

國立嘉義大學資訊工程學系
計算機專題報告

Department of Computer Science and
Information Engineering
National Chiayi University
Computer Project Report

舞尬 Python：混合實境舞蹈學習系統

指導教授： 盧天麒 老師

年度： 一百一十二學年度

組別： 347-113-11

學生： 1094842 蕭張媛

1092913 黃郁晴

中華民國 一百一十二 年 十二 月

國立嘉義大學資訊工程學系 計算機專題報告推薦書

國立嘉義大學資訊工程學系

蕭張媛、黃郁晴 君

所提之計算機專題報告(題目)：

舞尬 Python：混合實境舞蹈學習系統

係由本人指導撰述，經審核同意交付本系歸檔留存。

指導教授 _____ (簽章)

系(所)主任 _____ (簽章)

____年____月____日

舞尬 Python：混合實境舞蹈學習系統

指導教授：盧天麒老師 學生：蕭張媛、黃郁晴

國立嘉義大學資訊工程學系

摘要

這是一個革命性的 MR 混合實境舞蹈遊戲系統，專為現代舞蹈愛好者和學習者而設計。

它打破了傳統舞蹈遊戲的硬件限制，使用者只需使用手機進行影像辨識，並搭配 MR 頭戴式眼鏡，即可輕鬆享受遊戲樂趣。在遊戲過程中，提供多種控制方式，包括手把控制器和手部操控，增加了遊戲的多樣性和互動性。使用者可以輕鬆瀏覽曲目，查看舞蹈動畫和詳細資訊，實時獲得舞蹈遊戲反饋。

此系統의 影片轉換功能使所有使用者能夠輕鬆匯入自己喜歡的舞蹈影片，並透過 MotionBERT 機器學習技術將資料自動轉換為可遊玩的舞蹈資料。同時，使用者還可以進行自主調整，實現更加個性化的遊戲體驗。這種創新的設計讓舞蹈愛好者更輕鬆地融入遊戲中，同時擁有更具互動性的學習體驗。

關鍵字：混合實境、虛擬現實、影像辨識、互動遊戲、舞蹈

目錄

| | |
|------------------------------------|----|
| 摘要 | i |
| 目錄 | ii |
| 圖目錄 | iv |
| 表目錄 | vi |
| 第一章、專題發想 | 1 |
| 1.1 研究動機 | 1 |
| 1.2 既有系統探討 | 2 |
| 1.3 目標研究成果 | 2 |
| 第二章、硬體配置 | 3 |
| 2.1 初始配置 | 3 |
| 2.2 教授建議 | 4 |
| 2.3 最終選擇 | 5 |
| 第三章、研究技術 | 7 |
| 3.1 AlphaPose | 7 |
| 3.2 MotionBERT | 8 |
| 3.2.1 影片偵測轉換動畫的問題 | 9 |
| 3.3 Mediapipe (BlazePose) | 11 |
| 3.4 Open Sound Control (OSC) | 12 |
| 第四章、動畫資料處理 | 13 |
| 4.1 MotionBERT 檔案轉換 | 13 |

| | | |
|------------|---------------------------|----|
| 4.2 | 反向動力學(Inverse Kinematics) | 14 |
| 4.3 | 曲譜結構 | 15 |
| 4.4 | 其他取用資料結構 | 17 |
| 第五章、動作辨識 | | 18 |
| 5.1 | 手機 Mediapipe 辨識傳送 | 18 |
| 5.2 | 頭戴顯示器判斷動作準確度 | 20 |
| 第六章、程式運行敘述 | | 21 |
| 6.1 | 影片轉換動畫程式 | 21 |
| 6.2 | 曲譜修改程式 | 22 |
| 6.3 | 全身動作辨識程式 | 23 |
| 6.4 | 主遊戲程式 | 25 |
| 6.4.1 | 舞蹈選擇介面 | 26 |
| 6.4.2 | 舞蹈遊玩介面 | 26 |
| 第七章、未來展望 | | 30 |
| 7.1 | 主遊戲系統 | 30 |
| 7.1.1 | 等級系統及商城介面 | 30 |
| 7.1.2 | 標籤分類 | 30 |
| 7.2 | 網路伺服器 | 31 |
| 第八章、結論 | | 32 |
| 參考文獻 | | 33 |

圖目錄

| | |
|------------------------------------|----|
| 圖 2.1：HTC VIVE PRO2[5] | 3 |
| 圖 2.2：ZED 2I[7] | 4 |
| 圖 2.3：META OCULUS QUEST3[10] | 5 |
| 圖 3.1：ALPHAPOSE 架構圖[13] | 8 |
| 圖 3.2：MOTIONBERT 架構圖[11] | 9 |
| 圖 3.3：ALPHAPOSE 資料 | 10 |
| 圖 4.1：關鍵座標點關係表 | 13 |
| 圖 4.2：偵測觸碰點位置 | 14 |
| 圖 4.3：IK 遊戲物件 | 14 |
| 圖 4.4：IK 設置 | 15 |
| 圖 4.5：VIVE 版本運行畫面 | 16 |
| 圖 4.6：曲譜結構 | 16 |
| 圖 4.7：儲存資料結構 | 17 |
| 圖 5.1：OSC 資料正確接收 | 19 |
| 圖 5.2：動作辨識失敗 | 19 |
| 圖 6.1：影片轉換程式 UI | 21 |
| 圖 6.2：曲譜讀入 UI | 22 |
| 圖 6.3：曲譜查看介面 | 23 |
| 圖 6.4：辨識 APP | 23 |
| 圖 6.5：影像輸入設定 | 24 |

| | |
|----------------------|----|
| 圖 6.6：OSC 傳送設定 | 24 |
| 圖 6.7：APP 運行畫面 | 25 |
| 圖 6.8：最佳成績 | 26 |
| 圖 6.9：舞蹈選擇畫面 | 26 |
| 圖 6.10：校正動作畫面 | 27 |
| 圖 6.11：動畫移動位置 | 27 |
| 圖 6.12：主遊戲遊玩畫面 | 28 |
| 圖 6.13：遊戲結束畫面 | 29 |
| 圖 6.14：手部選單 | 29 |

表目錄

| | |
|----------------------|----|
| 表 3.1：影片處理遮擋對比 | 11 |
| 表 5.1：判定結果對照表 | 20 |

第一章、專題發想

1.1 研究動機

我們進行這項研究的動機源自於研究團隊成員過去學習跳舞的經驗。在這些經驗中，大多數學習方式涉及觀看舞蹈教學影片，然而，這種方式存在一些不便之處。舞蹈影片需要事先下載，並且在進行練習時，需要左右翻轉影片。此外，學習者必須持續調整影片的播放速度以便更好地理解 and 掌握舞蹈動作，學習過程相對冗長且缺乏趣味。

若以遊戲娛樂為主要目的來學習跳舞，目前現有的跳舞遊戲及相關機台普遍存在著精準度和商業需求方面的問題。遊樂場機台通常使用紅外線偵測技術，不論為 2012 年日本的 Dance Evolution[1]或 2023 年中國的舞戰紀[2]皆是，雖然具有低延遲和高準確度的特點，但其操作複雜度及造價較高，大眾較不會僅為運行此遊戲購入，故無法在家自行遊玩。另一方面，現有的家用舞蹈遊戲如 Just Dance 會須搭配手把或手機的陀螺儀[3]進行動作偵測，但這種方式因只能偵測到手部變化，故準確度相對較低。

此外，由於商業考量，舞蹈遊戲製作商通常優先開發熱門曲目的舞蹈動畫和曲譜，而對於較小眾的曲目則關注度較低。這意味著學習者難以在現有遊戲平台上體驗到自行編排的舞蹈，例如學系或班級自主編排的運動會舞蹈。總體而言，現有的學習和娛樂方式對於跳舞愛好者來說，仍存在諸多不盡如人意的問題，因此我們希望通過本研究提供更為便利、有趣且具有創新性的跳舞學習解決方案。

1.2既有系統探討

為了解決上述問題，我們找到了一款名為「OSU!」的電腦音樂遊戲，[4]其獨特之處在於玩家能夠自主創作曲譜，自由選擇製作曲目和難度，並即時遊玩最新的曲目。更為特別的是，玩家能夠公開分享自己的曲譜至社群，實現一人創譜，眾人共樂的理念。此外，玩家可以與其他使用者進行競技對決，比拚分數，為遊戲增添更多趣味性。

1.3目標研究成果

在受到「OSU」的啟發後，我們決定開發一款具有簡易製作舞蹈動畫和自行設定曲譜功能的遊戲。這款系統的特色在於讓使用者即使沒有 3D 建模和動畫技術的背景，也能輕鬆製作屬於自己的舞蹈動畫及遊玩曲譜。在與盧教授的深入討論後，我們決定以目前最新的混合實境技術作為主要的遊戲體驗方式，這能夠讓使用者更深度地沉浸在遊戲的體驗之中。這不僅提供了更有趣和互動性的學習方式，同時也滿足了玩家對於自主創作和個性化體驗的需求。我們希望透過這款遊戲，為跳舞愛好者提供一個全新且令人滿足的學習平台。

亦希望能讓更多人進行遊玩，所以希望能讓使用者能夠更輕鬆地使用系統，在遊玩初始開銷及過程中都能達成，即不使用大家平常較難購入且除了此系統外幾乎很難使用到的設備進行開發，也盡可能地使用無線傳輸避免意外發生。

