

摘要

本計畫希望能夠在虛擬實境中進行 3D 繪圖，使用者會進入 VR 空間進行繪畫，在多次實測後發現使用者對此空間會有認知上的難度與負擔。經過考慮之後，本計畫決定實作使線帶間的空隙利用程式自動補齊與修正功能，減少使用者遊玩上的困難，提供他們可以揮灑自己創意的平台是本研究的開發目的。

使用的開發程式為 Unity3D，透過結合 SteamVR 來進行介面設置與手把控制。本計畫的遊玩方式相似於小畫家等繪圖軟體，所以在介面設計上也參考小畫家的功能做出了相似功能，並且做出相關的功能使用指南方便使用者上手。

縫合功能是為了方便使用者繪圖的功能，作法相對複雜。基本定義是若兩線帶距離足夠相近，程式會自動將其帶縫合成平面，減少多餘的空隙產生，可以降低繪圖難度。不過這樣的特殊功能會衍生出，因為資料點重複存取及搜索相近資料點的距離，導致程式緩慢甚至卡頓的問題。

為了解決這樣的問題，本計畫決定使用 KdTree 來解決此問題。這個快速搜尋法可以減少為了檢索資料點而執行過多次的情況，透過分類的方式給將需要的資料點放入 KdTree 當中。因為經過多層的分類，檢索會加速許多，同樣的程式也不會多次執行，可以有效的使程式負擔減少。最終成品期望做到使用者可以繪畫出完整的圖畫，並且順利的縫合線帶間的空隙，建構出完整的 3D 物件。

研究初期在做 Unity 專案的時候，經常遇到版本控制的問題，所以本計畫最後利用 GitHub¹解決此問題。GitHub 是 Microsoft 旗下一款專門上傳專案以及專案版本控制的一個軟體。正確的更新專案可以免去很多版本錯誤所浪費的時間，而且放在網路上也可以被更多的人看見。

關鍵詞：線帶、縫合、虛擬實境

¹ 專案: <https://github.com/vr-paint/brush>

Abstract

This project hopes to be able to perform 3D drawing in virtual reality. The user will enter the VR space to draw. After many actual measurements, it is found that the user will have cognitive difficulties and burdens in this space. After consideration, this project decided to implement the program to automatically fill and correct the gap between the lines to reduce the difficulty of users in playing. It is the development goal of this research to provide a platform where they can use their creativity.

The development program used in Unity3D, which is combined with SteamVR for interface settings and handle control. And make the relevant function.

The stitching function is to facilitates the user's drawing function, and the method is relatively complicated. The basic definition is that if the distance between the two lines is close enough, the program will automatically stitch them into a plane to reduce the generation of excess gaps and reduce the difficulty of drawing. However, doing so will make the data point is repeatedly accessed and searched for the distance of the similar data point, which causes the program to slow down or even freeze.

To solve this problem, this project decided to use KdTree to solve this problem. This quick search method can reduce the number of executions to retrieve data points, and put the needed data points into KdTree through classification. Because after multiple levels of classification, retrieval will be much faster, and the same program will not be executed multiple times, which can effectively reduce the burden on the program. The final product expects that the user can draw a complete picture and smooth the gaps between the seams.

At the beginning of the research, when working on Unity projects, we often encountered version control problems, so this project finally solved this problem using GitHub. GitHub is a software owned by Microsoft that specializes in uploading projects and project version control. The correct update project can save a lot of time wasted by version errors, and it can be seen by more people on the Internet.

Keywords: Streamers, Suture, VR

目錄

摘要.....	i
目錄.....	iii
圖目錄.....	iv
第一章 前言.....	1
第一節 研究目的.....	1
第二節 研究問題.....	1
第二章 文獻探討與回顧.....	2
第一節 文獻探討.....	2
第二節 相關探討.....	3
第三章 研究方法及步驟.....	4
第一節 使用者介面.....	4
第二節 使用者操作.....	5
第三節 修改工具.....	8
第四節 從座標系統畫出線帶.....	11
第五節 色環與線帶色彩.....	13
第六節 線帶縫合.....	15
第七節 用 GitHub 做版本控制.....	16
第四章 研究結果.....	18
第五章 未來展望.....	19
參考文獻.....	20

圖目錄

圖 1 本研究在 TiltBrush 繪畫出了幾張作品	3
圖 2 使用者拿著握把	4
圖 3 可用油漆刷進行繪製	4
圖 4 介面示意圖（雙手）	4
圖 5 介面示意圖（左手）	4
圖 6 手把的各項功能	5
圖 7 手把上 Hair Trigger 鍵和 Grip 鍵的功能介紹	5
圖 8 調色盤	6
圖 9 右手的操作細節	6
圖 10 左手觸控板操作細節	7
圖 11 左手的操作細節	7
圖 12 清除資料 icon	8
圖 13 返回鍵 icon	8
圖 14 返回鍵 icon	9
圖 15 橡皮擦 icon	9
圖 16 選擇橡皮擦功能，白色的圓球出現在右手把前面	10
圖 17 白色圓球刪除掉了右邊的線條	10
圖 18 左手座標系統	11
圖 19 嘗試實作線帶中軸線及立體生成	11
圖 20 產生的線帶是 quad 組合而成	12
圖 21 手把畫出線帶	13
圖 22 觸控板的 Axis 值	13
圖 23 RGB 色環	13
圖 24 改變線帶色彩	14
圖 25 改變線帶色彩飽和度	14
圖 26 三條距離不同的線帶	15
圖 27 將兩條相鄰線帶縫合	15
圖 28 原始線帶座標點	16
圖 29 相近點中位座標	16
圖 30 縫合線帶座標點	16
圖 31 距離相近縫合	16
圖 32 距離較遠不縫合	16
圖 33 GitHub 專案更新指令	17
圖 34 繪畫成果	18
圖 35 vr-paint/brush	18

第一章 前言

第一節 研究目的

本研究之所以會選擇這個主題，是來自繪畫的濃厚興趣。狂熱且執著地尋找著各種可以繪畫的地方，白紙、手掌、桌面、黑板等等，都可以是畫紙。

除了這些熟悉的平面繪畫方式，在看過許多其他的論文及作品之後，嘗試使用 VR 裝置遊玩 TiltBrush 時，發現到竟然能運用手把在 3D 空間自由的繪畫！這拓寬了本研究對繪畫的想法。比起傳統的 2D 繪畫必須藉由陰影的效果才會有立體感，VR 世界能夠更快的展現出實體的 3D 樣貌。雖然 TiltBrush 有著許多不同的繪畫功能，以及產生許多炫麗的光彩，唯一的缺憾就是不能產生 3D 物件。

因此選擇的主題是 3D 繪圖，利用 VR 裝置在虛擬空間中使用畫筆繪畫，藉此製作出有立體感的繪圖。畫作的構成預期是由線條組成，本研究打算以 VR 手把作為畫筆，來繪畫出基本的帶狀線條，簡稱為線帶；同時本研究想到若結合漆刷式繪畫功能，繪出物體的表面，就可以藉由表面縫合成出想要的 3D 物件。

