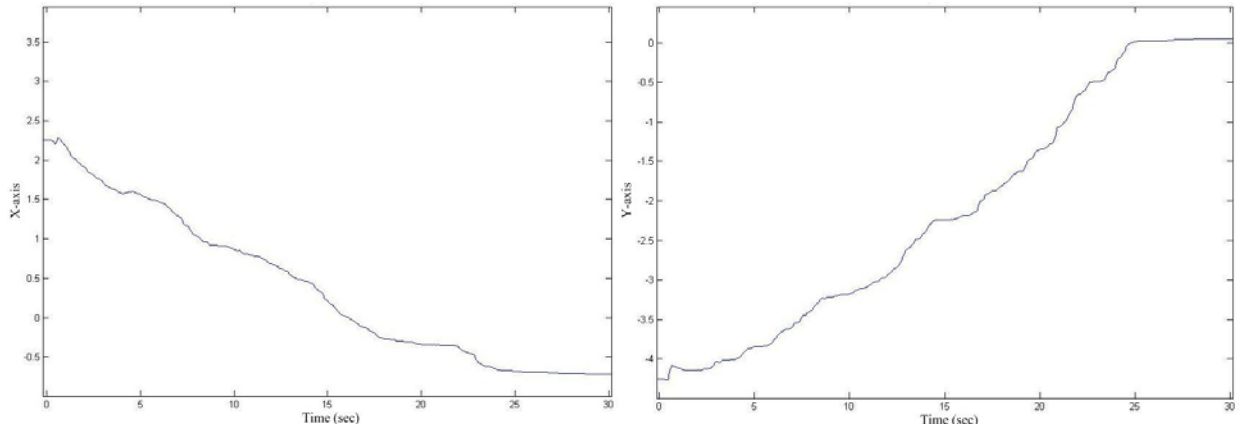


(a)Server 端與 Client 端未對物體施力前

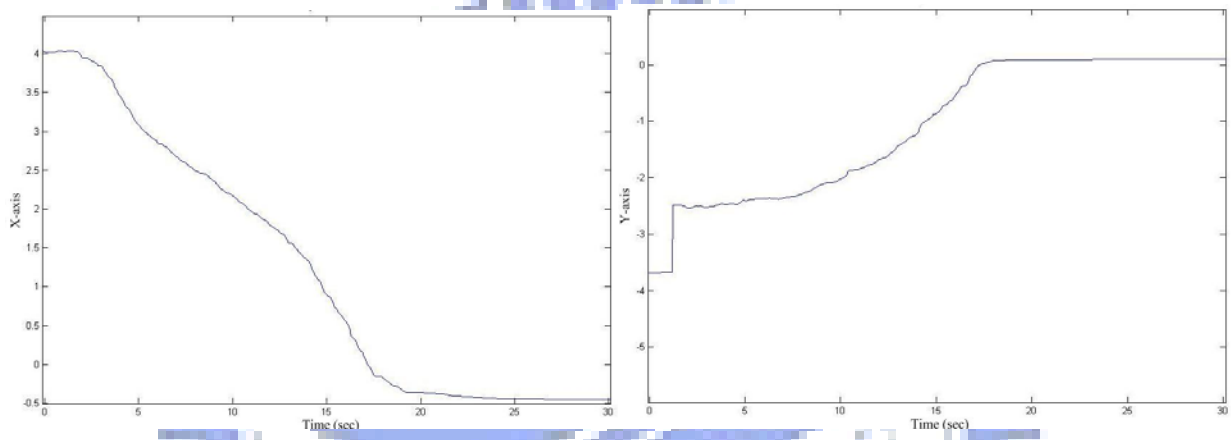


(b)Server 端與 Client 端對物體施力

圖 5-2 Server 端操作者及 Client 端操作者對虛擬物朝同樣方向施力：(a)Server 端與 Client 端未對物體施力前 和(b)Server 端與 Client 端對物體施力