第二章、硬體配置

2.1 初始配置

我們於 2023 年 5 月確定研究題目並開始進行研究時，我們選擇使用 HTC Vive Pro2 頭戴顯示器和羅技普通視訊攝影機，以滿足實驗室內的「輕鬆遊玩」和「大家都能體驗」的需求。

從官方給的設備規格及使用說明[5]，可以發現 HTC Vive Pro2 有一些優勢，其中包括使用 Windows 系統進行開發、有線連接操作，使得開發過程相對簡單。且在 Unity 上觀看的即為最終輸出後的結果，減少了誤差可能性。此外，我們使用電腦連接視訊攝影機進行人體影像辨識，這意味著辨識後的資料和主遊戲程式本地傳送，使得資料傳輸更即時，減少封包遺失問題。



圖 2.1：HTC Vive Pro2[5]

我們預想其對於研究之 MR 需求一定程度上能夠達成，HTC vive pro2 在頭戴顯示器前方有兩顆鏡頭，此能夠很好的處理人眼視角，讓使用者看到的並非是單純的平面畫面，而可以呈現出立體感，若能經過部分演算法的處理，可能亦能辨識出畫面的深度[6]，可以以此進行手部動作的辨識，不用使用手柄控制器即可進行操作，讓使用者互動體驗更佳。

然而，在研究的過程中，我們發現我們對這個系統有一些錯誤的期望。首先，HTC Vive Pro2 於 2021 年 6 月發布，當時 MR 概念尚未普及，因此主要是為 VR 遊戲而設計，而非我們研究中所需的 MR。這導致實時取得使用者視角畫面的程式存在相當大的問題。其次，最新的官方 MR 程式 SRWorks 於 2020 年更新[6]，而很多資料已經無法支援。第三，前方的雙鏡頭並非專為 MR 即時使用而設計，傳輸速度緩慢、畫面模糊，而在 Unity 中的絕對位置誤差不小，無法即時更新及移動到相對應的位置。導致在實際遊玩中使用體驗非常不舒適，即使是對 VR 相關遊戲及產品有豐富使用經驗的人也容易感到暈眩。

由於純粹的畫面展示已經存在耗能過大、延遲嚴重的問題，單秒刷新數最高僅 36 張/秒，故我們選擇不再繼續增加研究手部動作辨識功能。總括而言，對於這個系統配置，我們只建議在原地且不太做頭部轉動的情況下進行使用。然而，這與我們專題的核心概念相矛盾，因為我們需要持續且即時、大幅地移動進行跳舞動作，因此該系統不適合作為我們研究的主要系統。

2.2 教授建議

在發現這種情況後，盧指導教授建議我們在 VR 眼鏡前方額外固定一個 Zed 2i 深度攝影機[7]。使用此方法的優勢在於，由於官方提供相關 API，因此我們可以相對輕鬆地處理手部互動的判定，同時更即時地獲得使用者視角的



圖 2.2：Zed 2i[7]

資料[8]。此外，若有必要，也可以搭配深度辨識進行影像處理，使混合實境的體驗更加真實。

然而，經過討論後，我們最終選擇不採用此方案。原因在於，VR 頭戴顯示器本身已經需要有線連接，而在舞蹈遊戲的使用需求上已經相對不太方便。在遊玩過程中，使用者需要擔心自己是否會因為連接線而絆倒，或者有可能由於拉扯線段而導致連接斷開等問題。若再額外加上深度攝影機，必然需要增加額外的連接線，類似的問題出現的機率就會提高，這與我們原訂的研究需求不太相符。因此，我們優先考慮使用較為輕便和無線的配置，以確保在遊戲體驗和實用性方面的平衡。

2.3 最終選擇

在整個選擇過程中，我們注意到了蘋果公司推出的 Vision Pro[9]。雖然其預估定價昂貴，且發售時間無法趕上我們專題結案，但在發表中特別強調了頭戴式裝置中的 MR 功能。這讓我們看到各大科技企業已經開始注意到 MR 在未來的商機，也意味著在不久的將來可能會有其他較平價且以 MR 為主要訴求的頭戴式裝置出現。因此，我們持續關注並追蹤相關企業的開發和銷售消息。



圖 2.3：Meta Oculus Quest3[10]

值得慶幸的是，在暑假期間，我們發現了一款即將上市的全新虛擬實境眼鏡，即 Meta Oculus Quest 3[10]。這是 Meta 公司在 Quest 2 這款 VR 眼鏡的基礎上進行升級，專門為 MR

用途進行開發的產品，根據商品資料，它正面有兩顆彩色鏡頭和一顆深度感測儀[10]。在測試中，我們發現它對於 MR 的操作上設置非常完善，具有完整的手部操作辨識系統，並展示實境畫面非常即時，不太會引起暈眩和出現動作畫面不統一的問題。

在實際取得 Meta Oculus Quest 3 後，我們發現了一些與 HTC 不同的特點。首要的差異是處理系統的不同，HTC 使用的是 Windows 系統[5]，而 Meta 使用的是 Android 系統[10]，這意味著我們先前開發的系統幾乎需要重新編寫。其次，其為無線的一體機架構[10]，因此不太建議連接到電腦處理全身人體姿態辨識的資料後進行傳送。雖然一開始我們認為這可能會成為問題，但深思熟慮後，我們發現這可能反而有助於更貼近我們原訂的研究目標，即在隨處都可以輕鬆遊玩的情境下進行使用。而對於此想法，我們希望能不須額外購買其他感測器，因此我們選擇使用手機作為全身人體姿態辨識的媒介，然後使用網路進行無線傳輸，將資料傳送到使用者配戴的頭戴式裝置內。

第三章、研究技術

考慮到我們的預期是能夠自動從影片中獲得 3D 動畫的資料，並直接呈現在使用者面前，我們在討論後決定選擇處理時間較長但動作更為準確的 3D 姿勢辨識方法。為了達成這個目標，我們找到了北京大學智能機器人開發實驗室於 2022 年 11 月發布的「MotionBERT」[11]。其於 papers with code 網站的 Monocular 3D Human Pose Estimation on Human3.6M 和 One-Shot 3D Action Recognition on NTU RGB+D 120 分類兩項任務中都取得了最佳的成績[12]。

然而，為了使用 MotionBERT 進行轉換，我們還需要另外一項技術，即上海交通大學盧策吾團隊和騰訊優圖團隊於 2018 年研究的「AlphaPose」[13]。在這個完整轉換的流程中，我們需要先使用 AlphaPose 從影片中提取平面的人體姿態資料，然後再將這些資料傳入 MotionBERT 進行後續的處理[13], [14]。這種組合方法有望提供高效而準確的 3D 動畫生成能力，並符合我們的研究目標。

3.1 AlphaPose

AlphaPose 是一種採用「top-down」方式進行人體姿態推算的方法。在定界框的選擇方面，論文中選擇使用 SSD (Single Shot MultiBox Detector) 的方式進行處理。相對於其他方法(如 Yolo)，它選用 Regression 的方式，直接使用一個網絡找出定界框。SSD 是基於 VGG-16 的網絡架構，將最後的 Fully Connected layer 換成 Convolution layer，同時提供了定界框

位置的網絡。此外，SSD 也在不同深度的 Convolution layer 進行目標預測，以解決 Yolo 在小目標上不容易偵測的問題[13]。

當透過 SSD 獲得原始影像中人的位置後，AlphaPose 的架構包含了 Symmetric STN 與 SPPE 兩部分（即圖中的 STN+SPPE+SDTN 這三個區塊）。每一組 SSD 產生的定界框都會進入這個架構，用於進行人體關節與肢體的預測[13]。

值得注意的是，Symmetric STN（Spatial Transformer Network）主要用於對定界框中的特定區域進行補償和校正。SPPE（Single Person Pose Estimation）則是負責單人姿態的估算，該模塊專注於單一人物的姿態推斷。整個 AlphaPose 的設計結合了 SSD 的高效物體檢測和 Symmetric STN、SPPE 的準確單人姿態預測，以實現對影片中人體姿態的精確估算[13]。

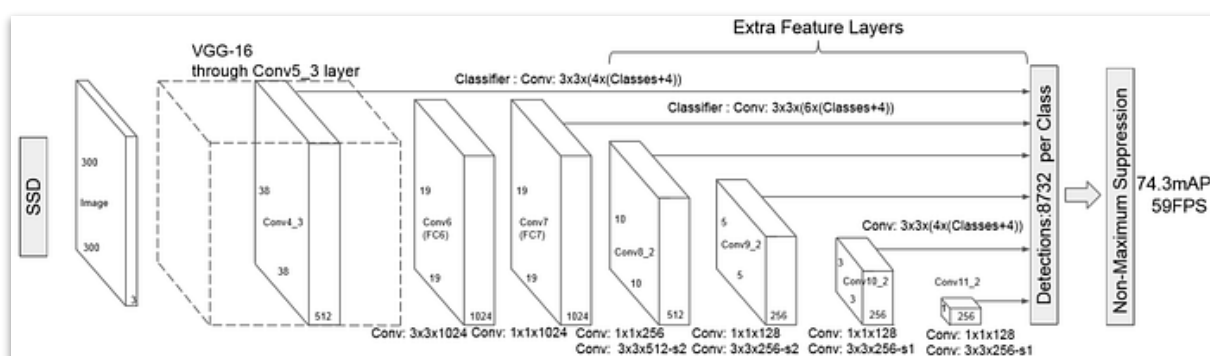


圖 3.1：AlphaPose 架構圖[13]

