摘要

壹、前言

一、研究動機

在高一時，我完成了一個名為「數位鏡面」的專案，靈感源自丹尼爾・羅森（2017）在桃園機場捷運的一系列藝術品。這些作品利用鏡頭捕捉現實畫面，經過電腦計算後，以獨特的方式呈現影像。其中，有一幅作品以線條形式重構畫面，宛如一面極具創意的鏡子（如圖 1），而我的專案正是以此為藍本進行模仿與實現。

圖 1: 丹尼爾・羅森的數位鏡面

在實作過程中，我發現許多參數會影響數位鏡面的效果，例如線條的長度與寬度等表層設定，或是每次刷新時繪製的線條數量等底層設定。此外，環境因素的變化也會影響數位鏡面的呈現效果。這些觀察激發了我的好奇心，使我想深入研究各種參數對數位鏡面效果的影響，進一步探索它背後的運作機制。

二、研究目的

（一）了解不同參數設定對於數位鏡面的影響

（二）了解不同影像環境對於數位鏡面的影響

（三）探討不同環境與需求下數位鏡面的最優設定

三、文獻探討

（一）數位鏡面

數位鏡面最初是丹尼爾·羅森（2017）在機場捷運展示的一系列藝術作品。這些作品透過鏡頭捕捉現實世界的影像，並經由電腦計算處理後，以特殊的方式呈現在螢幕上。影像如圖 2所示，經過數位化處理後，現實畫面顯得模糊且朦朧，營造出獨特的視覺效果。該系列中的每件作品皆展現了不同的風格與表現形式。本次實驗將聚焦於其中一種以類似線條形式呈現的數位鏡面作品，進行深入探討。

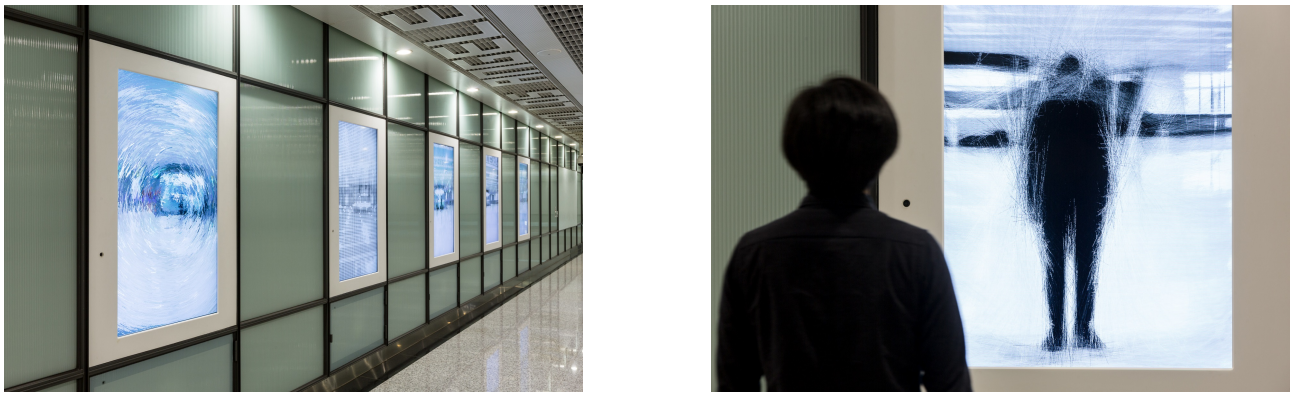

圖 2: 丹尼爾・羅森的數位鏡面

本次研究使用的是 happycorn 在 GitHub 上提供的數位鏡面模仿作品（2024）。程式中展示了以線條為主要呈現方式的數 33 位鏡面，其每一幀的繪製過程包括：讀取影像、繪製多條線條，以及刷新畫面。繪製線條的過程則可細分為以下步驟：模擬一條虛擬線、計算該線條路徑上所有像素點的平均值、以及根據計算結果繪製線條。更詳細的流程如圖 3所示。