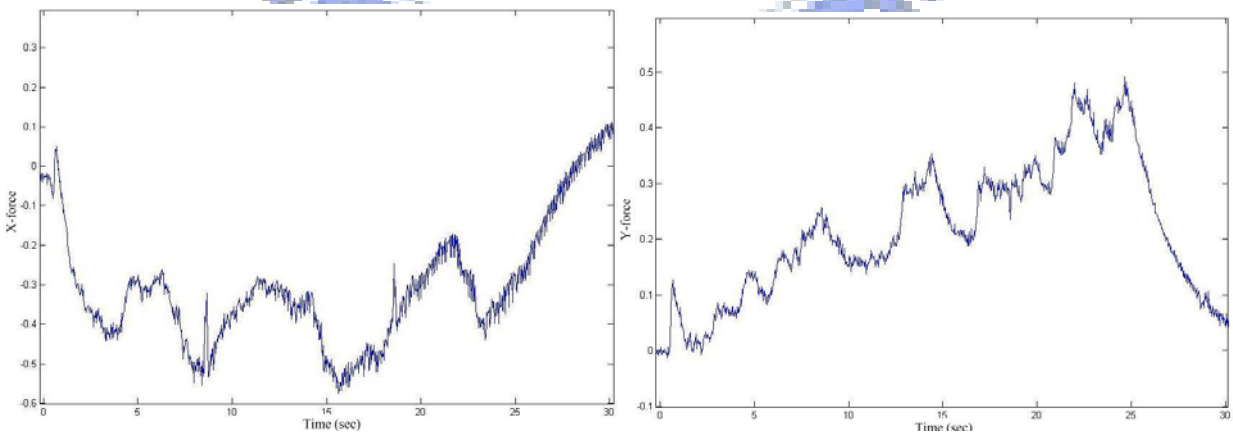


(a)Server 端 X 軸位置變化　　(b)Server 端 Y 軸位置變化



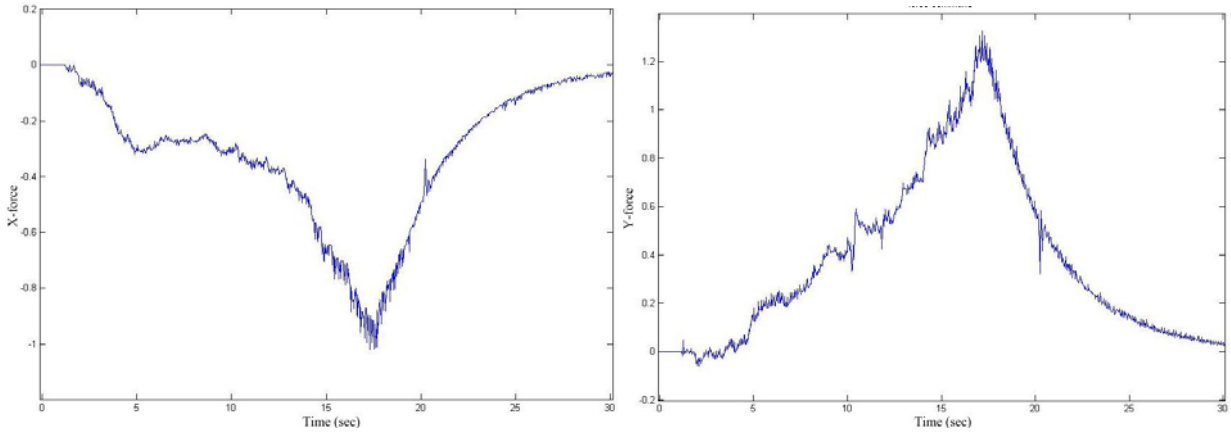
(c)Client 端 X 軸位置變化　　(d)Client 端 Y 軸位置變化

圖 5-3 Server 端及 Client 端位置變化：(a)Server 端 X 軸位置變化，(b)Server 端 Y 軸位置變化，(c)Client 端 X 軸位置變化 和(d)Client 端 Y 軸位置變化



(a)Server 端 X 軸力變化　　(b)Server 端 Y 軸力變化

圖 5-4 Server 端及 Client 端力變化：(a)Server 端 X 軸力變化，(b)Server 端 Y 軸力變化，(c)Client 端 X 軸力變化 和(d)Client 端 Y 軸力變化(cont.)

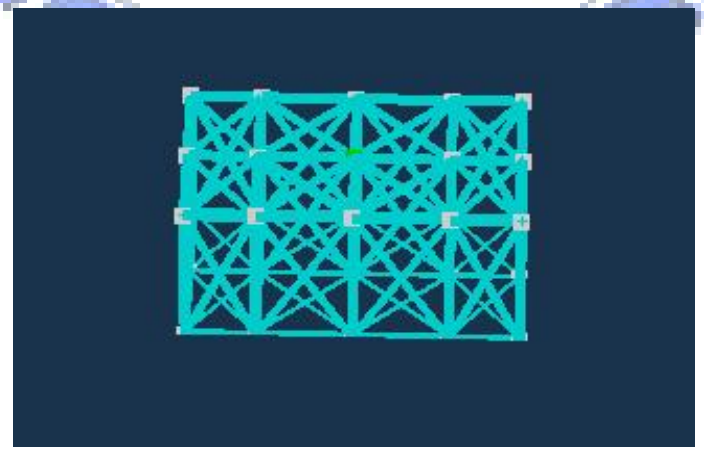


(c)Client 端 X 軸力變化　　(d)Client 端 Y 軸力變化

圖 5-4 Server 端及 Client 端力變化：(a)Server 端 X 軸力變化，(b)Server 端 Y 軸力變化，(c)Client 端 X 軸力變化 和(d)Client 端 Y 軸力變化

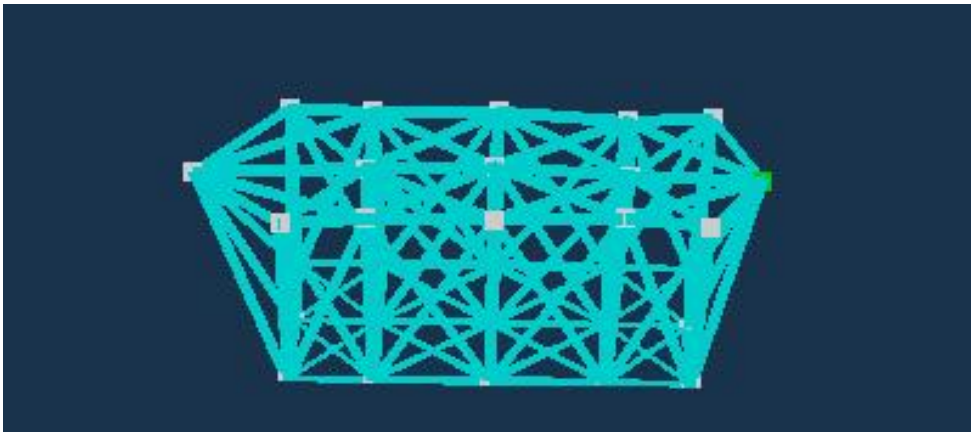
B. 第二種情形

Server 端操作者及 Client 端操作者對虛擬物朝相反方向施力，如圖 5-5 所示，從圖中可以很明顯看出 Server 端與 Client 端力對抗時虛擬物的形變，在圖 5-6 則可以看到 Server 端與 Client 端的位置移動並且能看出有拉扯的動作，Server 端與 Client 端拉扯時互相感受到力的大小如圖 5-7 所示，從圖 5-7，中明顯看到彼此的力對抗。