所以修補線帶與線帶間的空白來形成 3D 模組成為任務之一。目標在完成之後，能夠整合整張畫作，使其轉變成一個物件。使用者能夠自由的在 3D 空間中做畫，以此激發使用者的想像力和創意，繪畫出與眾不同的作品。

第二節 研究問題

主要針對四大問題來進行研究：(1)判斷在虛擬空間中畫線的位置與方向；(2)如何建構出正確的色環介面；(3)將以畫出的線帶縫合成平面；(4)在完成作畫後，讓畫作變成物件。

若解決這些問題能夠避免使用者在虛擬空間中遇到的方向感錯亂問題，並且避免畫出的畫作過於單調或是無法精確修正所產生的缺憾問題。

第二章 文獻探討與回顧

第一節 文獻探討

本研究主要發揮在 VR 空間中，因此參考銘傳大學中與 VR 有關的三篇論文，例如謝其叡等人的陶藝與浮雕[1]、曾俞豪的虛擬雕塑系統[2]、許志遙等人發表的 VR 3D 繪本[3]。

前兩者是在 VR 空間中對 3D 模型的研究，其中前者目標在於對基礎物件的修改，從而作出對細節的修正；後者以堆疊方塊的概念建立架構，利用拖拉座標點進行形狀變形。而 VR 繪本則專注在 VR 空間中的繪畫上，以簡單的方式快速在 3D 表面上著色。本研究參考這三篇論文，可惜的是三篇論文都建立在有實體模型才能進行創作。

而 SIGGRAPH 2019 有一篇 Enrique Rosales 發表的 SurfaceBrush[4]，本研究看中它對物體表面進行繪畫的能力。以此為基礎，延伸找出另一篇在成大的論文 Ontlus[5]，兩篇論文以操作 VR 手把為主，來建立出 3D 模型。區別在前者是以線帶將表面畫出來後建立模型，而後者是由內而外的雕刻模型，因此前者較後者更為直覺且容易使用。以上兩篇論文使本研究對於 3D 模型建構的方式有更進一步的了解。

而關於 VR 的繪畫在更早之前也有 2007 年 D Keefe 發表的 Drawing on Air[6]，因為那個時候 VR 裝置尚未普及，所以比較少人知道。為了畫出立體線條，他用自己設計的裝置來進行 3D 線段的建構，讓設計出來的 3D 繪畫可以展現給他人觀賞。

對於可以畫出多種類的線帶問題，本研究也參考 S. Schkolne 等人發表的 SurfaceDrawing[7]這篇論文。他們主要是研究在 3D 世界中畫出各種各樣的線帶，本研究的目標也是要畫出與其相似的線條樣式，所以重要性不言而喻。

建構完線帶之後，開始參考 S.-H. Bae 等人發表的 ILoveSketch[8]，該論文研究的是修正使用者畫出的多餘線帶，將其清除，使作畫更為清晰。他們對於線條的處理有著自己獨到的做法，本研究認為參考他們的做法可以強化對線條的認知，使對目標的清晰度提升。

第二節 相關探討

在 2017 年，TiltBrush 的上市讓使用者能夠直接在 VR 的世界進行繪畫，是 VR 世界中最常被使用的工具之一，並且被 Google 收購後進行多次的更新。在 TiltBrush 擁有著多種繪筆以及效果，這特殊畫筆型式完全顛覆了我對 3D 繪圖的想象(圖 1)。在試玩之後，見識到頂尖工程師為 VR 介面設計的繪圖技術，促使我也想要努力的去完成線帶縫合的研究，希望可以完成那樣精彩的作品。

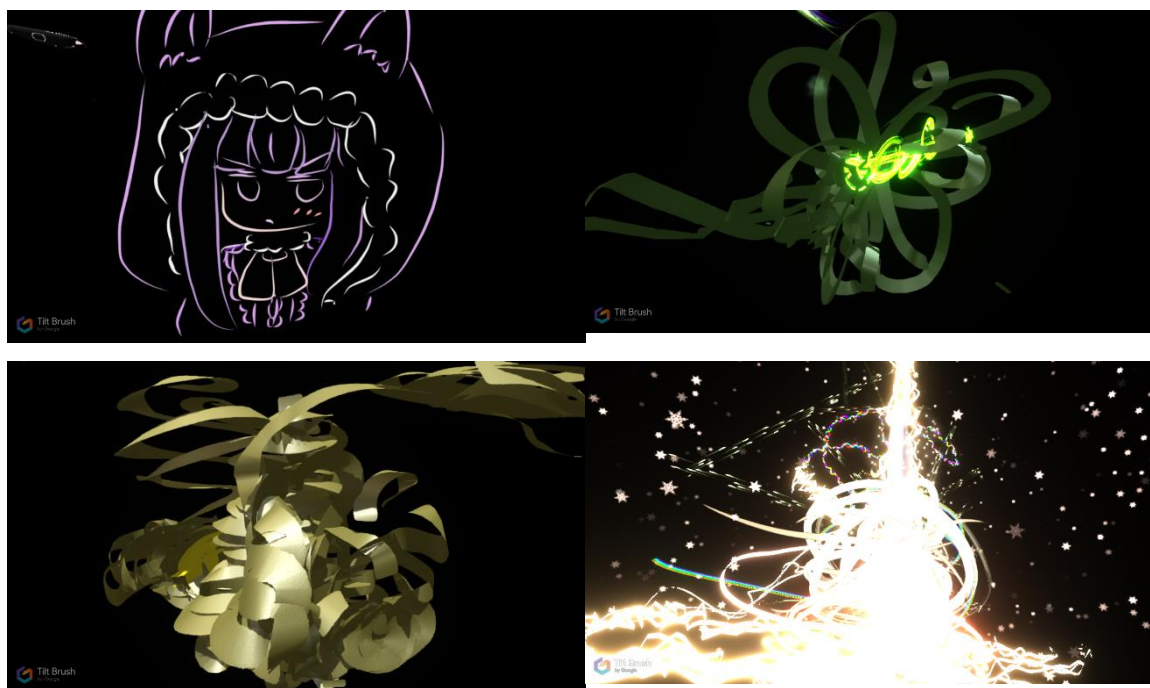


圖 1 本研究在 TiltBrush 繪畫出了幾張作品

遊玩時嘗試過了所有畫筆以及功能，之後總結了這款繪畫遊戲以下的優缺點。優點就是畫筆種類與系統的高穩定性，且運用了流體，閃光，反光物體等困難的技術在他們的線帶上。而缺點則是發現到 TiltBrush 的線條都是一個一個的物件，導致刪除的時候也是一個一個物件回收，所以不能使用橡皮擦來改變線條的形狀，最終無法對線條做出細節的修改，沒辦法修改出想要的線條來畫出理想的畫作。