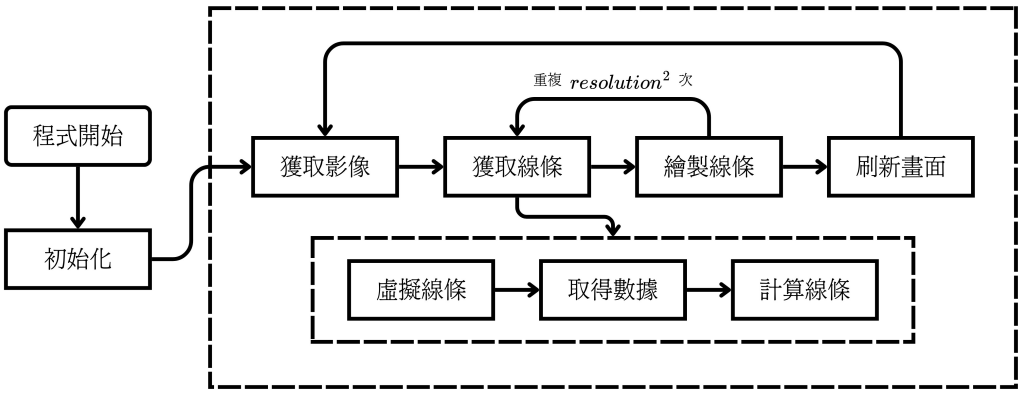

圖 3: 數位鏡面流程圖

在整面數位鏡面中，可調整的參數有三項：線條寬度（width）、線條長度（length）與解析度（resolution）。線條寬度與長度分別表示數位鏡面在繪製時的線條寬度與長度；而解析度則是與繪製過程中的區域劃分相關。為了同時達成「繪製多條線」與「分散線條位置」的效果，作者透過 for 迴圈將整個鏡面切分為多個小方格，並在每個方格內繪製一條線，解析度即代表這些方格的寬度。

（二）時間複雜度

為了比較程式的計算速度，本研究引用了《Introduction to Algorithms》（Cormen et al., 2022）中關於執行時間（running time）的計算觀念。該觀念首先假設，在程式中第 i 步的執行時間為常數 c_i。接著，針對輸入大小 n，寫出相應的操作步數。例如，若要透過 for 迴圈從長度為 n 的陣列中找出最大值，則需要執行 $n \cdot c_i$ 次操作。

然而，直接這樣表示可能顯得過於繁瑣，因此可以進一步簡化為「增長的量級」。具體而言，只需保留最高次項，並忽略其係數，因為在輸入規模增長後，係數對總體增長的影響微乎其微。這樣的計算方式被翻譯成中文時，通常被稱為該演算法的「時間複雜度」。

（三）Sum of absolute difference

在本次實驗中，我需要計算圖片的相似度，為此將採用《Intelligent Image and Video Compression》（Bull & Zhang, 2021）中提出的 Sum of Absolute Difference（SAD）演算法。該演算法針對兩張圖片 s_1 和 s_2，計算它們之間的差異。

假設圖片 s_1 和 s_2 的尺寸為寬 X 和高 Y，則 SAD 的計算方式如下：

$$SAD = \sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} |s_1[x, y] - s_2[x, y]|$$

其中，$s_1[x, y]$ 和 $s_2[x, y]$ 分別表示兩張圖片在座標 (x, y) 的像素值。若圖片為彩色，則需引入額外的色彩通道維度，分別計算每個通道的差異，並累加為最終結果。

貳、研究設備及器材

表 1: 研究設備與器材

類別	項目	型號或版本
硬體	電腦乙台	ASUS M3401QC
軟體	Python	3.12.7
	Anaconda	24.11.0
	Jupyter Notebook	7.2.2
	OpenCV	4.10.0
	NumPy	2.1.3
	Matplotlib	3.9.2