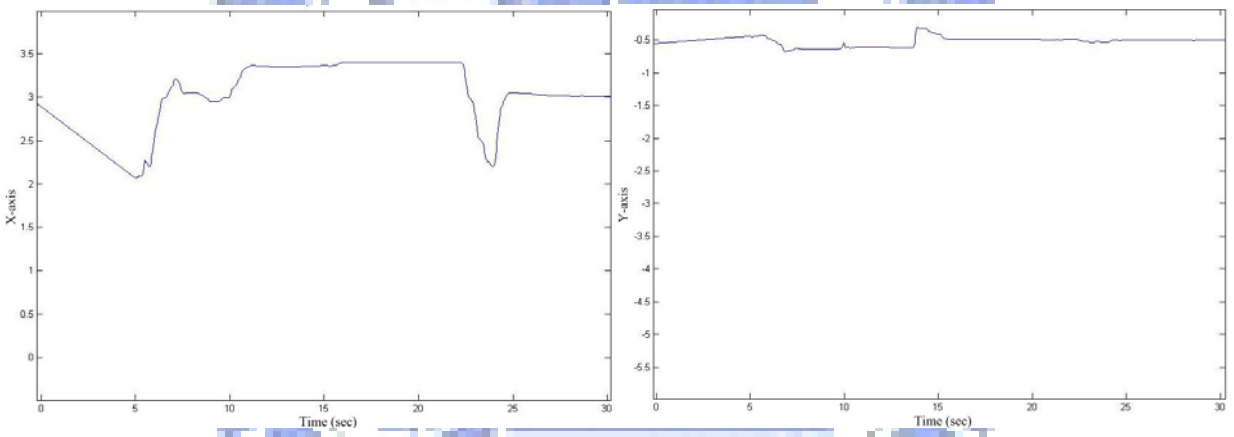
(a)Server 端與 Client 端未對物體施力前

圖 5-5 Server 端操作者及 Client 端操作者對虛擬物朝相反方向施力：(a) Server 端與 Client 端未對物體施力前 和(b) Server 端與 Client 端對物體施力(cont.)

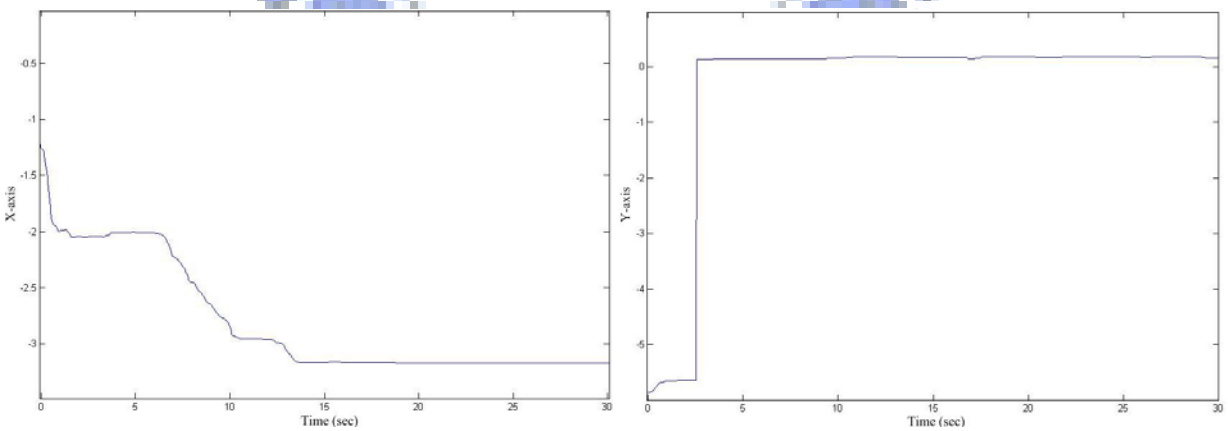


(b)Server 端與 Client 端對物體施力

圖 5-5 Server 端操作者及 Client 端操作者對虛擬物朝相反方向施力：(a)Server 端與 Client 端未對物體施力前 和(b)Server 端與 Client 端對物體施力