本研究將 TiltBrush 當作主要的參考對象，所以打算吸收 TiltBrush 的優點，以此為基礎增加新的功能，並以新的特色作為專研的研究方向與目標。而前面講述的擦除問題，我打算利用細節雕刻的方法來解決。所謂的細節雕刻就是通過細化物體表面之後，能夠藉由消除細小的區域，讓作品繪畫更細緻。同時也保留回收線條的功能，後期便可以做到對細節的小修改與大面積線條的回收。

第三章 研究方法及步驟

第一節 使用者介面

本研究使用 VIVE 進行實作，下面(圖 2)、(圖 3)為使用者在 VR 世界繪畫的示意圖，使用者手拿著 VR 手把，頭上戴著 VR 顯示器。藉由油漆刷工具，線條在 VR 世界會以 3D 帶狀的方式呈現，線帶本身就具有一定的寬度，會讓使用者感覺更為立體，也可以做出纏繞效果。



圖 2 使用者拿著握把



圖 3 可用油漆刷進行繪製

下圖為本研究的使用者介面(圖 4、圖 5)。左手選擇簡易的 UI，有三個功能：顏色選擇、位置移動、線帶擦除，透過觸控板可以進行功能之間的切換；右手則沒有 UI，其上方觸控板的方向鍵可以直接控制線帶粗細。縫合功能是搭載在右手的程式碼當中，在使用者繪圖時，會根據程式碼內建的判定進行縫合作業。



圖 4 介面示意圖（雙手）



圖 5 介面示意圖（左手）

第二節 使用者操作

本研究打算使用手把的板機(HairTrigger)與觸控板(TouchPad)來操控介面(圖 6)，右手手把板機的按鍵力道可以決定線帶的粗細，手指輕觸手把板機鍵可產生細長線條，若用力一點觸碰即可加粗線帶，藉此來方便使用者自由的更改。

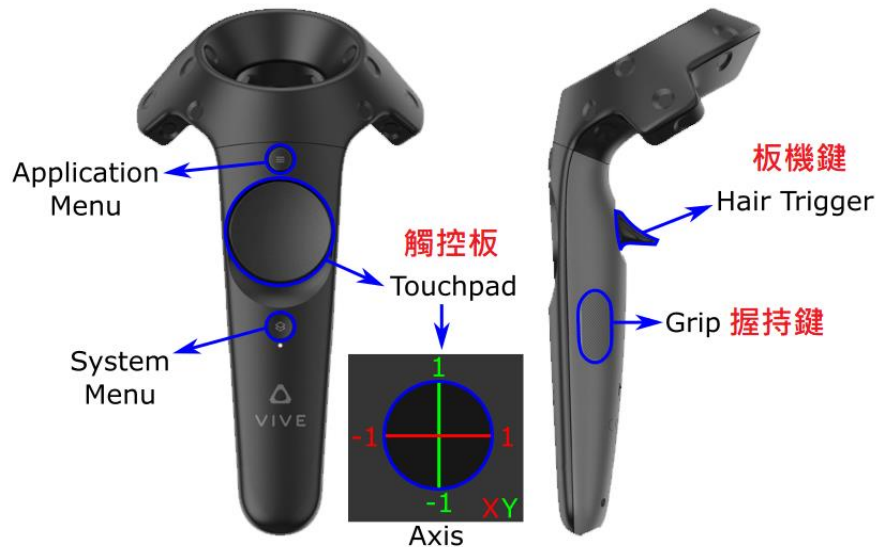


圖 6 手把的各項功能

除此之外，本系統利用左手手把操控 UI 介面，只要拿起手把就可以用觸控板來切換各項功能，依序用觸控板左右切換調色盤、返回鍵、線帶擦除、移動位置、清除資料、還原鍵，在移動到想要的選項後，左手扣下板機鍵即可選取該功能(圖 7)，而若選擇調色盤，即可跳出調色盤選單(圖 8)，手指滑動觸控板可以選取顏色，同時中間方塊的顏色會隨著手指滑動到的位置改變。



