行政院國家科學委員會補助專題研究計畫成果報告

※※※※※※※※※※※※※※※※※※※※※※※※※
※ ※
※ **高解析度虛擬實境系統之多台電腦全然分散式架** ※
※ **構的即時傳輸機制的研究與設計** ※
※ Study and design of a real-time communication ※
※ mechanism for the multicomputer-based high fidelity ※
※ virtual reality system ※
※ ※
※※※※※※※※※※※※※※※※※※※※※※※※※

計畫類別：¦**個別型計畫　　□整合型計畫**

計畫編號：NSC 89-2213-E-032-017ー

執行期間：88 **年** 8 **月** 1 **日至** 89 **年** 7 **月** 31 **日**

計畫主持人：黃俊堯

共同主持人：

本成果報告包括以下應繳交之附件：

□**赴國外出差或研習心得報告一份**

□**赴大陸地區出差或研習心得報告一份**

□**出席國際學術會議心得報告及發表之論文各一份**

□**國際合作研究計畫國外研究報告書一份**

執行單位：淡江大學資訊工程學系

中　華　民　國八十九年　九　月 十二 日

行政院國家科學委員會專題研究計畫成果報告

高解析度虛擬實境系統之多台電腦全然分散式架構的即時傳輸機制的研究與設計

Study and design of a real-time communication mechanism for the multicomputer-based high fidelity virtual reality system

計畫編號：NSC 89-2213-E-032-017
執行期限：88 年 8 月 1 日至 89 年 7 月 31 日

主持人：黃俊堯　淡江大學資訊工程學系
計畫參與人員：蔡明典　淡江大學資訊工程學系

一、中文摘要

本計畫的目的是在研究與設計虛擬實境系統之多台電腦全然分散式(Fully Distributive)**架構的即時傳輸機制。本計畫運用** Candy **所提出來的概念，將模擬器視為由許多不同功能的** physical processes **所組成的，以** pipelining **的方式來達到虛擬平行運算的目標。而此** pipelining **方式的最大變因是** LPs **之間的** synchronization **問題，因此本計畫針對此問題加以探討之後，提出一個全然分散式架構的即時傳輸機制。**

關鍵詞：虛擬實境系統，模擬器，高擬真度模擬，全然分散式系統，同步延遲，等待法則。

Abstract

The goal of this project is to study and design the real-time communication protocol for the multicomputer-based high fidelity virtual reality system. A high fidelity virtual reality system is capable of revealing all of the physics phenomena of the simulated entity and displaying the virtual image at the frame rate between 18 to 30 frames per second. The legacy high fidelity virtual reality system often used multiple multiprocessor computers to construct its computer architecture.

This project focuses on designing transparent communication layer over a fully distributive environment. With this mechanism, researchers can design a high fidelity virtual environment over a fully distributive networked computers.

Keywords: Virtual Reality System, High fidelity, Fully distributive system, Synchronization lag, Waiting rule.

二、計畫緣由與目的

一般高擬真度的虛擬實境系統，例如飛行模擬器或汽車駕駛模擬器，是採用多台的多處理器電腦作為其電腦架構。所謂高擬真度的虛擬實境系統是指該系統能正確的顯示虛擬環境的所有物理現象，同時能達到虛擬實境系統的每秒約 18 **至** 30 **個畫面的顯像速率需求。**[1] **以訓練用飛行模擬器為例，當使用者搖動搖桿時，模擬器電腦系統必須依模擬飛機當時的速度、加速度、高度、風速及地心引力來計算出真實飛機應有的迴轉角度與速度，而這些繁雜計算公式使得原本需要複雜運算的虛擬實境系統執行速度更加緩慢，因此，傳統的高擬真度虛擬實境系統皆採用多台的多處理器電腦作為其電腦架構。美國的福特汽車公司的駕駛模擬器採用一台** Evans & Sutherland **公司的** ESIG-2000 **為該模擬器的顯像電腦，一台** 4 **個處理器的** Apollo **公司的** DN-10000 **電腦來計算所需之汽車動力學，以及一台** Harris **公司的雙處理器** NightHwak Real-Time I/O **電腦來作為資料匯整及交換即為一例。**[2]

然而隨著電腦硬體技術與超大型積體晶片的日益演進，使得個人電腦的售價越來越便宜但運算功能卻越來越強。另一方面，分散式系統是近幾年來學術界一個熱

門的研究題目。所謂的分散式系統是將一個工作分為數個子工作，然後將這些子工作分配給透過網路串聯的數台電腦來執行，以快速地完成該工作。基本上，分散式系統的架構有兩種類型，一種稱為叢集（Cluster）方式，另一種則為全然分散式（Fully Distributive）。而截至目前為止，大部分有關模擬器的分散式電腦系統的研究較偏重在叢集式分散式系統，且大都採用工作站級以上的電腦為工作平台。