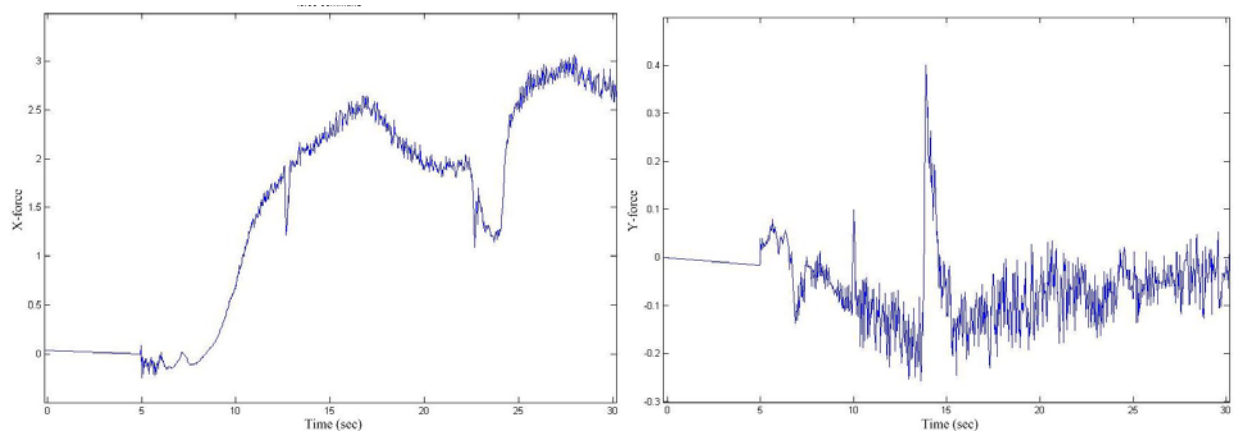


(a)Server 端 X 軸位置變化　　(b)Server 端 Y 軸位置變化

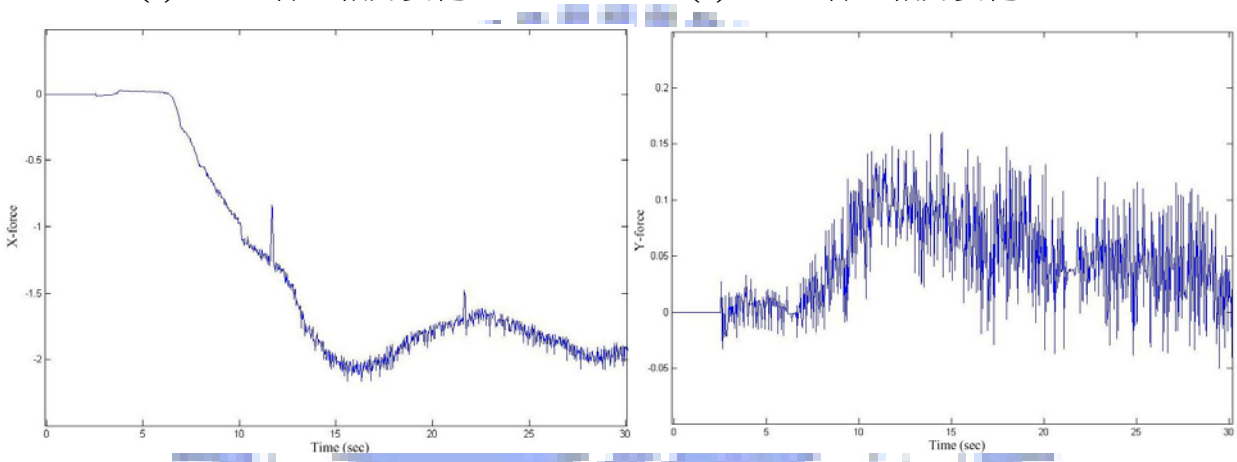


(c)Client 端 X 軸位置變化　　(d)Client 端 Y 軸位置變化

圖 5-6 Server 端及 Client 端位置變化：(a)Server 端 X 軸位置變化 ，(b)Server 端 Y 軸位置變化，(c)Client 端 X 軸位置變化 和(d)Client 端 Y 軸位置變化



(a)Server 端 X 軸力變化　　(b)Server 端 Y 軸力變化



(c)Client 端 X 軸力變化　　(d)Client 端 Y 軸力變化

圖 5-7 Server 端及 Client 端力變化：(a)Server 端 X 軸力變化，(b)Server 端 Y 軸力變化，(c)Client 端 X 軸力變化 和(d)Client 端 Y 軸力變化

C. 第三種情形

Server 端操作者與 Client 端操作者任意一端固定不動，另一端隨意拉扯在這我們讓 Server 端操作者拉住一角不動 Client 端操作者隨意拉扯，如圖 5-8 所示，我們將上方固定然後由 Client 拉下角並任意移動，在圖中可以很明顯看出 Client 端隨意拉扯時虛擬物的形變。在圖 5-9 很清楚看到 Server 端位置變化很小 Client 端位置變動很大，Server 端位置變化是因爲 Client 端拉扯的關係，Server 端與 Client 端之間的力的變化如圖 5-10 所示，Server 端確實依照 Client 端位置改變而感受到力變化。

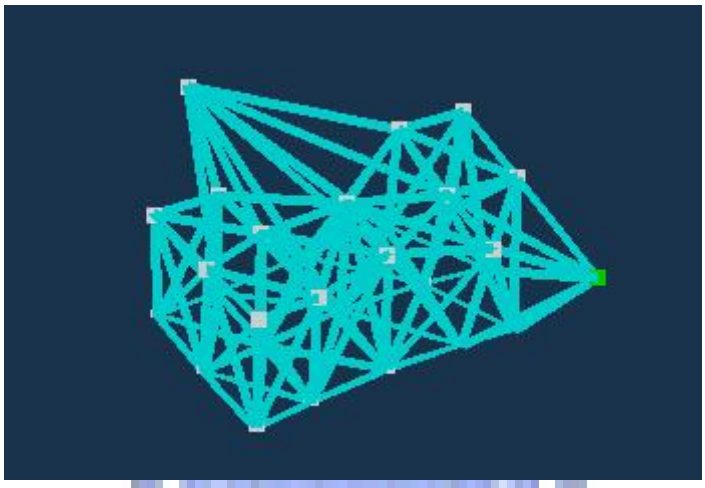
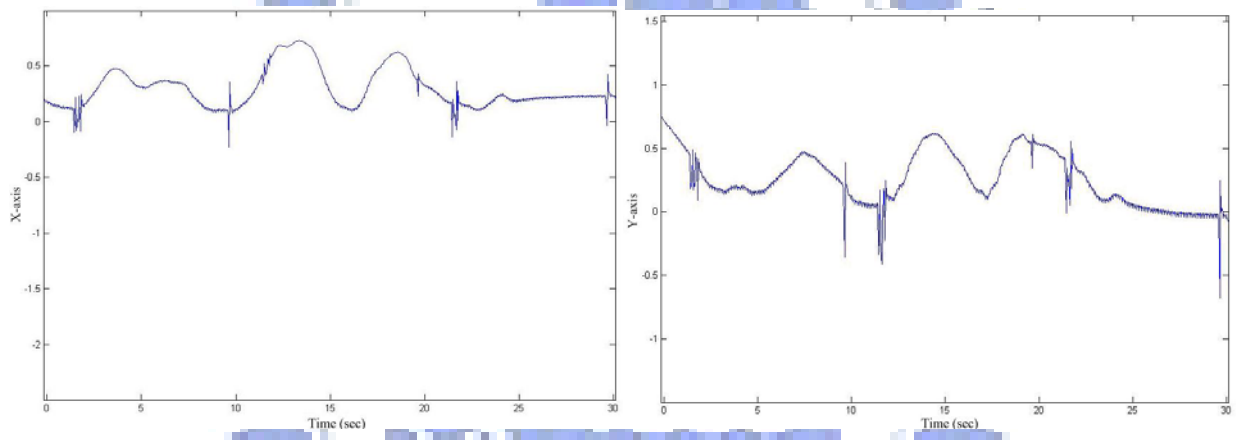
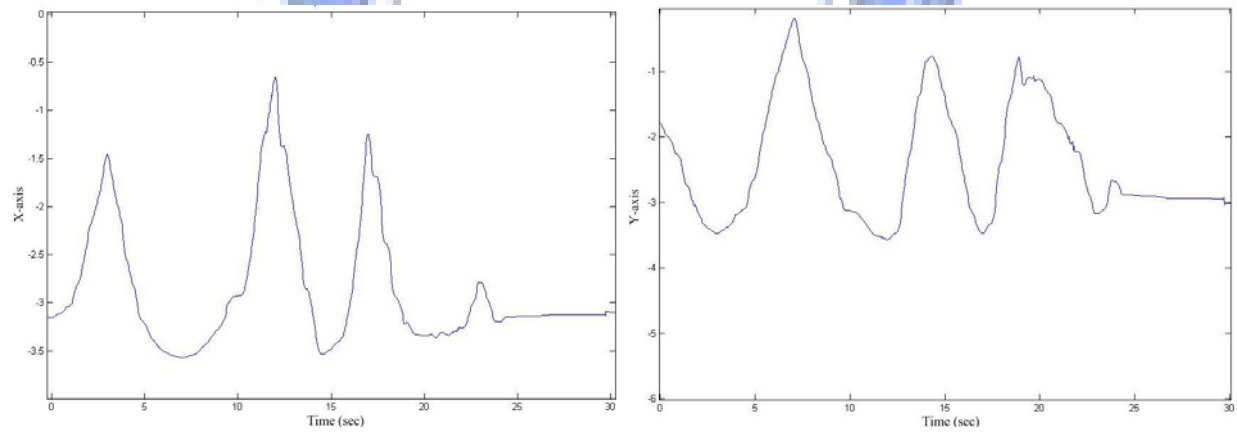