參、研究過程與方法

一、研究架構圖

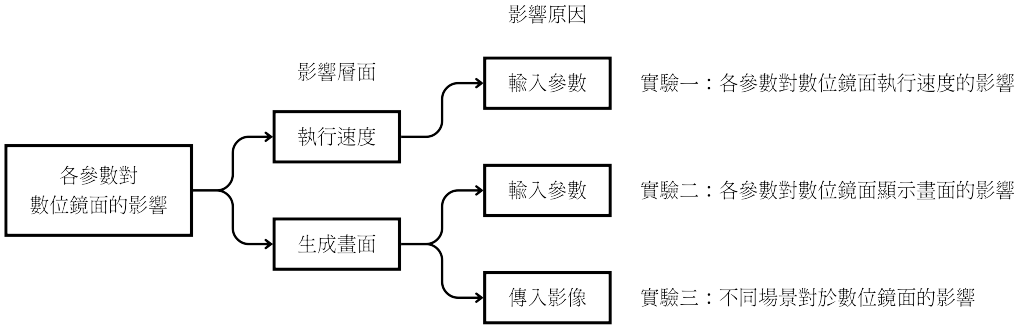

圖 4: 研究架構圖

二、實驗一：各參數對數位鏡面執行速度的影響

為了分析數位鏡面程式的執行速度及其影響因素，本實驗記錄程式的執行時間，並探討可調整參數（線條寬度、線條長度及解析度）對每幀渲染時間與單條線條繪製時間的影響。實驗設計如表 2 所示。

表 2: 執行速度與參數關係的實驗設計

	操縱變因	應變變因	控制變因
1	線條寬度（width）	繪製一條線所需的時間	線條長度、解析度等其他參數
2	線條長度（length）	繪製一條線所需的時間	線條寬度、解析度等其他參數
3	線條寬度（width）	繪製每幀所需的時間	線條長度、解析度等其他參數
4	線條長度（length）	繪製每幀所需的時間	線條寬度、解析度等其他參數
5	解析度（resolution）	繪製每幀所需的時間	線條寬度、線條長度等其他參數

在實驗設計中，每組實驗將重複執行 100 次，並計算所得結果的平均值以作為最終數據。非操縱變因的參數均固定為預設初始值，其中線條寬度（width）設定為 5，線條長度（length）設定為 100，解析度（resolution）設定為 100。

三、實驗二：各參數對數位鏡面顯示畫面的影響

數位鏡面在繪製過程中，當畫面發生變化時，所呈現的內容會逐漸接近現實場景，直至達到一定相似程度後停止變化。這一階段被稱為穩定狀態。本實驗主要觀察兩項指標：一是數位鏡面進入穩定狀態後與現實的差異，二是畫面變動後達到穩定狀態所需的時間。

在實驗中，我將數值分別設定為以下範圍：線條寬度（width）為 1 至 41，間隔 2；線條長度（length）為 1 至 501，間隔 25；解析度（resolution）為 10 至 510，間隔 25。值得說明的是，解析度的起始值設定為 10，這是基於實驗一的結果推算得知，若將解析度設為 1，運算時間將增加至預設值的 10^4 倍，不僅超出實際應用的可能性，也難以有效控制實驗時間。

因此，解析度選擇從 10 開始，該設定的運算時間約為預設值的 10^2 倍，既符合實驗條件要求，也更接近實際應用場景。

為了量化數位鏡面與現實場景之間的相似程度，本研究採用前述文獻探討中提及的 SAD 演算法。為使計算結果更具直觀性，將 SAD 值歸一化為百分比形式，其方法為將 SAD 值除以理論最大值，公式如下。其中，X 和 Y 分別表示圖片的寬度與高度，255 為像素值的最大可能差異。

$$SAD_{\text{normalized}} = \frac{\sum_{x=0}^{X-1} \sum_{y=0}^{Y-1} |s_1[x,y] - s_2[x,y]|}{X \cdot Y \cdot 255}$$

SAD 結果的形態預期與分布如圖 5 所示。為判定穩定狀態，數據被劃分為 20 組，並將出現頻率最高的組別作為穩定狀態的指標。當實驗數據逐漸收斂至該組別，即可判定實驗已達穩定狀態。