本計畫則是以全然分散式電腦系統為目標，研究與探討如何在以網路串聯的一群個人電腦之間建立一個快速傳輸機制，來達到高擬真度虛擬實境系統所要求的每秒 30 個顯像畫面執行效率，並嘗試設計此全然分散式計算環境的雛型試驗系統。此全然分散式系統的研究概念是源自於計畫主持人 85 年國科會計畫的研究成果，以及以美國國防部最近正傾全力發展的高階模擬架構(High Level Architecture)為藍圖。

該年度計畫驗證了在分散式個人電腦環境下建立高擬真度虛擬實境環境的可行性。然而該年度計畫成果引發了一個重要的研究課題，亦即如何在此全然分散式運算環境下確保所建構之虛擬實境系統的執行效率？此執行效率的確保工作包括了每秒顯像速率是固定的，不會隨著執行時間的長短、分散式電腦的數量、及子工作的個數所影響。

為達成本計畫的工作目標，本計畫將對高擬真度虛擬實境系統的特性需加以詳細研究，並設計實驗來研究與驗證分散式運算環境的傳輸機制，同時，本計畫執行過程將收集與研讀美國國防部所制定的高階模擬架構中即時執行介面(Run Time Interface)規格的通訊協定[3]，以提出一個方法在全然分散式計算環境下，建立快速資訊傳輸機制來達到高擬真度虛擬實境系統的需求。

三、結果與討論

本計畫運用 Candy 所提出來的概念[4,5]，將模擬器視為由許多不同功能的 physical processes 所組成的，並且藉由 physical processes 彼此間的互動(interaction)連繫而構成的一個 physical system。利用這個特性，可將由 physical processes 所構成的 physical system 對應到由 logical processes(LPs) 所構成的模擬系統，而且每一個 LP 分別模擬其對應的 physical process，至於 physical processes 之間的互動，則藉由 LPs 之間訊息的傳遞來模擬。透過這樣的方式，可將循序執行的模擬系統分散到一群電腦上，利用這群電腦的分工合作使整個模擬器電腦系統能達到每秒 20~30 個畫面的處理能力。

由於 LPs 之間的互動是藉由它們彼此間 message 的傳遞來達成，對任何 message 而言，一定有所謂的訊息的發送端及訊息的接收端。因此發送端在傳送之前一定要知道接收端在何處，才能確保把資料正確地傳送到達對方。若是要讓每一個 LP 自己去處理這些瑣碎的問題，則程式會顯得相當沒有彈性。以分散式系統(distributed system)的觀點來看，它的主要精神是要對使用者隱藏系統的分散性，讓使用者產生一個假象，以為整個電腦網路只是一個虛擬的單一處理機(a virtual uni-processor)。換句話說，分散式系統的一切資源(resource)對使用者來說都是透明化的(transparent)，使用者本身並不需要去分辨哪些資源是屬於遠端機器所有，哪些資源是本身就有的，就可以將它們視為本身的資源來使用。本計畫的分散式運算系統的構想即來自於分散式作業系統的精神--提供一個在分散式環境下發展模擬器電腦系統之基本執行架構，使得建構在模擬器電腦系統之基本執行架構之上的 LPs(應用程式)，可以透過此基本執行架構所提供的溝通介面，在不需要要知道對方之存在(實際上對方要存在)的情況下，即可以和其他 LPs 溝通。而且只要汰換部份不同功能的 LPs，就可構成不同性質的模擬器電腦系統以提升模擬器電腦系統開發的速度，節省模擬器電腦系統開發的時間與成本[6]。

在分散式的架構中，系統的正常運作是依賴 LP 之間的互動來達成，唯有 LP 之間彼此協調才能保證系統的正常運作。而

本計畫所提之系統是以 pipelining[7]的方式來運作，因此，以下將就 pipeline 架構中的 synchronization lag[8]來探討 LPs 之間的 synchronization 問題。所謂的 synchronization lag 定義如下：

Synchronization lag ：在 LP_i 傳 message 給 LP_{i+1}(指下一個 LP)時, message 由 LP_i 送出後，直到被 LP_{i+1} 處理的這一段時間。

Synchronization lag 在分散式的架構下有著下列特性：若減少系統內的 synchronization lag，會使整個系統的 throughput 提高；但相反的，若增加系統的 throughput，並不意味著已經減少 synchronization lag。