圖 5-8 Server 端與 Client 端對物體施力

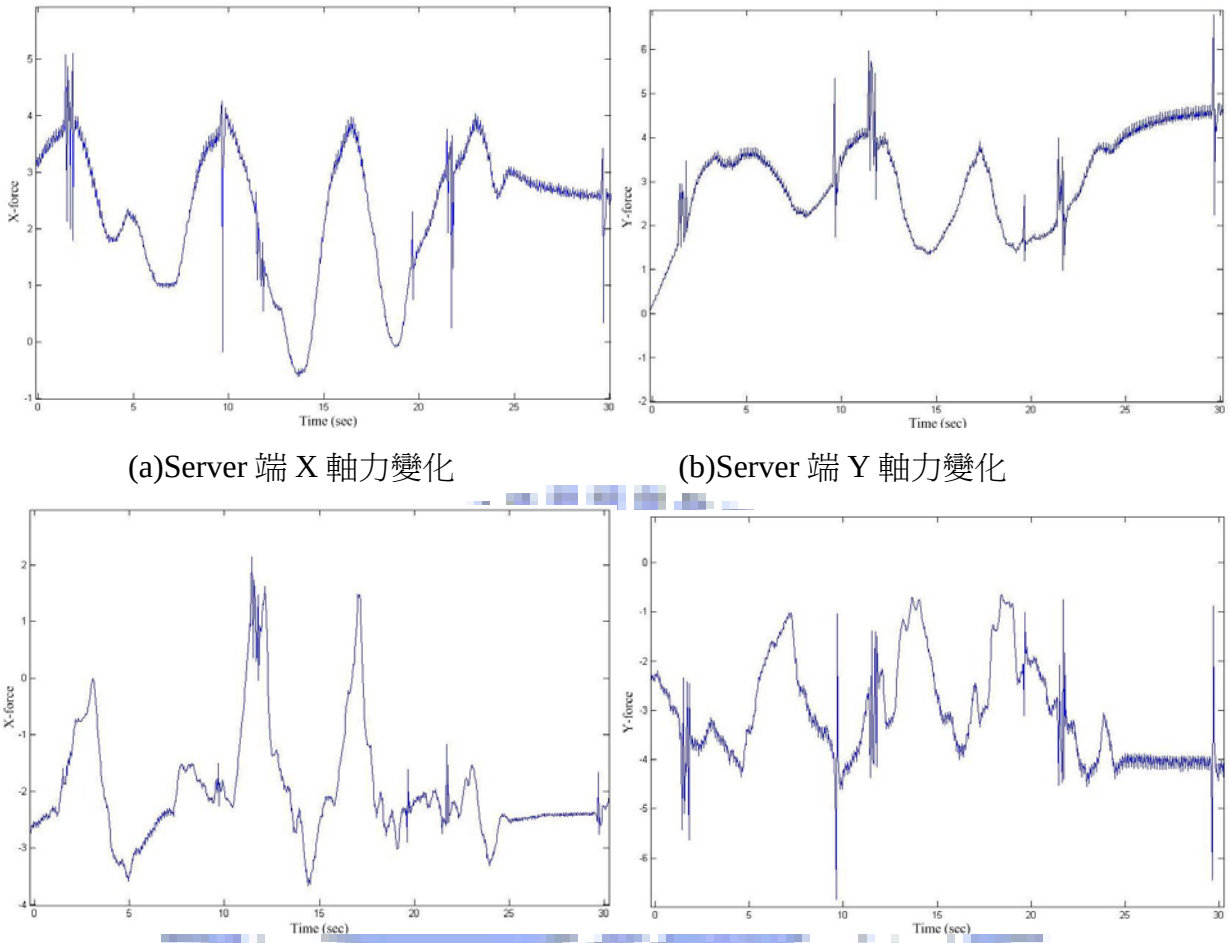


(a)Server 端 X 軸位置變化　　(b)Server 端 Y 軸位置變化



(c)Client 端 X 軸位置變化　　(d)Client 端 Y 軸位置變化

圖 5-9 Server 端及 Client 端位置變化：(a)Server 端 X 軸位置變化，(b)Server 端 Y 軸位置變化，(c)Client 端 X 軸位置變化 和(d)Client 端 Y 軸位置變化



(a)Server 端 X 軸力變化　　(b)Server 端 Y 軸力變化

(c)Client 端 X 軸力變化　　(d)Client 端 Y 軸力變化

圖 5-10 Server 端及 Client 端力變化：(a)Server 端 X 軸力變化，(b)Server 端 Y 軸力變化，(c)Client 端 X 軸力變化 和(d)Client 端 Y 軸力變化

以上三種型態的合力搬運實驗，我們都是在實驗室中完成，從所得到的結果不論是力的反應或虛擬物的 3D 形變呈現都符合當初的預期，但是因爲實驗室的網路是高速網路，所以接下來我們要討論在家用網路環境下對我們系統有何影響。