圖 7 手把上 Hair Trigger 鍵和 Grip 鍵的功能介紹

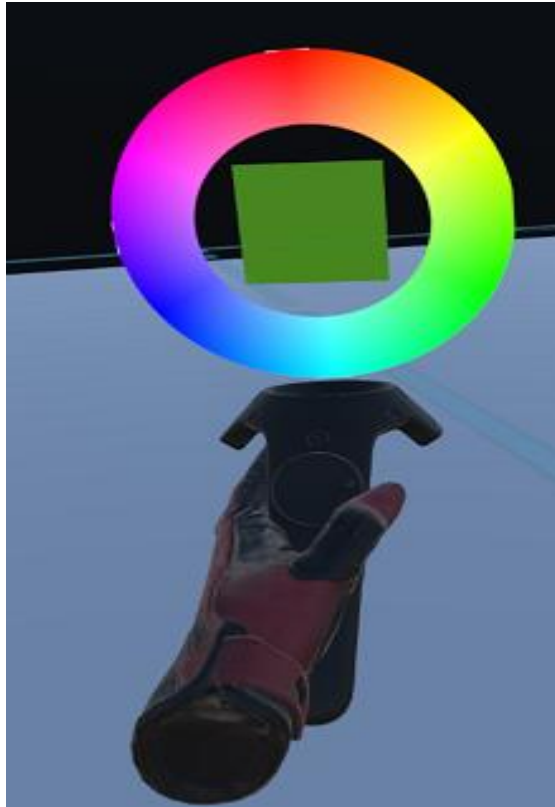


圖 8 調色盤

本研究在使用者進入 3D 空間當中，其中一面牆上會將下圖的操作介面顯示出來，方便使用者快速理解操作。

右手操作如(圖 9)僅使用扳機鍵便可完成所有操作，按下扳機鍵可以直接在 3D 世界中產生線條。然後在改變扣下扳機鍵的力道的情況下，則可以改變線條的粗細。

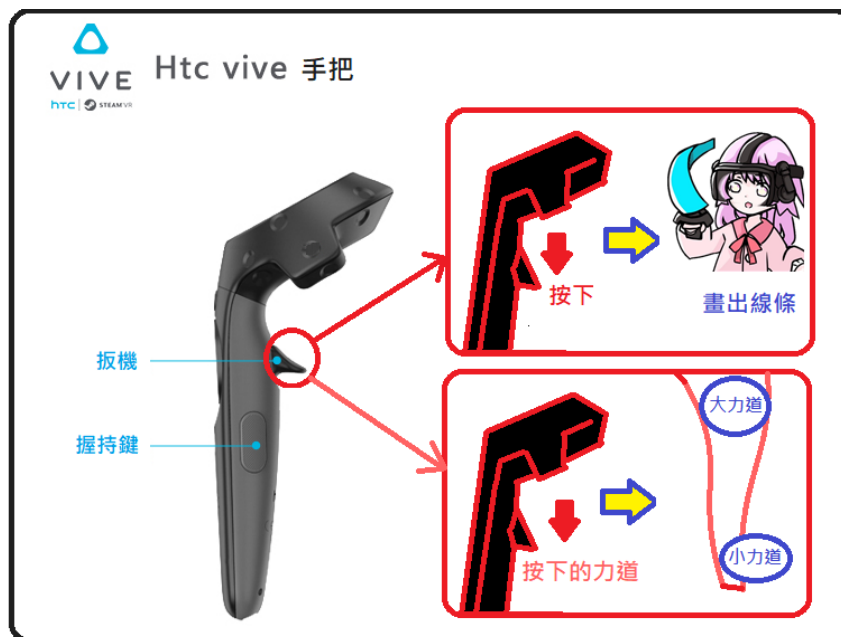


圖 9 右手的操作細節

左手操作如(圖 10、圖 11)使用到了觸控板，扳機鍵以及握持鍵三個鍵位。使用觸控板的左右兩側可以切換使用者界面的 icon，在進入調色盤功能後按照觸控板上手指所在的座標，算出與 x 軸的夾角度，可以同步決定色環上的顏色。為了減少使用者操作的麻煩，左手扳機同時代表使用功能與退出功能，在進入切換顏色的功能後，扳機鍵則會轉變成退出的功能，統一操作會使操作介面方式比較容易理解。最後是握持鍵，其功能是進入調整色度的介面，因為繼續使用扳機鍵會讓使用者混淆，所以最終決定使用握持鍵來進入改變色度的功能。



圖 10 左手觸控板操作細節

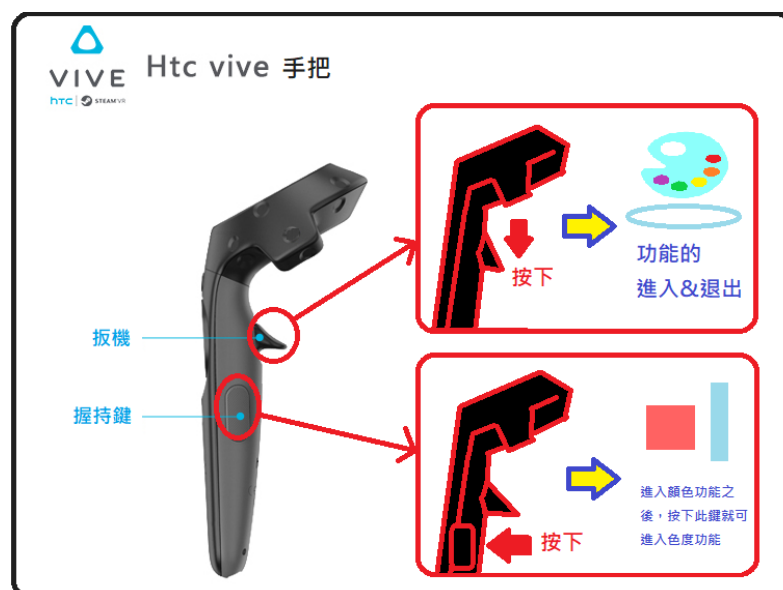


圖 11 左手的操作細節

第三節 修改工具

有鑒於畫出線帶之後，會想要修改或者清除該條線帶，所以我特別做了三個修改工具供使用者使用，分別為清除資料、返回鍵、橡皮擦擦拭這三大工具，能夠有效的輔助使用者的繪畫需求。

1. 清除資料(圖 12)：將 VR 空間中所有的線帶全部刪除，並且同時清空其資料點。操作方法是使用觸控板切換 icon，選擇標示 clearall 的符號，按下扳機鍵即可刪除所有線帶。



圖 12 清除資料 icon

2. 還原資料(圖 13)：將 VR 空間中所有的線帶全部復原，此操作僅可以在使用 clearall 功能後使用。這個功能保存了被清空資料點的位置資料，在使用功能時將這些資料點再次找回並使用它。操作方法是使用觸控板切換 icon，選擇標示 undo 的符號，按下扳機鍵即可恢復所有的線帶。



圖 13 返回鍵 icon

3. 返回鍵(圖 14)：將 VR 空間中的最後一條線帶刪除，這個功能對於想要不斷重新修同一條線帶的使用者來說比較方便，是實用的一種繪畫功能。操作方法是使用觸控板切換 icon，選擇標示 undo 的符號，按下扳機鍵即可刪除最後一條線帶。



圖 14 返回鍵 icon

4. 橡皮擦擦拭(圖 15)：呼叫一個 sphere 物件與 VR 空間中的任意一條線帶，通過距離之間的計算，達到刪除指定線帶的效果。操作方法是使用觸控板切換 icon，選擇標示 eraser 的符號，按下扳機鍵後左手的前方會出現一個 sphere 物件(圖 16)，使用此圖形可以刪除觸碰到的線帶(圖 17)，再按一次扳機鍵即可隱藏掉 sphere 物件。



圖 15 橡皮擦 icon



圖 16 選擇橡皮擦功能，白色的圓球出現在右手把前面



圖 17 白色圓球刪除掉了右邊的線條

第四節 從座標系統畫出線帶

在 VR 世界中本研究會使用一個 3D 座標系統來畫出線帶，先簡單說明座標系統，以左手座標系統為例(圖 18)，其中紅色 Tangent 向量與藍色 Normal 向量是兩個互相垂直的方向向量，而用此兩個向量做外積形成第三個綠色 Bi-Normal[4]向量，此三個向量是 3D 空間中相互垂直的方向向量，用此產生的座標系統來形成線帶。

將這些觀念運用在物件上的結果，讓每個 3D 物件都會具有上述的三種向量，以更好在空間的狀態下理解，並且撰寫物件的程式碼後，能夠藉由操作在資料點上的座標向量，來靈活做出不同角度的線帶。

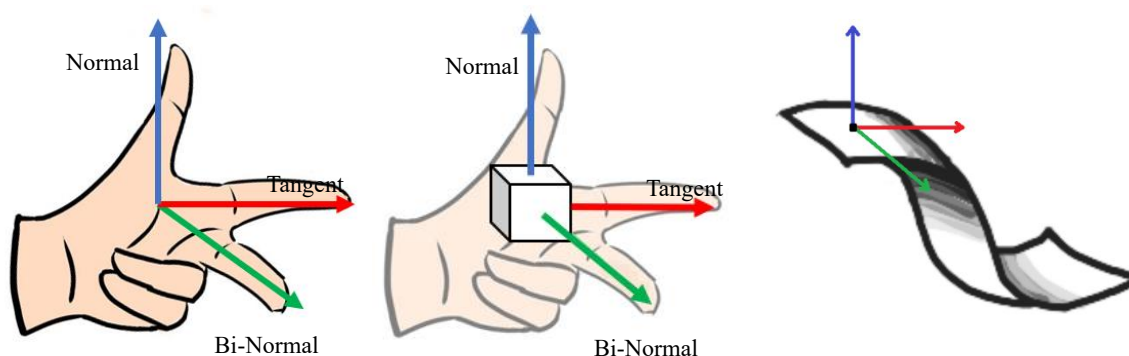


圖 18 左手座標系統

為了能夠在 VR 空間中使用刷的方式繪圖，本研究從電腦 2D 的畫圖為基礎往 3D 方向邁進。以油漆刷為例，為了在空間中能夠精確畫出一段有寬度的曲線，本研究以基礎三個步驟來確定成果(圖 19)。