3.2 MotionBERT

MotionBERT 的方法分為兩個階段，即統一的預訓練和特定任務的微調。

在第一階段，進行統一的預訓練，研究使用一個動作編碼器來完成 2D 到 3D 的提升任務，並使用 DSTformer 作為骨幹。DSTformer 是一種序列對序列模型，專門用於處理動作

編碼的任務。此階段的目標是在不同的運動源中學習有用的資訊，以提升對 2D 到 3D 動作轉換的理解[11]。

在第二階段，對預先訓練的運動編碼器和下游任務（即微調的任務）的一些線性層進行微調。此時，使用 2D 骨架序列作為預先訓練和微調的輸入，因為這些序列能夠可靠地從各種運動源中提取資料。這一階段的目標是使模型更適應特定的應用場景，提高其對下游任務的性能[11]。

研究論文顯示，在不同的下游任務中，使用 2D 骨架序列是一種有效的方法。這表示在 MotionBERT 的架構中，2D 到 3D 的轉換是通過使用預先訓練的 DSTformer 和微調的線性層，以及特定任務的 2D 骨架序列，來估計 3D 運動的座標資料。整體而言，這種方法提供了一個多階段的學習框架，使得模型能夠更好地處理從 2D 到 3D 動作的轉換[11]。

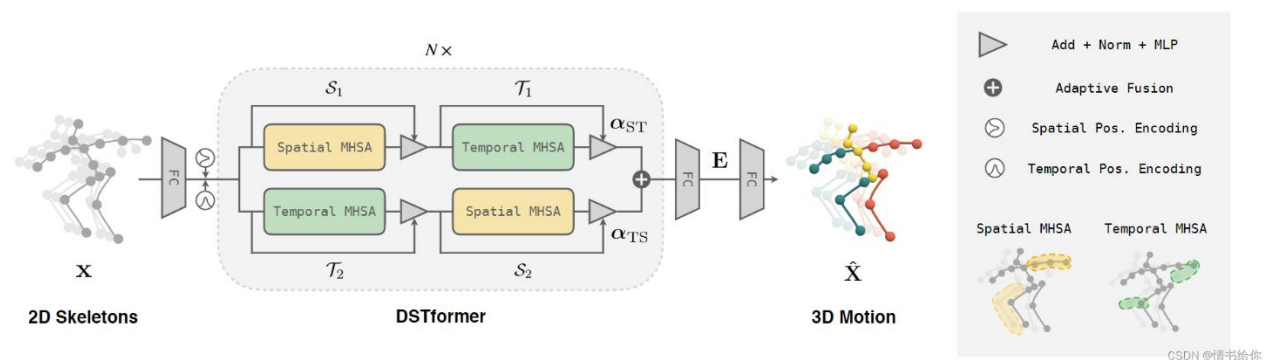


圖 3.2：MotionBERT 架構圖[11]

3.2.1 影片偵測轉換動畫的問題

在實際進行環境建置和安裝時，遇到了一個無法解決的主要問題，即系統只能辨識單一人。這個問題的根本原因可以追溯到影片經過 AlphaPose 處理後的檔案。在架設

AlphaPose 環境時，發現原論文所使用的分辨影片人物編號的函式庫需要的 Python 版本為 3.8 以下[14]，但其他主要系統所需的 Python 版本卻為 3.8 以上，導致版本衝突問題，無法成功安裝此功能。

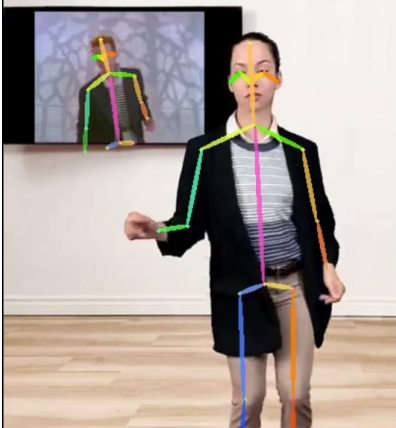
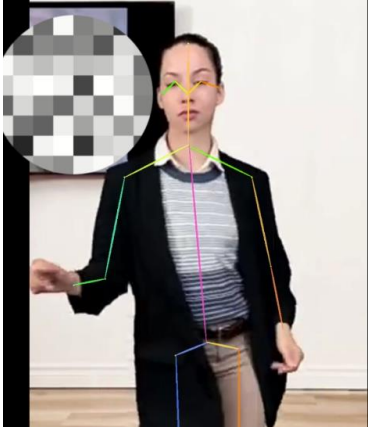
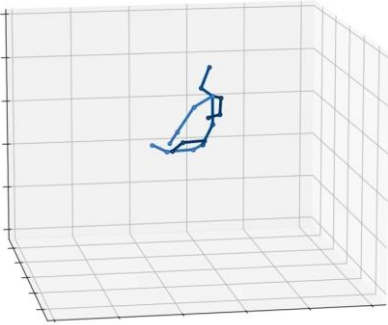
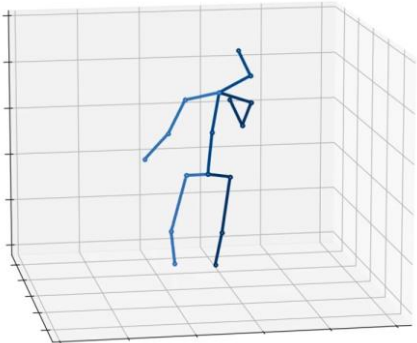
```
{
  "image_id": "0.jpg",
  "category_id": 1,
  "keypoints": [ 526.5582885742188, 788.0856323242188, 0.9641287326812744, 548.4618530273438, 766.1820678710938,
  "score": 3.1434760093688965,
  "box": [ 304.6441650390625, 670.9014282226562, 454.77996826171875, 1121.4642944335938 ],
  "idx": [ 0.0 ]
},
```

圖 3.3：AlphaPose 資料

在影片匯入 AlphaPose 並經過處理後，會輸出一份包含所有 frame 中人體辨識的 JSON 資料，如圖 3.3。其中，image_id 表示影片中第幾個 frame 所得出的資料，而 category_id 表示辨識後認定此為編號為幾號的人。在沒有分辨函式庫的情況下，所有人物的 category_id 都會是 1，也就是說所有人都被辨識為同一個人，並且可能會有多筆相同 image_id 和 category_id 的資料。這將導致在後續進行 MotionBERT 時出現不可預測的錯誤，因為模型會將多筆資料都辨識為同一個人，造成判斷錯誤，同時也浪費了處理時間。

表 3.1 中列舉了一份 6 秒（共 192 幀）的影片，並比較了是否對第二人進行遮擋處理的時間。兩部影片在 AlphaPose 處理動作的耗時相近，因此我們主要關注 MotionBERT 的處理效果。由表可見，在沒有阻擋背景螢幕中的人物的情況下，AlphaPose 會將兩人都進行標記，然後使用此資料進行 3D 處理時，會出現奇異的動作姿態，且有抽搐的狀況發生，其運算耗時也是單人影片的 3.8 倍。這表明存在對多人情境的辨識問題，需要進一步的處理和優化。

表 3.1：影片處理遮擋對比

| | 影片未處理 | 影片處理遮擋後 |
|-------------------------|--|---|
| 影片經 Alpha Pose 標記 |  |  |
| MotionBERT 3D 動作展示 |  |  |
| MotionBERT 影片花費時間 | 92 分鐘 | 24 分鐘 |

3.3 Mediapipe (BlazePose)

在即時動作偵測方面，我們選擇了 Google 開發的 Mediapipe[15]，主要使用了其全身姿勢追蹤功能，即 BlazePose[16], [17]。Mediapipe 以其高度準確的模型聞名，且由於其輕量化的特性，能夠在手機等較低性能的設備上運行，符合我們希望實現輕鬆遊玩的需求。

BlazePose 是一種專為人體姿態判斷而開發的輕量級卷積神經網路架構，特別針對在移動設備上的應用進行了優化。它能夠即時生成 33 個人體關鍵點，這些關鍵點包括身體的各個部位，如頭部、手臂、腿部等，可以提供對人體動作的詳細追蹤[16], [17]。

這個選擇具有的優勢包括高精度、輕量化、在手機上的實時運行，使其成為在實現即時動作偵測時的理想選擇[17]。使用 Mediapipe 的 BlazePose，使我們可以在不需要高昂硬體設備的情況下實現準確的動作捕捉，這對於我們的研究目標來說非常重要。

3.4 Open Sound Control (OSC)

在研究如何將手機的辨識資料傳輸到 MR 眼鏡的過程中，我們發現在大型 VR 線上遊戲「VR Chat」中，一些玩家使用自己編寫的程式來辨識動作並將資料傳送到眼鏡中[18]，而這種方法使用的是 Open Sound Control (OSC) [19]。這種協議的使用非常靈活，只需要確保發送端和接收端處於相同的網路中，然後設定發送端要將資料發送到哪個 IP、使用的 port 是什麼，以及接收端的 IP 是什麼[19]，就可以在接收端接收到相應的資訊。而我們將其設定為 Quest 3 眼鏡的 IP，即可於眼鏡端收到資料[20], [21]。

OSC 支援使用 TCP 或 UDP 進行資料傳送，但通常情況下建議使用 UDP。這個協議可以傳送任何數字型態的資料，因此具有極高的自由度[19]。由於 OSC 的特性，它不僅可以用於音訊傳輸，也可以應用於傳送其他數據，如動作辨識的資料。這使得 OSC 成為在不同裝置間靈活、快速地進行資料交換的一個理想選擇。