D. 遠距實驗

首先對於網路延遲的狀況，我們測試家用網路和實驗室中的網路延遲現象是

否差異很大，進行的測試環境同樣是在非強制性即時作業系統，前面所建構的實驗即是在此情形下進行，同樣採用 Microsoft 的 Windows XP。方法爲利用一台電腦作爲 Client，不斷地對另一台電腦 Server 發送資料，而 Server 端接收到資料後即發送訊息給 Client，如此可以得到資訊來回所花費的時間，連續進行 40000 筆測試結果，在實驗室的環境中進行的測試結果如圖 5-11 所示，可知操作過程中使用者可以感受到的延遲大約會在 1 ms 左右，而因爲大部分的延遲還是在低於 2ms 的狀態下，可知雖有少許時刻會有較大的延遲 (高達 8 ms)，但並不影響視覺的更新頻率，而力覺的更新並沒有透過網路傳輸所以更不會造成影響，因爲我們將力的計算放在各自的主機上，若發生網路延遲的事件，力的反應會維持在前一時刻。

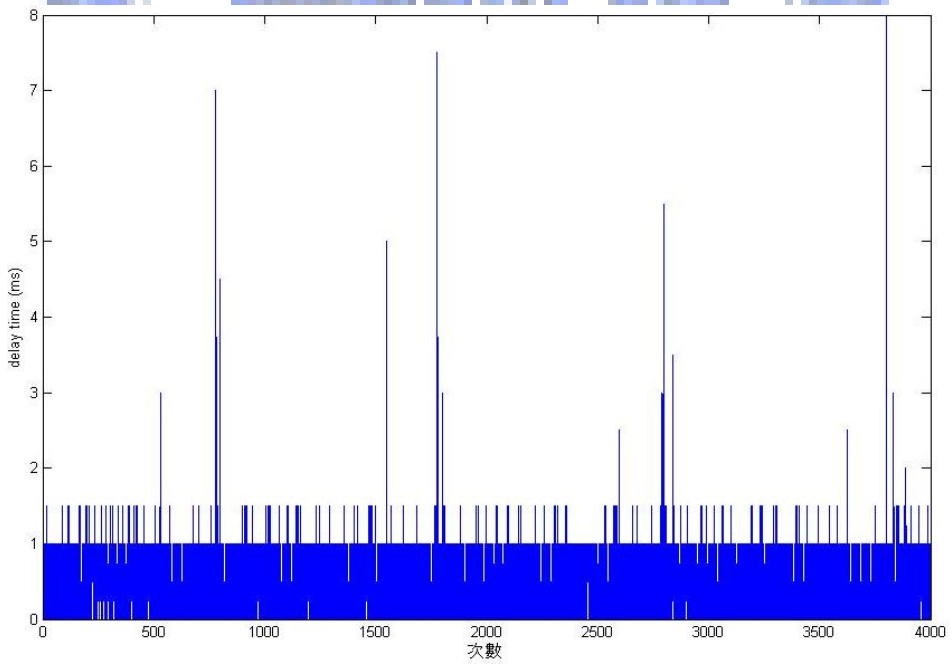


圖 5-11　我們實驗室所測得的網路延遲狀況

但是在一般的家用網路，撥接寬頻 ADSL-2M/256K 測試所得的網路狀況，與實驗室所用的網路傳輸速度差 10 倍以上，如圖 5-12 所示，一般情況下可以感受的網路延遲在 15 ms 上下，換言之，其能達到的力更新頻率應該在 66 Hz 左

右，但這並不包含 Client 執行程式工作的時間，若再加上 Client 執行程式工作的時間，視覺的更新頻率應該會更低。

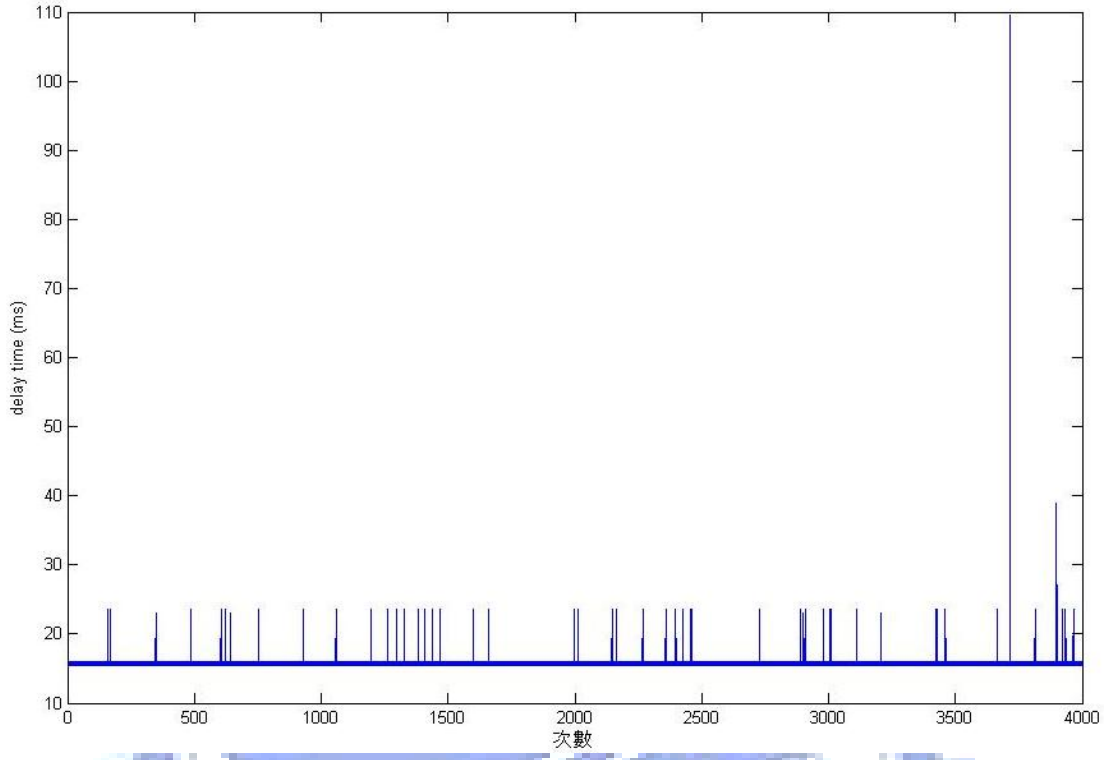
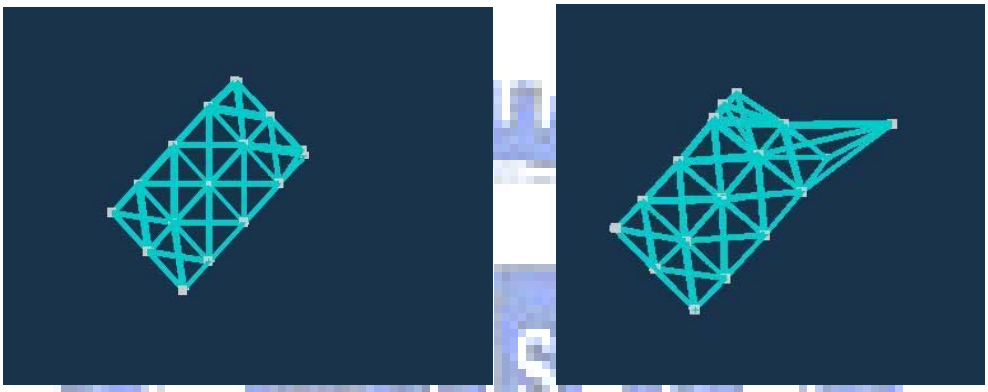