為了畫出一條有寬度且有轉折的線帶為目標，本研究將座標形成線段所需的資料點，使用鏈結串列的方式進行紀錄。為了減少系統效能的過度負擔，本研究選擇減少記錄過多的點座標，且將點座標改以等距的方式作記錄。同時目標是要在 VR 空間中進行漆刷式繪畫，因此利用點座標上的向量座標系統，讓線段產生出一個水平延展的平面，以成為一個宛如彩帶般的立體線帶。

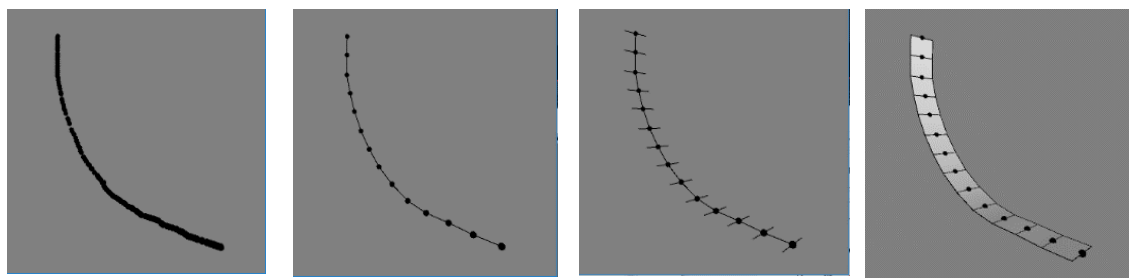


圖 19 嘗試實作線帶中軸線及立體生成

綜上所述，在 VR 世界中形成線帶，需要點座標的 Tangent 向量以及 Normal 向量，用來算出延展平面的 Bi-Normal 向量。在 Unity 空間中，VR 手把本身就擁有自己的座標與向量，而在扣下手把板機時可以抓取手把在空間中的座標位置與座標向量，因此本研究使用手把的參數來產生線帶。

在 Unity 的手把上的三個向量：forward、up、right 向量，當使用 forward 與 up 向量做外積，可以形成延展平面所需的 Bi-Normal 向量，當使用者拖曳手把畫線時，能夠產生對應的等距座標點，在座標點產生的當下就可連接點與點產生四個三角形組成的平面(圖 20)。

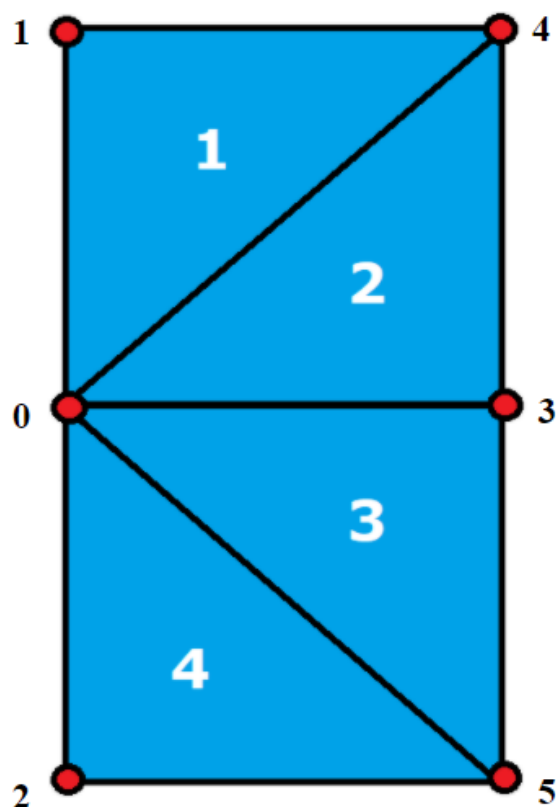


圖 20 產生的線帶是 quad 組合而成

最後利用手把的 right 向量，用以產生法向量，來畫出所需的立體線帶(圖 21)。當使用者拖曳手把畫線時，就能夠運用此向量來延展平面，畫出所需的立體線帶。順帶一提，手把上的 right 向量就是延展平面所需的 Bi-Normal 向量，因此直接使用 right 向量仍可以畫出相同的結果。

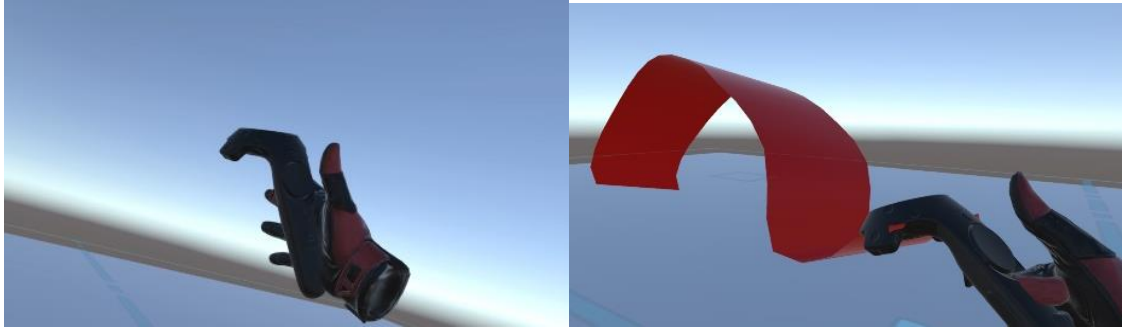


圖 21 手把畫出線帶

第五節 色環與線帶色彩

本研究為了做出色彩線帶，直接使用 VR 手把上觸控板的 Axis 值(圖 22)，運算出 HSV 色彩空間的色相，以做出目標的 RGB 色環之色彩(圖 23)。

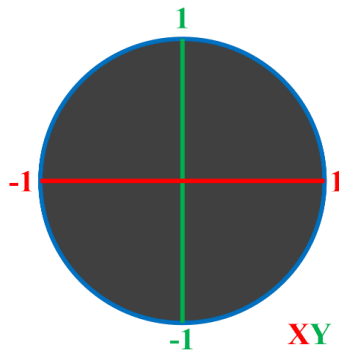


圖 22 觸控板的 Axis 值



圖 23 RGB 色環

為了直覺的修改顏色，需要觸控板與色環做相互對應，才能讓使用者操控觸控板的同時可以改變顏色。為此利用觸控板上的 Axis 值，並利用 atan2 函數將 axis 值轉換成角度，以確認使用者轉動的角度。而為了完全對應到目標色環上的顏色位置，按照切線角度決定顏色，還需再將角度值作轉換。由於 atan2 函數的角度值是與使用的色環呈現相反的狀態，所以需要乘負數使其轉往另一個方向。由於目標色環的起始點在上方，所以再加上 $\pi/2$ 來對應色環上的顏色。

$$Hue = \frac{\left[-\text{atan2}(Axis.y, Axis.x) + \frac{\pi}{2} \right]}{2\pi} \quad (1)$$