第四章、動畫資料處理

4.1 MotionBERT 檔案轉換

在進行 MotionBERT 檔案轉換的過程中，產生的 NPY 檔包含一個三維陣列，其中第一維度代表影片的 frame 數量，第二維度表示關節數量，按照 Human3.6M 格式排序，包括 'root', 'RHip', 'RKnee', 'RAnkle', 'LHip', 'LKnee', 'LAnkle', 'torso', 'neck', 'nose', 'head', 'LShoulder', 'LElbow', 'LWrist', 'RShoulder', 'RElbow', 'RWrist'，而第三維度包含 XYZ 座標資料，分別以 0 到 2 表示之[22], [23]。

為了在 Unity 中正確展示動畫，我們調整了資料的格式，將其設定為（15 個關鍵點，動畫 Frame 數量，XYZ），並將其儲存成 BYTES 檔。這 15 個關鍵點以身體中心 torso 為基準，向肢體延伸，節點皆儲存設特定父節點為原點之相對座標。

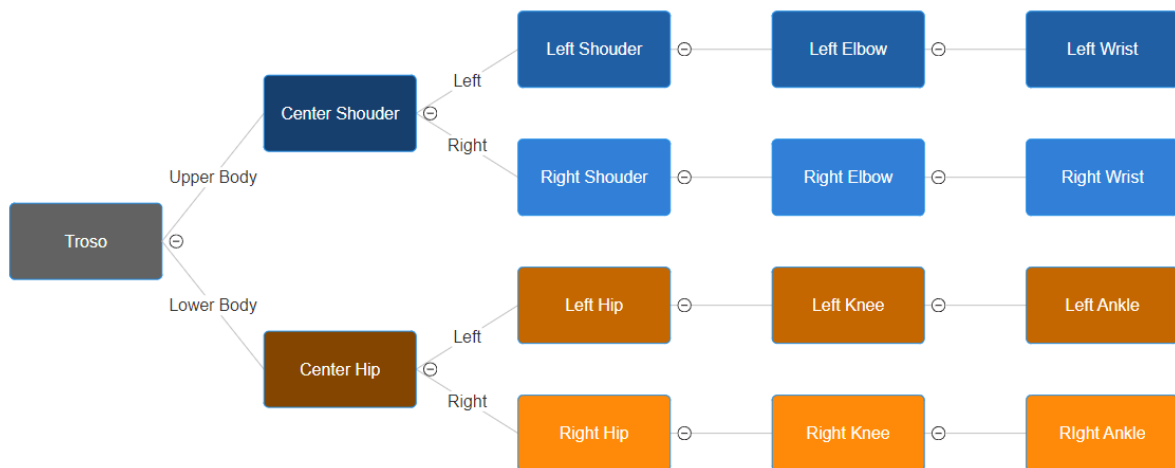


圖 4.1：關鍵座標點關係表

為了方便 Unity 讀取資料，我們對動畫資料進行正規化處理。計算每個關鍵點的相對座標，然後將這些相對座標進行正規化，使它們的距離固定為 1。這樣的處理有助於在 Unity 中展示動畫時降低運行時的計算成本。

在處理 NPY 檔時，我們注意到因為記錄的是肢體在影片中的絕對位置，所以需要特別處理身體中心的 Troso 座標，以避免出現部分動畫於運行時會飄浮在空中，而部分則會有插入地面的情況。這個處理確保了動畫在 Unity 中的呈現更加自然和合理。

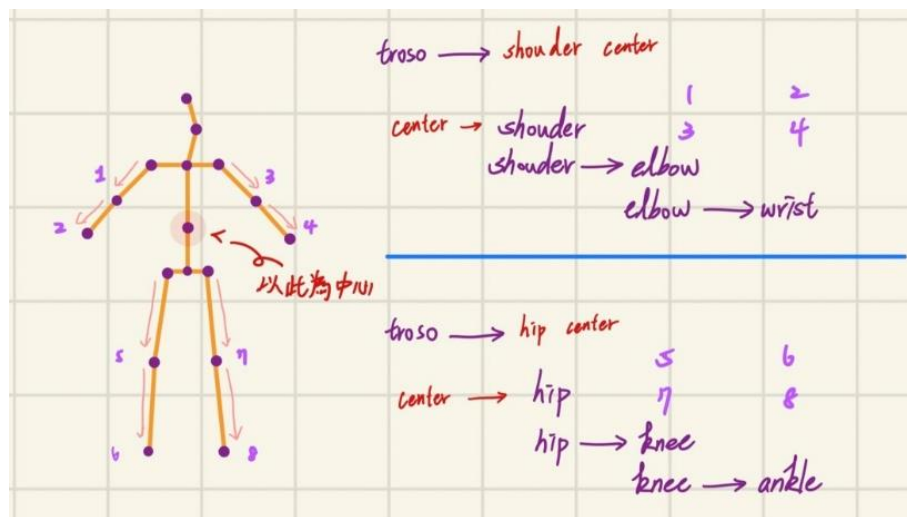


圖 4.2：偵測觸碰點位置

4.2 反向動力學(Inverse Kinematics)

在反向動力學（Inverse Kinematics）的概念中，這是一種以肢體末端的運動來影響根部的鏈狀運動方法[24]。這種技術需要設置運動的角度限制，並且通常用於確定使得末端位置正確的關節角度。在 Unity 中，由於已經有 Humanoid 的骨架，

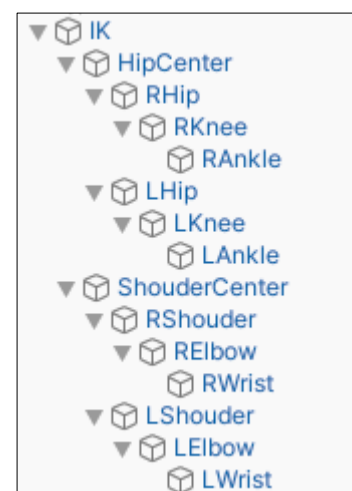


圖 4.3：IK 遊戲物件

因此處理以前部分製作的動畫資料時，使用反向動力學的難度降低了很多[25]，且可配合 Final IK，簡單拖拽物件資料進入指定位置運行[26], [27]。

為方便程式編寫及修改，我們把個別 IK 的處理編寫成一種 class，僅需在更換動畫檔案及更新 frame 動畫的時候進行呼叫，而不必對整個程式碼進行大規模修改。這不僅簡化了程式碼的管理，還有助於提高效能。

為了更方便地設置反向動力學的輔助點，所以照指定層次排列輔助點位，在以此控制動畫運行時可以只利用正規化的相對位置、肢體長度就找出輔助點應在的位置，並設定其為輔助點的 local position[26]。於圖 4.3 中可見 3D 模型會根據 IK 輔助點所在的位置進行移動，藉此展現出舞蹈動畫。



圖 4.4：IK 設置

4.3 曲譜結構

曲譜我們採用儲存 frame 數字的方法進行處理，此處理方式為自行構想，可以確保在動畫播放到特定 frame 時能夠同步放出特定時刻的動作偵測點。這對於確保動畫和音樂之間的同步性是非常重要的。

在資料處理時，會同步讀入舞蹈動畫後，利用影片音樂和舞蹈動作的特徵自動辨識可能是舞蹈特徵點的 frame。這個步驟的目的是為了辨識在動畫中哪些部分對應到音樂的節奏或特定的舞蹈動作。將這份資料儲存成 JSON 檔，方便主遊戲進行取用，這樣的作法確保了資料的方便存取和使用。

辨識標記方法為使用 aubio 取出所有音樂的節拍位置[28]，而將對於這些節拍所在 frame 位置取出，比對前後 5 個 frame 中 8 個肢體座標點的移動狀況及判定閾值決定是否為標記點，主要會被記錄的兩類動作為：移動到定點後進行角度大於 70 度的轉向動作、移動到定點後停下。

在一開始製作 Vive 眼鏡版本的遊戲時，我們是標記各個 frame 的 8 個偵測點中需要偵測那些點位，例如只需判定右手手腕，其他都可忽略。這個方式確實會比較清晰地向使用者展現需要注意的重點為何，但在點位數較多或較密集時就會變得很雜亂，無法一時之間就確定要用什麼部位碰觸哪裡，所以在