圖 5-12　ADSL-2M/256K 家用網路所測得的網路延遲狀況

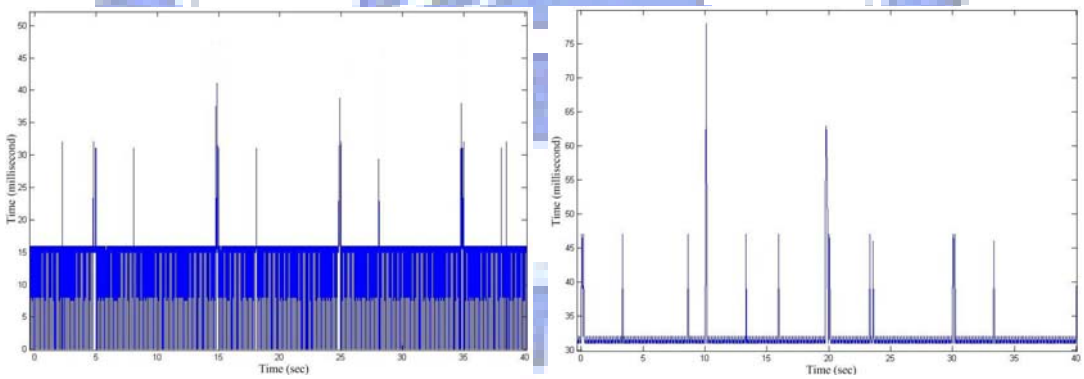
我們以家用網路來重做合力搬運的第一類型實驗，Server 端操作者及 Client 端操作者對虛擬物體朝同樣方向施力，圖 5-13(a)是 Server 端及 Client 端還未移動的狀況，最上方爲 Client 端，最右方爲 Server 端。當 Server 端及 Client 端開始移動虛擬物體時，因爲網路延遲的關係造成 Client 端位置來不及送到 Server 端，使得虛擬物體表現出不正常的反應，如圖 5-13(b)所示，與圖 5-2(b)比較即可知道網路延遲會造成 Server 端與 Client 端彼此的位置資訊無法即時送給對方，進而使得 Server 端與 Client 端畫面更新不能同步，造成明明是做合力搬運的實驗，卻在網路的延遲下變成力對抗。再來爲了確實了解實驗室網路和家用網路在個別的網路延遲下，對本系統的畫面更新有何影響而做以下的量測，圖 5-14 是量測 Client 端經由我們實驗室網路和家用網路每次更新畫面所需的時間，從圖 5-14(a)可看到使用實驗室網路所測得更新一次畫面大約 17ms(58fps)，但偶爾會有要高

達 43ms(23fps) 以目前的實驗來說還在可接受的範圍。但從圖 5-14(b)中看到使用家用網路所測得更新一次畫面大約 32ms(31fps)，而且偶爾會有要高達 78ms(12fps)這就超出了我們所能容許的範圍。所以要在使用網路的狀況下能有順暢即時的畫面，目前還是需要依靠寬頻專線，或是引入其它的預測技術、利用補插間等等的方式來改善。



(a)未施力前　　(b)施力後

圖 5-13 以家用網路重做合力搬運的第一類型實驗：(a)未施力前 和(b)施力後



(a)實驗室網路　　(b)家用網路

圖 5-14 量測 Client 端經由我們實驗室網路和家用網路每次更新畫面所需的時間：(a)實驗室網路 和(b)家用網路

第六章

結論與未來工作

本論文建立了一可攜性的力感系統，讓使用者可以即使身處不同地點，只要帶著力回饋裝置與安裝軟件，就可以在所建置的虛擬環境中進行力資訊的應用。在系統實現方面，我們將合作搬運分爲近端(Slever)與遠端(Client)兩大部份，利用 Impulse Engine 2000 力回饋搖桿與系統相關模組整合成近端操作系統，再利用 Laparoscopic Impulse Engine 力回饋搖桿與系統相關模組整合成遠端操作系統，並且搭配虛擬實境發展了更爲真實與互動性更高的遠端操作環境。而力感部份，因爲沒有在搖桿上裝上力感測器僅是依靠力感模型來實現，所以出力部份可能會有所誤差但也足夠用來感測力的變化，若要得到精確的力值，還是應該在搖桿上裝上力感測器。在程式方面我們使用 Visual C++ (VC++)，一方面是因爲 VC++有執行快速的優點外，另外只要將原始碼經過轉譯，就算不同作業系統也可以執行，如此更達到了力可攜的目的。