但知道了正確的轉動角度還不夠，還需再將角度值進行換算，使其符合 HSV 中的色相值。由於 HSV 的色相值介於 0~1 之間，因此將角度值除以 2π ，會使數值變為百分比的形式限制在 0~1 之間。同時 atan2 角度值會有正負號問題，所以當輸入的角度值為負的時候，將此值+1，使其避免輸入色相的數值有負號。

HSV 即色相、飽和度、明度，RGB 則是紅綠藍的三原色光模式，將 HSV 轉換成 RGB，可以更清楚對應 RGB 色環，手把觸控板滑動到色環的哪邊，就自動切換到對應的 RGB 色環顏色，色環介面中間的方塊也會同步改變顏色，讓使用者能夠直覺簡單地切換自己想要的顏色(圖 24)。

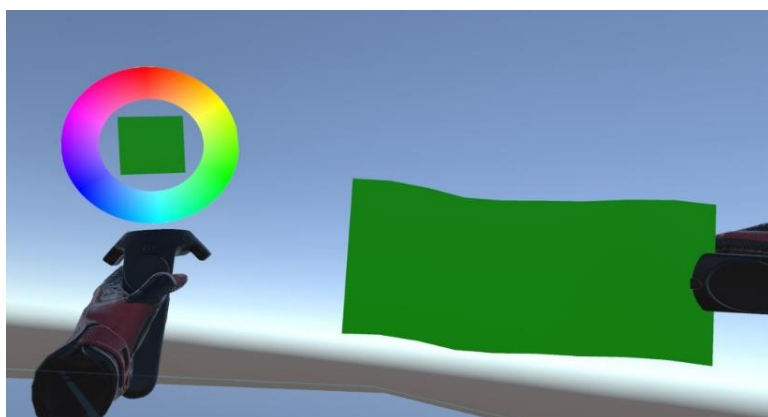


圖 24 改變線帶色彩

也可以按下握持鍵進入改變飽和度介面，飽和度簡單來說就是在同一個顏色下，不同深淺的色彩樣式，進入切換飽和度的介面之後，為了方便使用者操作，我選用了和改變顏色相同的操作方式。這個功能可以增加作品的層次感和美感，也能做出更多的繪畫效果，實際呈現效果如(圖 25)。

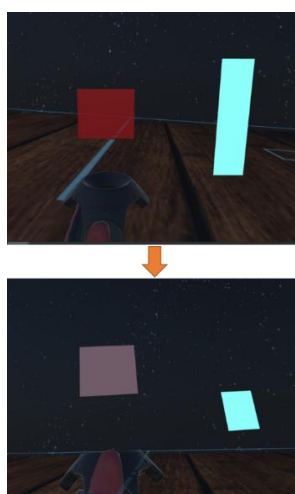


圖 25 改變線帶色彩飽和度

第六節 線帶縫合

考慮到使用者在 3D 世界中進行繪畫時，會因為視覺上的誤差，導致難以將兩條線帶重合。本研究決定使用自動縫合空隙來解決這個問題，除了減輕使用者在 3D 繪圖上的一些困難，還可以方便繪畫完畢後的成品能以一整個整體的方式展現(圖 26、圖 27)。

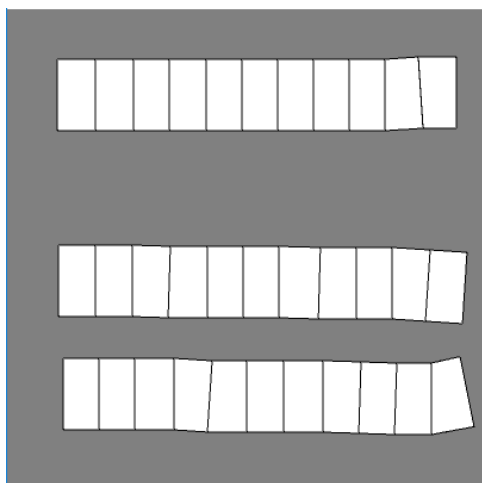


圖 26 三條距離不同的線帶

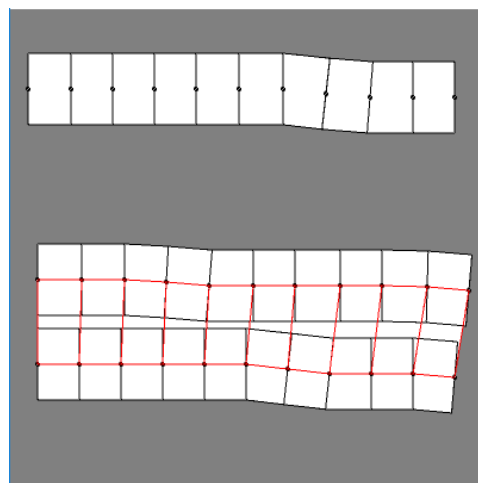


圖 27 將兩條相鄰線帶縫合

開始使用 Unity 進行 3D 實作後，本研究利用鏈結陣列找出是哪兩條線帶的座標點需要進行縫合。首先做出一個可擴張的雙層鏈結陣列，可以將所有新增線帶的座標點都可以讀入該陣列中。

而在做縫合之前要先進行座標點之間的距離判斷，判斷後就可以將兩條距離接近的線帶給找出來。距離判斷本研究使用了 `Vector3.Distance` 函式，這個函式可以計算兩點之間的距離，通過計算出來的距離來決定線帶合線帶的資料點之間是否需要縫合。本研究所設定資料點之間需要縫合的距離為 0.05，所以點與點之間的長度小於 0.05 便會進行縫合。

縫合部分使用的方法十分直觀，最快的方法就是將點合併(圖 28)。在找到距離相近的座標點後，根據距離算出其對應的兩個座標點的中位座標，利用此座標代替原本的最近點座標(圖 29)，這樣便可以縫合出一個完整的平面，能夠縫合兩條線帶間的空隙(圖 30、圖 31)。而如果若距離太遠則維持原樣不進行縫合(圖 32)。