若以 synchronization 的觀點來看，在分散式架構下 LPs 之間的 communication delay 包含於 synchronization lag 中，並以 D 來表示 communication delay 的時間(假設每一次的 communication delay 的時間都一樣)。考慮在第二個畫面(如圖 1 所示)的 LP_1 執行完畢，LP_1 所輸出的結果必須等到處理第一個畫面的 LP_2 執行完畢之後，才會被處理第二個畫面的 LP_2 所接受，於是 synchronization lag 的長度為 $(C_2 - C_1)+ D$，同理，在第三個畫面(如圖 2 所示)的時候 LP_1 與 LP_2 的 synchronization lag 為 $2*(C_2 - C_1)+ D$，因此可歸納出在第 j 個畫面的時候 LP_i 與 LP_{i+1} 之間的 synchronization lag 為 $(j-1)*(C_{i+1} - C_i)+ D$。

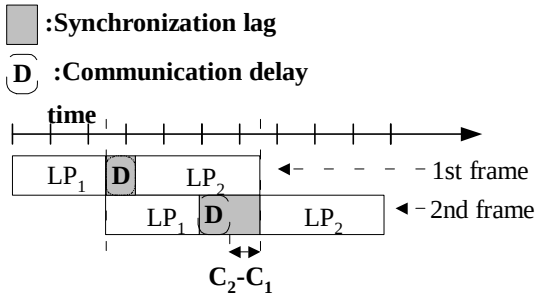


圖 1. 第二個畫面時 LP_1 和 LP_2 之間 synchronization lag 的時間長度

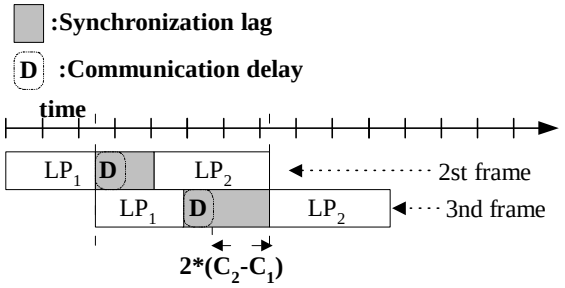


圖 2. 第三個畫面時 LP_1 和 LP_2 之間 synchronization lag 的時間長度

雖然 LPs 之合併似乎可以解決 synchronization lag 所造成的資源浪費，但是並不是任意組合的 LPs 都可以合併，所以 synchronization lag 仍然未得到妥善的解決，最好的解決辦法是在分析系統準備切割 LP 之際，讓每一個 LP 的執行時間都一樣，即為最佳的解決方案，但是要滿足讓最多 LPs 可以同時執行又不會互相干擾，而且還要維持每個 LP 的執行時間都一樣，對模擬器的設計者來說是一件不容易的事。為了解決這一個問題，本計畫提出解決 synchronization lag 之方法為：在 LP_i 執行完畢欲將結果傳給 LP_{i+1} 時，必須遵循 wait rule。而 wait rule[4,5]的定義如下：

Wait rule：當 LP_i 輸出的計算結果傳給 LP_{i+1} 後，必須等到 LP_{i+1} 接收到且準備處理時，LP_i 才能繼續執行下一個畫面的運算。

以即時性系統的時間規範 (time constraint)之觀點來看，考慮遵循 wait rule 的系統與不遵循 wait rule 的系統對特定的 task 之抵達時間、運算時間和規定完成時間，經過比較發現該 task 的運算時間和規定完成時間在這兩種系統下都沒有改變，只有 task 的抵達時間在不遵循 wait rule 的系統裡時間提早了，相對的使得系統必須提早為其安排執行的時間，在多個 tasks 合併於同一台機器上執行的情況之下，會影響到原先已經排定好的 tasks 之執行時間被迫往後順延。

一個虛擬實境系統經過 Timed Petri Nets 的分析(analysis)和模型化(model)後，

可分割成許多 logic processes(LPs)，且這些 LPs 之間會存在著溝通(communication)的問題。若讓每一個 LP 自己去處理溝通的問題，則對程式設計者來說是個沉重的負擔，而且程式會顯得很沒彈性。所以本計畫設計了一個溝通的基本架構(baseline infrastructure)來處理 LPs 在分散式環境中的溝通問題，並對 LPs 隱藏系統的分散特性，使模擬器電腦系統可以很容易地在分散式環境下運作，並得到加速運算的效果。本計畫所設計之基本架構依其功能可分成三層(如圖 3 所示)：