6.1 未來工作

我們在最後的實驗中雖然只以雙人的合力搬運來驗證我們的可攜性的力感系統，但並不代表這個系統只能作雙人的合力搬運，例如，只要稍微修改網路的通訊協定我們就可達到多人連線的效果，在實驗中我們只使用一個虛擬物體主要

是爲了讓 3D 的碰撞偵測及碰撞反應系統不要過於複雜，當多個物體的碰撞偵測及碰撞反應能完成時，接下來我們就可以做一個多人連線並有眾多物體的模擬環境，或者做一個碰碰車的遊戲。在本研究中的合作搬運遠端操作系統來說，雖然已具備應有的基本功能，但仍有許多需要改進的地方，也是我們未來的工作，如以下所列：

1. 遠端操作系統需要透過網路傳送訊息，必定會有時間延遲的現象，時間延遲可能會造成系統的不穩定性與不同步性，在網路傳輸時需要藉由適當的控制策略來解決時間延遲對系統所造成的影響。

2. 爲使感覺更加真實，視覺上能接近真實世界的物體，勢必要將模型精緻化，因爲隨著模型的精密度與維度提高，勢必會使系統效能降低，故對於 3D 空間的力感模型與視覺呈現要如何進行，可以再作研究。

3. 進行虛擬世界與真實世界的連結，例如將實驗中，虛擬臂的位置連接指定爲真實機械臂的末端位置，並推動真實的物體，然後同樣藉由目前的力資訊架構，感受力回饋資訊，如此要連結外部的力感測裝置，作多維度的方向與力量判別，而在此方面，整合力回饋裝置以外的實體裝置進入系統後，對其參數的調校，以及採用何種智慧控制，要再依當時應用場合而定。

4. 將碰撞偵測及碰撞反應系統完善，使其在多人連線時能有更多的應用，而不必拘限在單一物體的模擬上，進而達到多物體的模擬。

參考文獻

[1] D. C. Ruspini, K. Koralov, and O. Khatib, "Haptic interaction in virtual environments," *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 128 –133, 1997.

[2] D. M. Bourg, *Physics for Game Developers*, Cambridage, O'Reilly, 2002.

[3] D. M. Eberly, *Game Physics, Boston*, Morgan Kaufmann, 2004.

[4] F. Vahora, B. Temkin, T. M. Krummel, and P. J. Golman, "Development of real-time reality haptic application: real-time issues," *12th IEEE Symposium on Computer-Based Medical Systems*, pp. 290 – 295, 1999.

[5] J. G. Webster and D. G. Hanger, "Telepresence for touch and proprioception in teleoperator system," *IEEE Trans. on System, Man and Cybernetics*, Vol. 18, No. 6, pp. 1020-1023, 1989.

[6] K. B. Shimoga, "A survey of perceptual feedback issue in dexterous telemanipulation: Part I. Finger force feedback," *IEEE Virtual Reality Annual International Symposium*, pp. 263 – 270, 1993.

[7] Kenneth C Finney, *3D game programming all in one*, Boston, Thomson/Course Technology, 2004.

[8] L. D. Joly and C. Andriot, "Imposing motion constraints to a force reflecting telerobot through real-time simulation of a virtual mechanism," *IEEE International Conference on Robotics and Automation*, pp. 357-362, 1995.

[9] L. Dongjun, O. Martinez-Palafox, and M. W. Spong, "Bilateral Teleoperation of Multiple Cooperative Robots over Delayed Communication Networks: Theory," *IEEE International Conference on Robotics and Automation*, pp. 360-365, 2005.

[10] M. A. Srinivasan and C. Basdogan, "Haptics in virtual environments: taxonomy, research status, and challenges", *Computers and Graphics*, Vol. 21, No. 4, pp. 393-404, 1997.

[11] M. Ouh-young, J.-R. Wu, W.-N. Tsai, T.-J. Yang, and C.-H. Huang, "A force feedback joystick and its use in PC video games," *IEEE International Conference on Consumer Electronics*, pp. 326-327, 1995.

[12] M. Ouh-young, W. N. Tsai, M. C. Tsai, J.R. Wu, C. H. Huang, and T. J. Yang, "A low-cost force feedback joystick and its use in PC video games," *IEEE Transactions on Consumer Electronics*, Vol 41, No. 3, pp. 787 -794, 1995.

[13] M. S. Yoh, “The Reality of Virtual Reality,” *Seventh International Conference on Virtual Systems and Multimedia,* pp. 666-674, 2001.

[14] N. Hogan, “Controlling impedance at the man/machine interface,” *IEEE International Conference on Robotics and Automation*, Vol.3, pp. 1626-1631, 1989.

[15] N. Hogan, “Stable Execution of Contact Tasks Using Impedance Control”, *IEEE International Conference on Robotics and Automation*, pp. 1047-1054, 1987.

[16] O. M. Al-Jarrah and Y. F. Zheng, “Arm-manipulator coordination for load sharing using variable compliance control,” *IEEE International Conference on Robotics and Automation*, pp. 895-900, 1997.

[17] R. C. Goertz and R. Thompson, “Electronically controlled manipulator”, *Nucleonics*, pp. 46-47, Nov. 1954.

[18] R. S. Wright and Jr. M. Sweet, *OpenGL SuperBible*, Indianapolis, Waite Group Press 2000.

[19] S. G. Hong, J. J. Lee, and S. Kim, “Generating artificial force for feedback control of teleoperated mobile robots,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1721–1726, 1999.

[20] T. L. Brooks, “Telerobotics response requirements”, *IEEE International Conference on Systems, Man and Cybernetics*, pp. 113 – 120, 1990.

[21] T. Tsumugiwa, R. Yokogawa, and K. Hara, “Variable impedance control with virtual stiffness for human-robot cooperative peg-in-hole task,” *IEEE International Conference on Intelligent Robots and Systems*, pp. 1075-1081, 2002.

[22] *Virtual hands reach across ocean* , http://news.bbc.co.uk/1/hi/technology/2371103.stm