圖 4.5：Vive 版本運行畫面

後來轉換設備至 Quest 3 時選擇棄用此方案。

總體來說，使用這樣的資料處理流程及架構有助於確保動畫、音樂和舞蹈特徵的協同，更具趣味性，亦提高了整體遊戲體驗的品質。

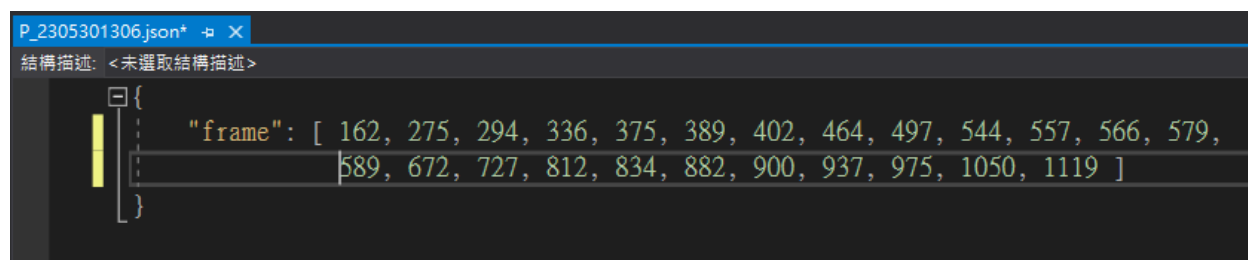


圖 4.6：曲譜結構

4.4 其他取用資料結構

由圖 4.7 展示所有轉換出的動畫資料類型，接下來會介紹所有資料夾內的檔案類型及檔案儲存資料說明。

`_AllDance` 資料夾儲存的僅為單一檔案，存有目前所有的舞蹈 ID，在選擇曲目時可以以此確認目前所有有的舞蹈曲目資料；`_BestData` 則存有每一首歌的最佳分數及最佳連擊數，因為儲存的為曲目的 ID，所以在曲目被刪除時也依然能夠存取到過往的最佳成績；`_ByteFile` 是舞蹈的.BYTES 資料，由此可以讀入舞蹈動作的資料，並在選單及遊玩主介面都進行動畫的展示，檔名命名方式為「B_id.bytes」；`_JsonFile` 則存有上一部分所提到的曲譜資料，可以在遊玩時提出使用，展示並處理要呈現及辨識動作的部分畫面，檔名命名方式為「P_id.json」；`_MusicFile` 存有每一首歌曲的音樂檔案，會結合舞蹈進行展示，檔名命名方式為「M_id.ogg」。

需要存取資料時，僅需使用 `UnityWebRequest` 程式[30]即可從資料庫取得及使用。

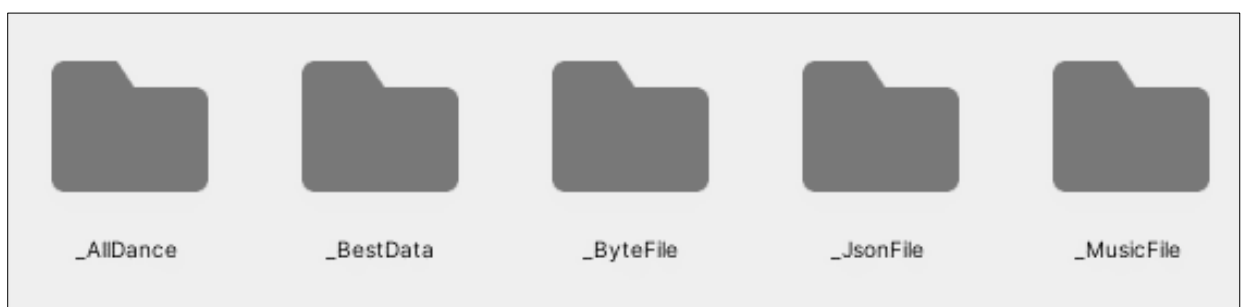


圖 4.7：儲存資料結構

第五章、動作辨識

我們使用 MediaPipe 的 Unity 官方插件[29]進行此部分程式的編寫。

在測試中，我們發現 MotionBERT 和 BlazePose 在 Z 軸（即垂直攝影機方向）的感知存在一些誤差，我們因此在動作辨識的階段選擇不使用 Z 軸的相關資訊。相反地，我們僅考慮在 XY 平面上計算動作的角度進行比對。這樣的處理方式是出於實用性和效能的考量，並在目前的系統中表現較佳。

在實務應用中，忽略 Z 軸的動作資訊有助於簡化演算法，降低計算複雜度，同時提高系統的穩定性。

5.1 手機 Mediapipe 辨識傳送

手機上的 Mediapipe 動作偵測軟體以每 0.1 秒的頻率計算 8 個舞蹈判斷點的角度，包括左右手腕、手肘、膝蓋、腳踝。我們將這些角度數據處理成 ± 180 度的範圍，減少傳輸數值的同時亦能降低傳輸錯誤機率。透過 OSC 協議[19-21]，這些資料被傳送至 MR 眼鏡，使用標籤為/mediapipe/的 address 進行識別。若未能辨識到人體姿態，則會連續傳送提示信息，頭戴顯示器持續接收並儲存資料，直至需要判定動作時才取出。

如圖 5.1 所示，當成功辨識到人體姿態時，手機端將持續偵測並傳送處理好的資料，每個 address 只包含一份 float 資料。這種傳送方式即使發生部分資料遺失，也不太可能導致嚴重錯誤。

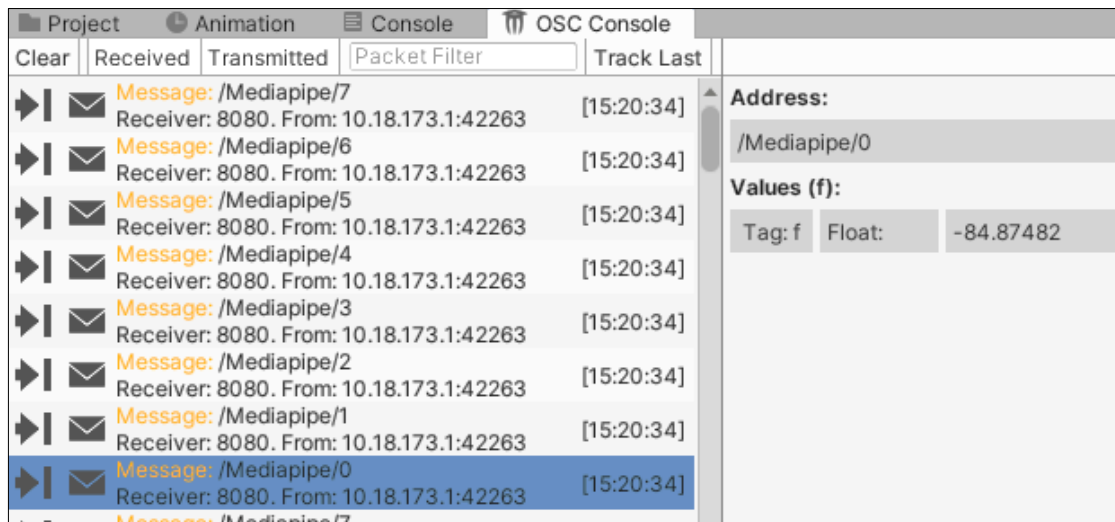


圖 5.1：OSC 資料正確接收

而在未能偵測到人體的情況下，將傳送 address 為"/Mediapipe/error"的提示信息，通知主遊戲程式目前缺乏人體資料。若在舞蹈動作進行中持續收到此訊息，則遊戲會主動提醒使用者存在問題並暫停遊戲。

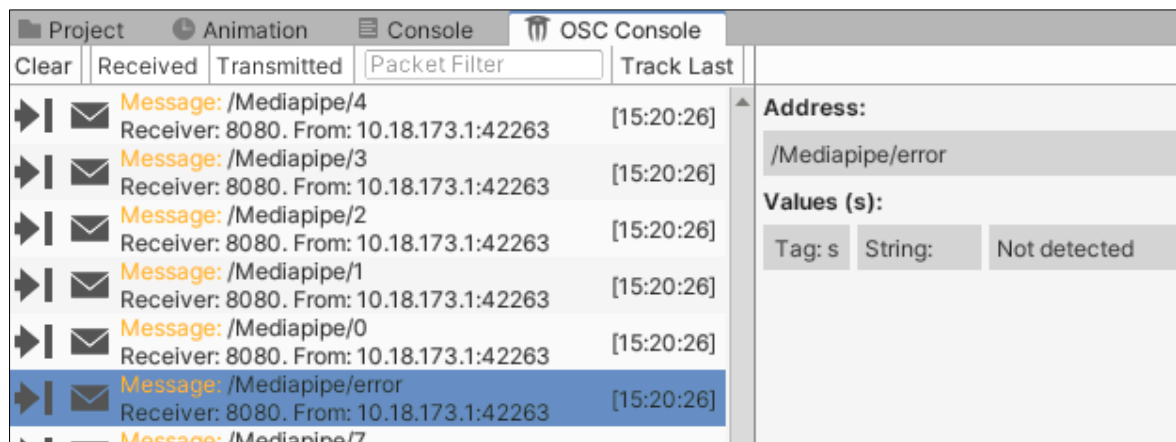


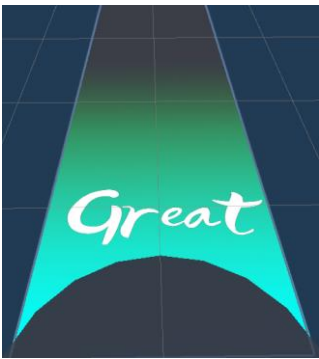


圖 5.2：動作辨識失敗

5.2 頭戴顯示器判斷動作準確度

頭戴顯示器會在接收到資料時同步進行儲存資料的更新，在碰觸點動畫移動到使用者站立位置、需要辨識時才進行判斷。

因可能還是有遊玩角度的誤差，故將辨識的寬容度調整至 ± 15 度角。為鼓勵盡可能準確的做出動作，當辨識時 6 個點位以上成功時分數加權最高，4 到 5 個點位成功時則進行一般分數加權，4 個點位以下成功時則判定為失敗不進行加分。

表 5.1：判定結果對照表

| 成功點位 | 6 個及以上 | 4 到 5 個 | 3 個及以下 |
|------|---|--|---|
| 類型 | Great | Nice | Fail |
| 地面畫面 |  |  |  |
| 分數加權 | 1.2 倍 | 1 倍 | 0 倍 |