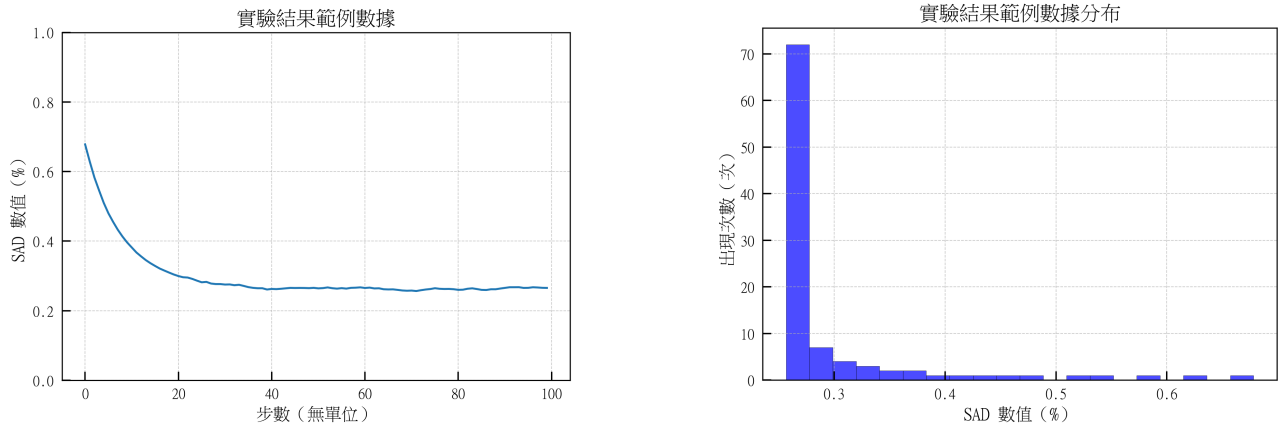

圖 5: 實驗結果範例數據

此外，為了統一不同參數下的測試條件，我選用了三組特定的測試圖片，分別為白藍交錯、紅黑交錯與紅黃交錯的圖案，作為數位鏡面的輸入來源。具體圖片如圖 6 所示。

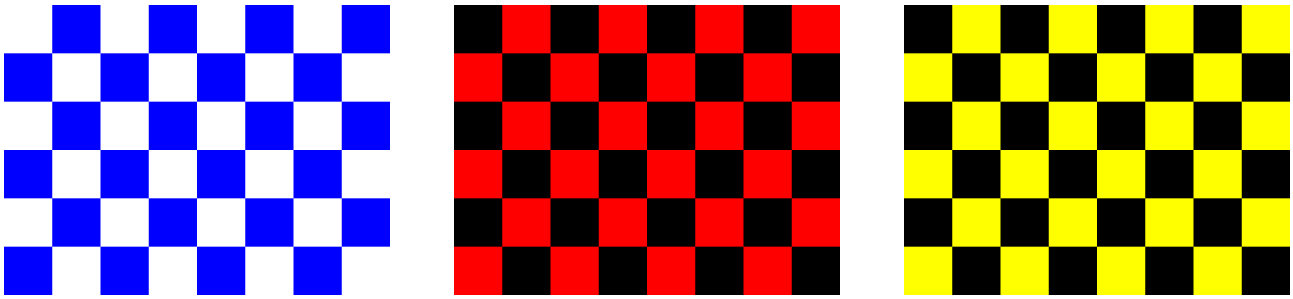

圖 6: 測試照片

四、實驗三：不同場景對於數位鏡面的影響

為探討不同場景對於數位鏡面的影響，我將場景可能不同的點列為：總體場景的複雜程度、總體場景的顏色數量與總體的顏色差異程度。

為了模擬這三種不同的向度，我將製作幾種圖片分別對應他們。總體場景的複雜程度我透過將畫面切分成 12 塊、48 塊、192 塊的方式模擬。總體場景的顏色數量我將其分成 2 種、3 種（紅、綠、藍）與 6 種（紅、綠、藍、黃、橘、紫）顏色。

總體的顏色差異程度因為在顏色多的狀況下難以量化，因此只對 2 種顏色的組別做測試，顏色分別是黑搭配紅、綠、藍、黃、橘、紫與白七色。

實驗設計表如下：

表 3: 不同場景複雜程度與顏色數量影響的實驗設計

	12 塊	48 塊	192 塊
三種	紅、綠、藍	紅、綠、藍	紅、綠、藍
六種	紅、綠、藍、黃、橘、紫	紅、綠、藍、黃、橘、紫	紅、綠、藍、黃、橘、紫

	12 塊	48 塊	192 塊
兩種	紅 綠 藍 黃 橘 紫 白	紅 綠 藍 黃 橘 紫 白	紅 綠 藍 黃 橘 紫 白

肆、研究結果

一、實驗一：各參數對數位鏡面執行速度的影響

二、實驗二：各參數對數位鏡面顯示畫面的影響

（一）穩定狀態與現實的差異

（二）達到穩定狀態所需的時間

三、實驗三：不同場景對於數位鏡面的影響

伍、討論

陸、結論

柒、參考文獻資料