圖 28 原始線帶座標點

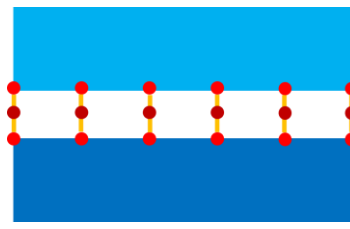


圖 29 相近點中位座標



圖 30 縫合線帶座標點

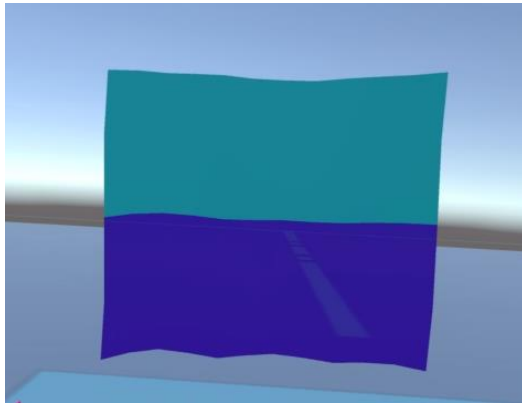


圖 31 距離相近縫合

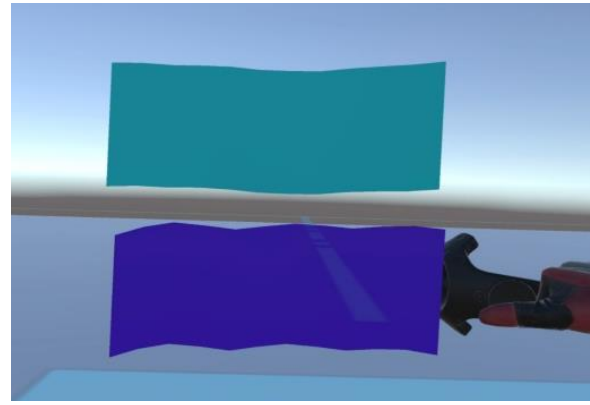


圖 32 距離較遠不縫合

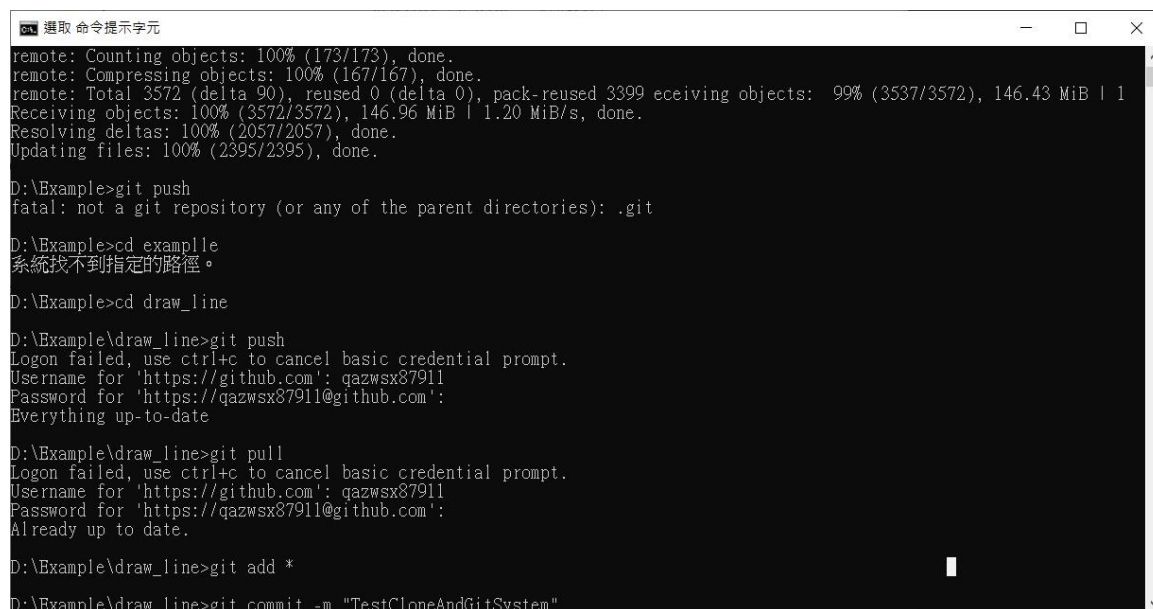
在實作時，傳統陣列結構的運算速度太慢，導致縫合功能造成延遲的狀態。所以透過老師指導得知了 KdTree。本研究利用 KdTree 建立二元樹，並以此搜尋最近點，運用此資料結構進行實作解決速度過慢的問題。本研究將第四章的座標系統來延伸利用，將每段線帶上的座標點套進 KdTree 來建樹，就能在每條線帶上面實作出 KdTree。利用這個方式，本研究再利用 KdTree 內建的搜尋方式，找出最近點。就可以改善縫合運算速度過慢的問題。

於是根據本研究畫線帶的邏輯，將 KdTree 的建樹過程放在線帶產生完畢時，這樣當畫出新的線帶，在途中 KdTree 就能夠偵測每一個座標點的距離是否達到縫合標準，就能在畫新的線帶的同時進行縫合功能，這樣不只可以更直觀上的展示縫合的效果，也能夠因此避免一次套入整條線帶進行搜尋產生的系統運算時間。

第七節 用 GitHub 做版本控制

開發過程中遇到了兩個重大的問題，一個是有關於版本控制的問題，在 Unity 的版本差異導致的程式碼開發問題，另一個則是在進行線上更新的時候，會容易出現程式開發同步問題。同步問題最具體的出現在多電腦開發的狀態，往往會因為忘記上傳更新導致功能開發多一塊少一塊。

本專案使用 GitHub 做為解決方式。利用了 GitHub 線上儲存功能，在電腦安裝需求軟體後，就能夠利用 CMD 來簡單地進行檔案的上傳與更新。這樣既可以解決版本不相符的問題，也可以減少在不同 PC 上下載新修改的檔案所需要的時間。



```
remote: Counting objects: 100% (173/173), done.
remote: Compressing objects: 100% (167/167), done.
remote: Total 3572 (delta 90), reused 0 (delta 0), pack-reused 3399 eceiving objects: 99% (3537/3572), 146.43 MiB | 1
Receiving objects: 100% (3572/3572), 146.96 MiB | 1.20 MiB/s, done.
Resolving deltas: 100% (2057/2057), done.
Updating files: 100% (2395/2395), done.

D:\Example>git push
fatal: not a git repository (or any of the parent directories): .git

D:\Example>cd example
系統找不到指定的路徑。

D:\Example>cd draw_line

D:\Example\draw_line>git push
Logon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': qazwsx87911
Password for 'https://qazwsx87911@github.com':
Everything up-to-date

D:\Example\draw_line>git pull
Logon failed, use ctrl+c to cancel basic credential prompt.
Username for 'https://github.com': qazwsx87911
Password for 'https://qazwsx87911@github.com':
Already up to date.

D:\Example\draw_line>git add *

D:\Example\draw_line>git commit -m "TestCloneAndGitSystem"
```