除了單次點位成功數量的加權外，連續成功也會進行分數加權，不論是 great 或 nice 都算是成功完成，Combo 數都會增加，而每增加 5 個 combo 數時，加權就會增加 0.2。故若連續成功數越多，則總分更高，以此鼓勵使用者完整完成所有的舞蹈動作。

第六章、程式運行敘述

6.1 影片轉換動畫程式

是一款運行於 Windows 系統的轉換應用程式。其主要功能包括打包 AlphaPose 和 MotionBERT 環境以及轉換 UI 程式。使用者只需開啟轉換 UI，選擇欲轉換的舞蹈動畫影片，填入表格所需的詳細資料，然後點擊「開始進行轉換」按鈕，即可自動完成轉換動作的過程。

該程式將資料轉換後以指定格式儲存，指定格式參考 4.4 其他取用資料結構，並額外提取動畫和音訊檔進行曲譜轉換。處理完畢的資料會方便地存放在指定的伺服器上，以供使用者在頭戴顯示器和曲譜修改程式中讀取、修改和遊玩動作。

舞蹈python影片動畫轉換

Step1. 選擇影片

選擇舞蹈影片

選擇影片

Step2. 輸入資料

提示：請輸入英文

舞蹈名稱 輸入舞蹈名稱

歌曲作者 輸入歌曲作者

舞蹈作者 輸入舞蹈作者

Step3. 開始轉換

點我開始進行轉換

圖 6.1：影片轉換程式 UI

6.2 曲譜修改程式

為運行在 Windows 系統電腦上的應用程式，其主要功能在於讓使用者能夠查看並修改既有的曲譜。這使得使用者在尚未使用頭戴顯示器的情況下，即可實時檢視電腦運算的動畫狀態，並能夠迅速選擇是否保留目前的運算結果或進行必要的修改。

在打開程式時，系統會預設讀取所有曲目排序中的第一首。使用者可以輸入欲查看的曲目 ID，輕鬆修改讀取的曲目。接著，透過拉動上方的拉條及點選 frame 的數字按鈕，使用者可以方便地變更欲查看的動畫 frame。

在程式初讀入時，會一同讀入目前的曲譜，如果是有被標註的狀況，此 frame 的按鈕會變更顏色為紅色，如果需要修改是否被標註的狀態，就在選取當 frame 的狀態繼續點選 frame 的按鈕，則程式會進行記錄，在切換曲目後會主動掃描所有目前狀態並進行儲存。

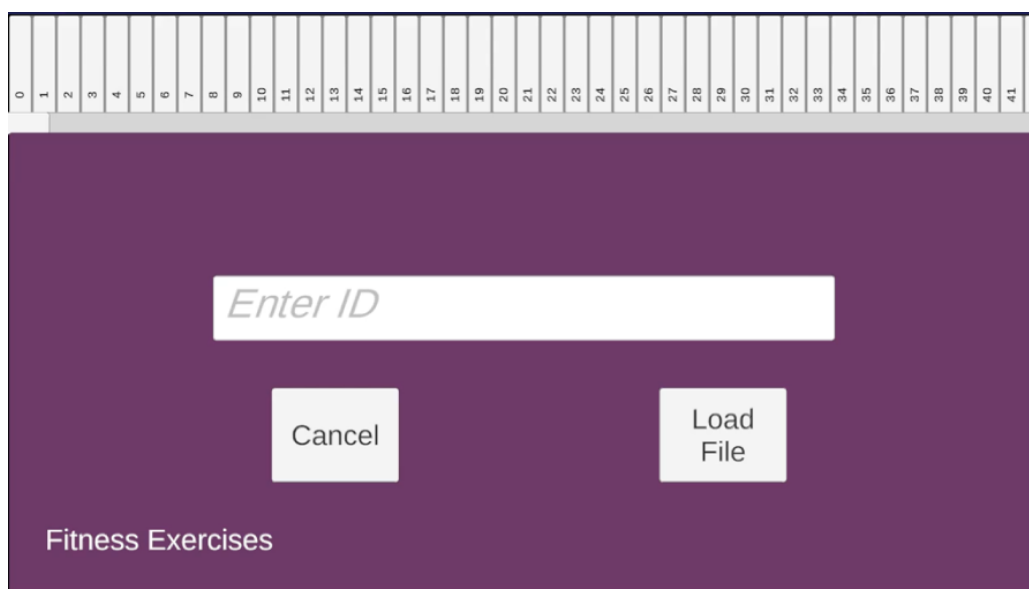


圖 6.2：曲譜讀入 UI

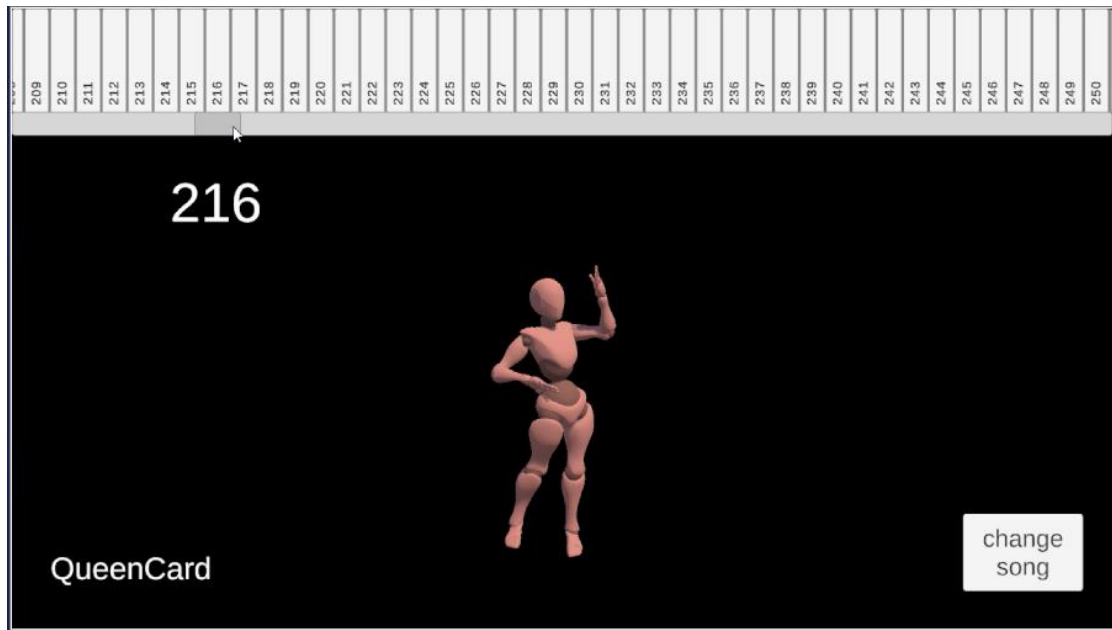


圖 6.3：曲譜查看介面

6.3 全身動作辨識程式

全身動作辨識程式是一款運行於 Android 系統的手機應用程式。建議使用相對較新的手機型號進行運行，經測試發現 3 年前的旗艦機型無法正常執行偵測功能。初始安裝時，程式會要求存取本機檔案的權限，使用者必須同意以確保 Mediapipe 的主要判斷系統正常運行。

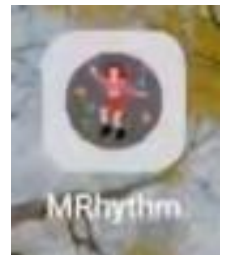


圖 6.4：辨識 APP

打開程式後，預設使用後方主攝影機，且鏡頭像素為 1280*720。儘管這對於一些最新旗艦手機的處理效能可能不會產生太大延遲，但對於一般普通手機可能較不推薦。此外，後置主攝影機在單一使用者進行操作時亦可能難以確認自身是否在拍攝範圍內。因此，建議在使用此程式時，將鏡頭調整為 camera 1，即前置自拍鏡頭，方便使用者確認狀態，不過具體相機編號與鏡頭的配對可能因手機而異，需要自行測試確認。畫素設定方面，可以

根據手機性能進行調整，建議使用較低畫素以減少延遲，儘管拍攝畫面可能較模糊，但動作辨識準確度在光線充足的情況下不會受到太大影響。

除了動作辨識功能外，辨識成功後會定時透過網路 OSC 協議將結果傳送至設定的 IP 和 Port。預設 IP 為我們研發時的電腦 IP，實際使用時建議確認頭戴式裝置的 IP 並輸入，或者設定為「255.255.255.255」的廣播頻道。需要注意的是，手機和頭戴式裝置必須使用相同的 Wi-Fi，否則資料傳送將無法正常進行。



圖 6.6：OSC 傳送設定



圖 6.5：影像輸入設定

在調整完設定後，將手機置於使用者前方進行拍攝。
若成功偵測到人體資料，系統將裁切出使用者以外的畫面，呈現為粉色（版本 1.0），以確保使用者即便戴著眼鏡也能清晰辨識是否成功辨識。

當辨識成功時，系統會在影片位置及右下角標示出偵測到的肢體關鍵點位置，左側標示為橘色，右側為藍色。使用者可以在遊玩前確認方向是否正確，若位置相反，可在影像設定中點選水平翻轉，以確保遊玩過程中取得正確的資料。這有助於提高辨識的準確性和遊戲體驗的一致性。