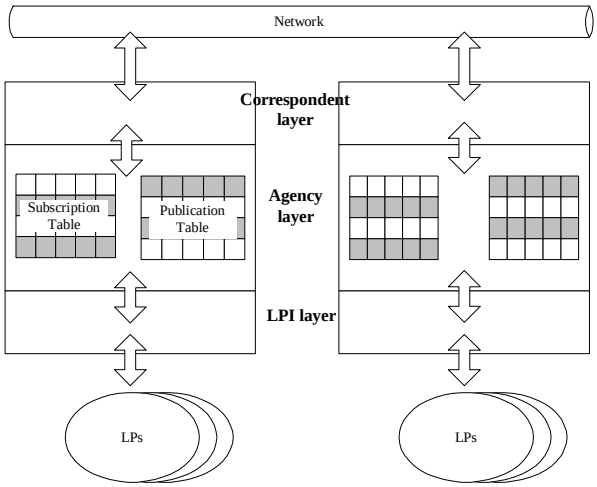


圖 3. 分散式虛擬實境系統基礎架構圖

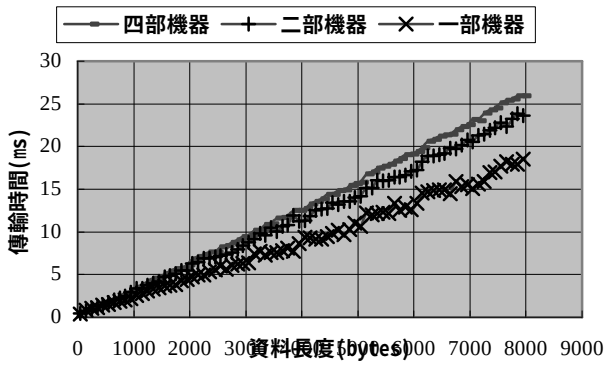
此三層式架構的功能為

- LP interface layer (LPI layer)：其主要的工作為提供 interface routines 給 LP 使用，並負責與 Agency 層溝通。
- Agency layer：其主要的工作為保存 LPs 的資訊，依照 LP 是 publisher(對 LPI Layer 而言，凡是呼叫 publish()者皆是)或是 subscriber(對 LPI Layer 而言，凡是呼叫 subscribe()者皆是)角色分別儲存於 Publication Table 或是 Subscription Table，然後 Agency Layer 會根據這兩個表格裡的資訊透過 Correspondent Layer 所提供的 routine 來與其他電腦端的 Agency layer 作〝交談〞，以解決 Publication Table 與 Subscription Table 之間的配對問題，建立它們的對應關係。等到 LPs 之間的連線關係建立後，Agency Layer 就可以處理 LPs 之間資料的交換。
- Correspondent layer：負責此分散式架構各電腦之間的實際網路溝通工作。

四、計畫成果自評

在本計畫所提出之架構中，影響這 pipeline 架構的執行週期之因素包括 task 本身的運算時間和 tasks 之間的通訊延遲。模擬器電腦系統所關心的問題是：已知 task 本身的運算時間和 tasks 之間所要傳遞之資料長度，則可以換算出傳送資料所花的時間，進而求出 tasks 中最大的執行時間(包含 task 本身的運算時間和網路傳輸的時間)以預測系統將來執行的 performance。

本計畫之網路架構架設在 Ethernet 網路上，經由在 10Mb 的網路環境下實驗測試的結果如下



由上述實驗得知若以每次傳輸資料封包大小為 250 Bytes 來看，傳輸時間大概要花 1~2 milliseconds，而此 250 Bytes 為計畫主持人同年度另一個計畫所測得之實際所需的資料封包大小。該計畫為設計一套六軸動感平台之模擬系統，而本計畫成果曾套用在該計畫上獲得每秒約 22 個畫面的效率，進而證明計畫理論的正確性與實用性。

五、參考文獻

[1] G. Burdea and P. Coiffet., “Virtual Reality Technology”, John Wiley & Sons, Inc., 1993.

[2] J. A. Greenberg and T. J. Park, "Driving Simulation at Ford", Automotive Engineering, pp.37-40, September 1994.

[3] ”High Level Architecture Interface Specification version 1.3”, 1998. Available at http://hla.dmso.mil/hla.

[4] K. Chandy and J. Misra, “Distributed Simulation: A Case Study in Design and Verification of Distributed Programs”, IEEE Transactions on

Software. Engineering , Vol. SE-5, No. 5, 1979

[5] K. Chandy and J. Misra, “Asynchronous Distributed Simulation via a Sequence of Parallel Computations”, CACM, Vol. 24, No. 11, 1981.

[6] D. E. Guckenberger, “Riding the Third Ware with DIS & HLA”, MS&T, May 1996, pp.28-36.

[7] V. Allan, R. Jones, R. Lee and S. Allan, “Software Pipelining”, ACM Computing Surveys, Vol. 27, No. 3, 1995, pp. 367-432.

[8] M. Wloka, ”Lag in Multiprocessor Virtual Reality” Presence, Vol. 4, No. 1, 1995, pp. 50-63.