圖 33 GitHub 專案更新指令

同時也因為 GitHub 在上傳之後，會自動對版本進行還原點設定，所以也可以減少一般在雲端硬碟上面的儲存空間的需求。考慮到上面的幾點，我認為將 GitHub 與專案連結是十分有效率且有益的。

第四章 研究結果

本研究可以進行 VR 漆刷繪畫，提供了簡易的繪畫工具，讓使用者創作作品(圖 34)。尤其著重於空隙縫合，透過將不同大小的線帶縫合成平面，讓畫作的完整性更高。同時也設計讓使用者根據扣下手把板機鍵的力道來更改線帶粗細的功能。

而在繪畫工具部分，大致分成三種。第一種是顏色切換工具，在繪畫中基礎的功能就是可以選擇多種顏色進行繪畫，所以提供了自由選擇顏色的工具。

第二種便是修改工具，此工具有四項功能：

1. 能夠刪除指定線帶。
2. 刪除最後一條線帶。
3. 可以刪除全部線帶。
4. 回復全刪除之前的所有線帶。

第三種是傳送功能，在 VR 世界中繪圖，因為使用時需要不斷移動才可以繪畫出完整的圖，所以增加了可以傳送到任意位置的移動功能。

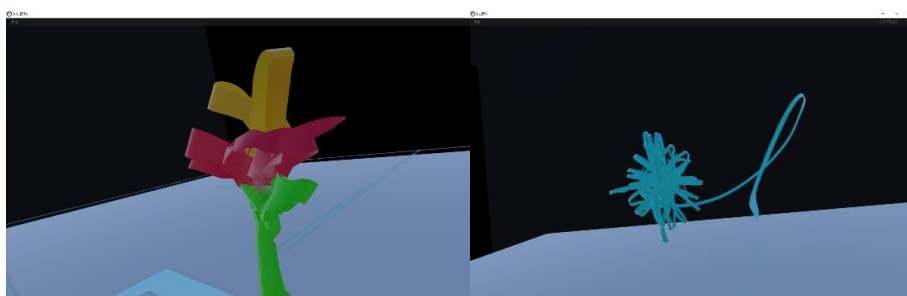


圖 34 繪畫成果

除此之外，本研究值得提及的是使用 GitHub 在進行版本控制，並且在其展示了程式碼以及其他相關檔案與連結。且在 GitHub 專案的 README 中，本研究進行了簡單的敘述來說明此專案的使用方法與用途。

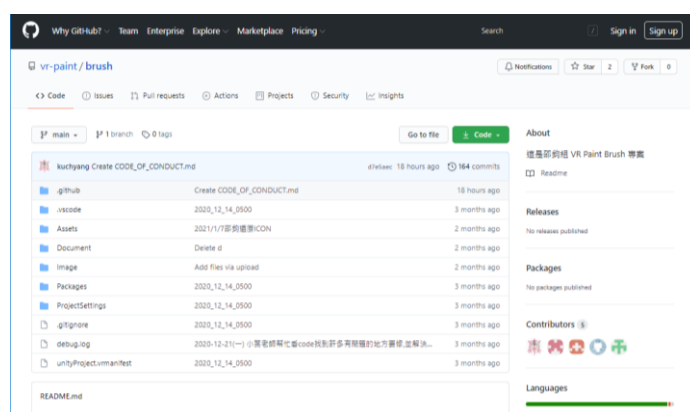


圖 35 vr-paint/brush

第五章 未來展望

目前已經完成了 VR 漆刷繪畫系統，除了完成簡易 VR 繪畫工具外，本系統還做了線帶縫合這項功能，輔助使用者建構 3D 模型，但建構模型後可能會想要修改模型表面細節，因此未來首要目標是細緻化雕刻，期望能用雕刻工具改變線帶表面的凹凸，來改變模型的形狀。

而在使用者介面上，不少人建議說 UI 介面不夠直覺，除了 ICON 不夠明確外，還須要讓使用者更加了解如何使用諸多功能。也建議若能夠將使用者功能做成物件放置在置物桌上，則可以捨棄左手手把，能夠直接使用右手把選擇想要的功能，手把接觸到即可切換，更能直覺明確的操作 3D 空間的繪畫。

最後，希望在完成畫作後可以儲存並留著資料匯出展示，供使用者回味自己所創造的作品。若覺得作品還需要更改，也可以把作品轉換成一個物件來儲存，作品就能在需要的時候拿出來在作改良，不會因為關閉而遺失資料。

本研究在 GitHub 這個平台上發表了作品，期待成果可以被更多人看見。在完成此次科技部大專生研究計畫後，可以透過 GitHub 繼續看到本人持續對此專案的完善與修改，看到此專案更大的可能性。

參考文獻

- [1] 謝其勸, 薛猷騰, 何誼庭, 黃慧緣, 呂昱辰, and 葉正聖, "陶藝與浮雕: Leap Motion 結合 VR 之互動塑模," presented at the Computer Graphics Workshop, 台中, 2017.
- [2] C. Tseng and J.-S. Yeh, "A Kinect-based System for Virtual Sculpture," presented at the Computer Graphics Workshop 台北, 2015.
- [3] 許志遙, 蔡閔鈞, 林伯儒, 邱俊澄, 呂昱辰, and 葉正聖, "以 HTC Vive 為基礎 VR 3D 繪本," presented at the Computer Graphics Workshop 台北, 2016.
- [4] E. Rosales, J. Rodriguez, and A. SHEFFER, "SurfaceBrush: from virtual reality drawings to manifold surfaces," *ACM Trans. Graph.*, vol. 38, no. 4, p. Article 96, 2019.
- [5] C.-W. Chen, J.-W. Peng, C.-M. Kuo, M.-C. Hu, and Y.-C. Tseng, "Ontlus: 3d content collaborative creation via virtual reality," in *International Conference on Multimedia Modeling*, 2018: Springer, pp. 386-389.
- [6] D. Keefe, R. Zeleznik, and D. Laidlaw, "Drawing on Air: Input Techniques for Controlled 3D Line Illustration," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 5, pp. 1067-1081, 2007.
- [7] S. Schkolne, M. Pruett, and P. Schröder, "Surface drawing: creating organic 3D shapes with the hand and tangible tools," presented at the Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, Seattle, Washington, USA, 2001. [Online]. Available: <https://doi-org.erm.lib.mcu.edu.tw/10.1145/365024.365114>.
- [8] S.-H. Bae, R. Balakrishnan, and K. Singh, "ILoveSketch: as-natural-as-possible sketching system for creating 3d curve models," presented at the Proceedings of the 21st annual ACM symposium on User interface software and technology, Monterey, CA, USA, 2008. [Online]. Available: <https://doi-org.erm.lib.mcu.edu.tw/10.1145/1449715.1449740>.