圖 6.7：APP 運行畫面

6.4 主遊戲程式

主遊戲程式運行於 Meta Oculus Quest 3 上，採用 Android 系統。在啟動前，需確認手機影像辨識程式是否正確設定及開啟，並放置在遊玩畫面前方，如所示。

使用主程式時，可完全依賴手部操作，無需攜帶額外的手把控制器，讓玩家在各種場合輕鬆遊玩。

6.4.1 舞蹈選擇介面

打開此介面後可以看到中央的曲目表及前方的按鈕，右前方會展示目前中間曲目的舞蹈，可以配合音樂及中央的詳細資料確認是否選擇此曲目，當要切換曲目時可以點選前方左右按鈕進行更換，點選後會即時變更目前曲目資料。

確定選擇此曲目後可以點選中央綠色的「選擇舞蹈」區域，點選後會於選單左方跳出過往的最佳成績及最佳連續擊打數，點擊「Play」按鈕後則正式進入遊戲遊玩介面。



圖 6.9：舞蹈選擇畫面

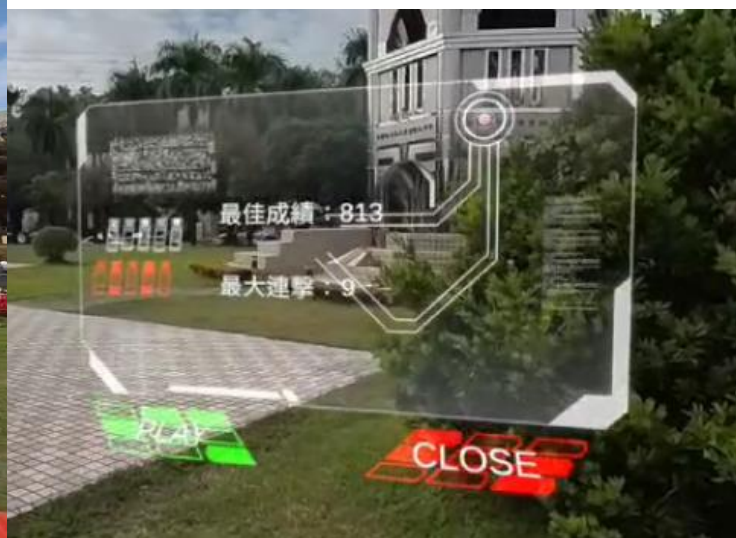


圖 6.8：最佳成績

6.4.2 舞蹈遊玩介面

進入遊玩介面後會先看到鎖的圖樣（見圖 6.10），此步需要將雙手平舉，擺出姿勢校正使用的 T 字動作，遊戲會進行動作資料的接收與偵測，若持續成功偵測到 T 字動作就會由紅轉為藍，當左右兩邊都轉為藍色後，鎖就會解開，並正式開始進行遊玩。



圖 6.10：校正動作畫面

開始遊玩後，畫面前方會出現傳送門，並由中出現需要擺出的靜態舞蹈動作，其會從的 from point 移動到 to point 位置，且會由小放大，讓使用者產生觸碰點由遠方向我們加速度移動的感覺。

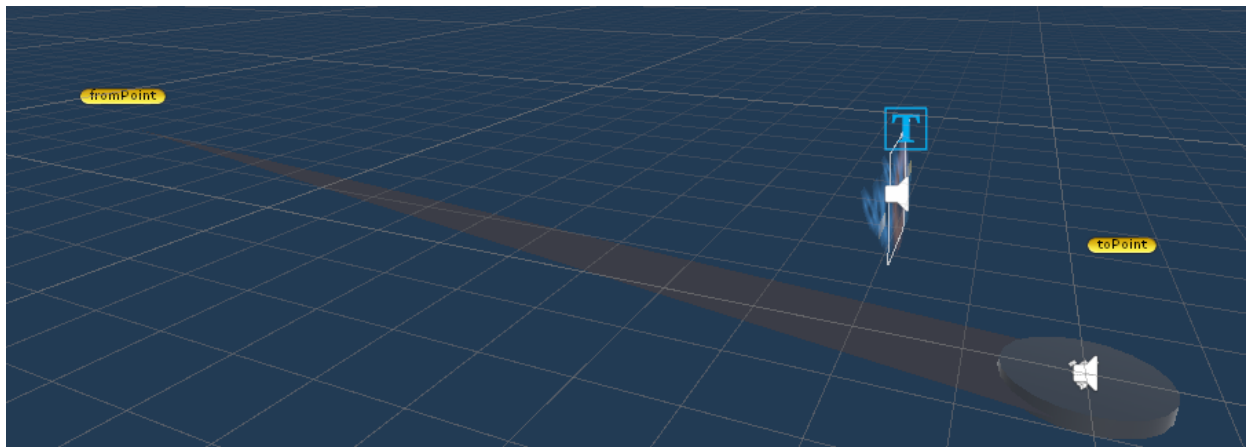


圖 6.11：動畫移動位置

當此靜態舞蹈動作移動到使用者的位置時就會進行動作準確度的判定，準確度越高成績越高，此成績計算方式請查看表 5.1：判定結果對照表的資料。

右前方跳舞模型會持續展示出舞蹈的動作，讓使用者清晰了解目前音樂段落的舞蹈為何，若能在過程中完全跟上則分數判定的成績必然不低。上方顯示目前分數，會隨著遊戲的進行持續計分。左方則為 combo 數，會紀錄遊戲開始至此連續成功的次數，清晰表示目前遊玩狀況。

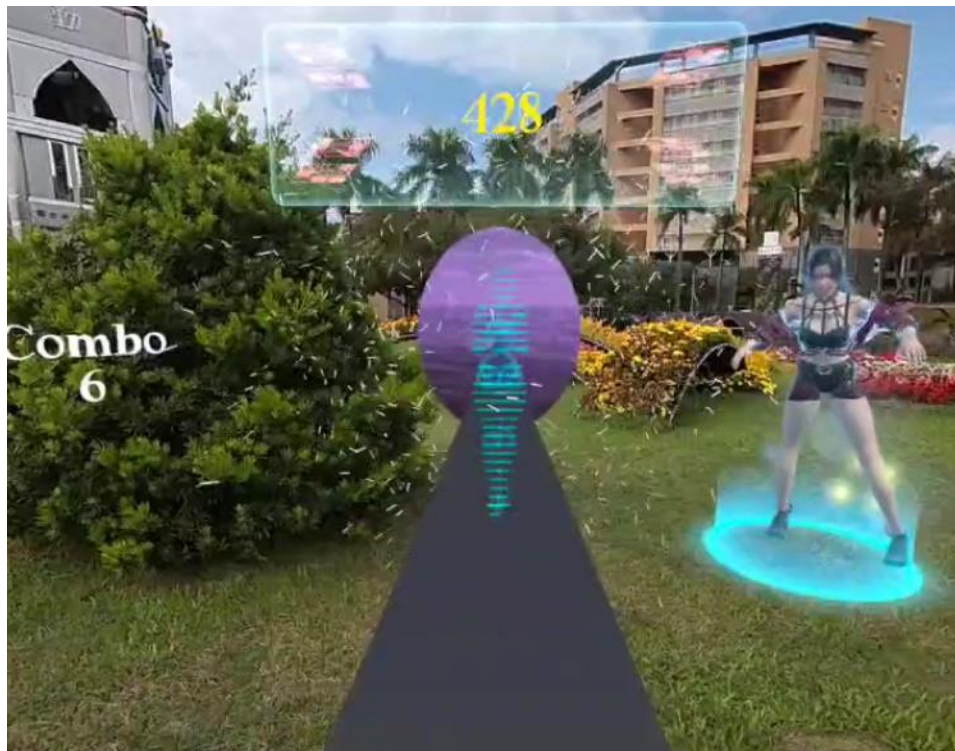


圖 6. 12：主遊戲遊玩畫面

在曲目遊玩結束後，畫面將顯示本次遊玩的分數、最高連擊數及本曲曲名，可見圖 6. 13：遊戲結束畫面，此讓使用者能夠自行比對過往紀錄，確認成績是否有提升及準確度更高。

若使用者確認完畢，且希望回到選曲介面，只需將其中一隻手打開並朝上舉起。此時，手部旁邊會出現一個跟隨手部位置移動的選單，詢問是否要回到選單介面。若使用另一隻手點選確認回到選單，則系統將停止目前的所有畫面並跳轉回選曲介面。這樣的操作方式提供了使用者友好的遊戲體驗，並使回到主選單的流程更加直觀和輕鬆。



圖 6.13：遊戲結束畫面

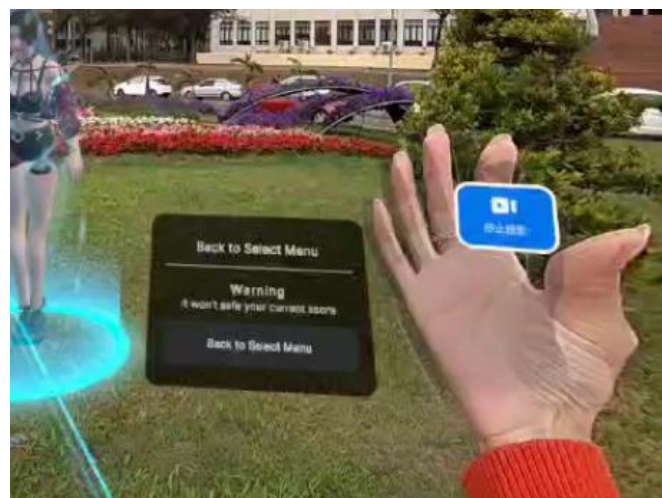


圖 6.14：手部選單

第七章、未來展望

目前系統算是完整可獨立運行，如果有後續希望商業化的想法，亦能夠將此做為雛形進行修改。可針對以下幾項進行處理：

7.1 主遊戲系統

目前只有兩個介面，分別為選單介面及遊玩介面。可以增加額外的系統及介面，讓遊玩體驗更完整，也讓此系統能被完整的善用。

7.1.1 等級系統及商城介面

系統部分可以增加等級系統及商城系統，在使用者每次遊玩都會根據遊玩難度及成績累積經驗值，每提升一個等級時，就可以隨機解鎖一個商城中的品項，以此激勵使用者持續遊玩。

為了讓遊玩體驗更加豐富，可在商城系統中添加讓使用者自由修改動畫模型及特效等遊戲過程中所見之物件的功能，讓使用者自定義喜歡的遊玩畫面及效果。

7.1.2 標籤分類

可以增加其他運動類型或將不同的舞種進行標籤分類，例如瑜珈、健身、嘻哈舞、宅舞、女團舞等等，並調整選歌介面讓使用者能根據個別想要練習的選項進行選擇，方便查找及選用目標曲目。

7.2 網路伺服器

目前是將資料庫建置在實驗室進行研究的電腦上，雖亦為使用網路存取，但如果電腦因特殊狀況無法使用就會沒有辦法新增曲目。所以如果在還是較少量的使用者時，可以考慮付費使用雲端伺服器的資料庫系統進行存取，這樣也方便各個不同位置的人都能夠載入遊玩到資料庫內的舞蹈。

亦能將轉換系統建置在伺服器上，並製作轉換的網頁 UI，讓使用者更方便的處理動畫檔案，也無須擔心是否使用者的個人電腦性能是否足夠運算其資料。

第八章、結論

此舞蹈學習系統，雖運行的程式繁多，共有四項。但就主要運行的設備及需求而言，有達成本研究一開始的訴求，即為「無須額外購入高昂的設備，可以輕鬆配戴遊玩」的目標，只需電腦、手機及 Quest 3 眼鏡即可使用遊玩。

因在過程中找到了蠻多前人所研究之動作處理資料，所以沒有過多的研究在影片轉動畫部分，而是選擇在程式的可擴展性上下手，雖較難在本報告書中展現出來，但在未來進行擴增或加強時會有較好的成果。

在進行研究前，確定應該要在硬體可支援想法時再動工，否則會像製作第一個 Vive 版本時一樣，花費了非常多的心力處理非其設計本意的功能，但結果還是無法達到想像的最低標準，和 Quest 3 輕鬆處理就能達到的效果差一大截。

於進行研究之時，Quest 3 與混合實境應用相關的 API 及官方文檔都尚未完善，且官方插件持續修改，很多功能都無法使用。故製作期間並未使用官方的插件，而是改使用 OpenXR 進行偵測及處理，所以一些偵測牆壁、地板位置及使用者設定的活動範圍之類的資料都無法取得，在應用上有點被侷限，可能需再過一陣子再根據官方文檔進行修改及添加功能。

參考文獻

- [1] Konami Digital Entertainment, “DanceEvolution ARCADE 遊戲について,” KONAMI, <https://p.eagate.573.jp/game/danceevolution/ac/p/howto/int.html>, 2012 (accessed Feb. 2023).
- [2] 廣州市番禺區動漫遊藝行業協會, “舞戰紀：原創新一代體感跳舞機簡介,” 廣州市番禺區動漫遊藝行業協會, <http://gzgaga.com/news/details?id=275>, Jan. 2023 (accessed Feb. 2023).
- [3] Ubisoft Entertainment, “Just Dance Now,” Just Dance Now, <https://justdancenow.com/>, 2023 (accessed Jan. 2024).
- [4] ppy, “welcome | osu!,” osu!, <https://osu.ppy.sh/home>, 2007 (accessed Jan. 2024).
- [5] HTC Corporation, “VIVE Pro 2 規格,” VIVE 台灣, <https://www.vive.com/tw/product/vive-pro2/specs/>, 2021 (accessed Feb. 2023).
- [6] HTC Corporation, “VIVE SRWorks SDK Guide,” SRWorks, <https://hub.vive.com/storage/srworks/>, 2017 (accessed Feb. 2023).
- [7] StereoLabs, “ZED 2i - Industrial AI Stereo Camera,” StereoLabs, <https://www.stereolabs.com/products/zed-2/>, 2020 (accessed Feb. 2023).
- [8] StereoLabs, “ZED Plugin for Unity,” Github, <https://github.com/stereolabs/zed-unity/releases/>, Sep. 2023 (accessed Feb. 2023).
- [9] Apple, “Apple Vision Pro — Apple 第一部空間運算設備,” <https://www.apple.com/tw/newsroom/2023/06/introducing-apple-vision-pro/>, Jun. 2023 (accessed Jun. 2023).
- [10] Meta, “Meta Quest 3：全新混合實境 VR 頭戴式裝置,” Meta, <https://www.meta.com/tw/quest/quest-3/>, Sep. 2023 (accessed Sep. 2023).
- [11] Wentao Zhu, Xiaoxuan Ma, Zhaoyang Liu, Libin Liu, Wayne Wu, Yizhou Wang, “MotionBERT: A Unified Perspective on Learning Human Motion Representations,” 2022, arXiv: 2210.06551

- [12] PapersWithCode, “MotionBERT: A Unified Perspective on Learning Human Motion Representations,” PapersWithCode, <https://paperswithcode.com/paper/motionbert-unified-pretraining-for-human>, 2023 (accessed May. 2023).
- [13] Hao-Shu Fang, Jiefeng Li, Hongyang Tang, Chao Xu, Haoyi Zhu, Yuliang Xiu, Yong-Lu Li, Cewu Lu, “AlphaPose: Whole-Body Regional Multi-Person Pose Estimation and Tracking in Real-Time ,” 2022, arXiv: 2211.03375
- [14] MVIG-SJTU, “AlphaPose-Real-Time and Accurate Full-Body Multi-Person Pose Estimation & Tracking System,” Github, <https://github.com/MVIG-SJTU/AlphaPose>, Jul. 2022 (accessed Feb. 2023).
- [15] Google, “MediaPipe,” Google for Developers, <https://developers.google.com/mediapipe>, 2023 (accessed Feb. 2023).
- [16] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, Matthias Grundmann, “BlazePose: On-device Real-time Body Pose tracking ,” 2020, arXiv: 2006.10204
- [17] Google, “Pose landmark detection guide | MediaPipe,” Google for Developers, https://developers.google.com/mediapipe/solutions/vision/pose_landmarker, Sep. 2023 (accessed Feb. 2023).
- [18] TheMysticle, FREE VR Full Body Tracking With Any Camera! (Jun. 1, 2023). Accessed:Oct. 2023. [Online Video]. Available: <http://www.youtube.com/watch?v=53JLgMyxi-4>
- [19] Wright, Matthew; Freed, Adrian, “Open SoundControl: A New Protocol for Communicating with Sound Synthesizers,” International Computer Music Conference (ICMC) vol. 1997, pp. 101-104, 1997.
- [20] Iam1337, “extOSC - Open Sound Control Protocol for Unity,” Github, <https://github.com/Iam1337/extOSC>, Dec. 2019 (accessed Sep. 2023).
- [21] dr_ext, “extOSC - Open Sound Control,” Untiy Forum, <https://forum.unity.com/threads/released-extosc-open-sound-control.436159/>, Apr. 2014 (accessed Sep. 2023).
- [22] Human3.6M, “Human3.6M Dataset,” Human3.6M Dataset, <http://vision.imar.ro/human3.6m/description.php>, 2014 (accessed Sep. 2023).

- [23] Walter0807, “MotionBERT: A Unified Perspective on Learning Human Motion Representations,” Github, <https://github.com/Walter0807/MotionBERT>, Mar. 2023 (accessed May. 2023)
- [24] M. Renaud, “A Simplified Inverse Kinematic Model Calculation Method For All 6r Type Manipulators,” Current Advances in Mechanical Design and Production VII, pp. 15-17, Feb. 2000.
- [25] Unity, “Unity-Manual: Inverse Kinematics,” Unity Documentation, <https://docs.unity3d.com/Manual/InverseKinematics.html>, 2023 (accessed Sep. 2023).
- [26] RootMotion, “Final IK | Animation Tools,” Unity Assets Store, [26]<https://assetstore.unity.com/packages/tools/animation/final-ik-14290>, Jan. 2014 (accessed Sep. 2023).
- [27] RootMotion, “Final IK Contains,” RootMotion, <http://www.root-motion.com/finalikdox/html/index.html>, 2014 (accessed Sep. 2023).
- [28] Meta, “Unity Audio: Unity | Oculus Developers,” Oculus Developers, <https://developer.oculus.com/documentation/unity/unity-audio/>, 2023 (accessed Oct. 2023).
- [29] homuler, “MediaPipe Unity Plugin,” Github, <https://github.com/homuler/MediaPipeUnityPlugin>, Oct. 2021 (accessed May. 2023)
- [30] Unity, “Unity-Scripting API:UnityWebRequest,” Unity Documentation, <https://docs.unity3d.com/ScriptReference/Networking.UnityWebRequest.html>, 2023 (accessed Sep. 2023).