國立台灣海洋大學資訊工程學系專題報告

題目

比奇堡天眼

作者

00757103	洪鈺凱	00757103@mail.ntou.edu.tw
00757129	劉永萱	j533688888@gmail.com
00757140	黃湘庭	tin890812@gmail.com
00857206	張祐琪	00857206@email.ntou.edu.tw

指導教授：蔡宇軒 博士

中華民國 110 年 12 月 09 日

Project Report, NTOU CSE

Title

Sky-eye of Bikini Bottom

Author

00757103	Yu-Kai Hong	00757103@mail.ntou.edu.tw
00757129	Yong-Syuan Liou	j533688888@gmail.com
00757140	Hsiang-Ting Huang	tin890812@gmail.com
00857206	You-Chi Chang	00857206@email.ntou.edu.tw

Advisor：Dr. Yu-Shiuan Tsai

2021 / 12 / 09

目錄

摘要

近年來資訊科技進步，深度學習與電腦視覺應用在生活隨處可見，結合 3D 技術也有非常大的發展性，因現在二維的影像效果已經無法再滿足人類需求。因此，我們選擇以三維重建為本次專題的重點，並以實現立體魚輪廓為目的，透過機器學習來重建 3D 魚影像。我們利用了 DeepLabCut 深度學習訓練工具，以機器學習訓練網路模型的方式，來追蹤魚部位的位置，找出其特徵點相對應之座標，並映射出來使其立體化，建構魚隻立體模型。

然而，因為會出現部位未辨識及偏移的問題，我們也利用 Kalman 濾波器優化結果。對於較為無法辨識的背面眼睛，利用魚眼估計計算並加入立體模型中呈現。未來透過將訓練資料多元化，可以解決目前魚缸環境限制的問題，並增進魚隻多樣性。也可以透過增加訓練迭代次數，並增加辨識的準確率，使魚隻輪廓更加立體與生動。通過這次研究我們了解到 3D 技術中的基礎，並實作機器學習流程，未來若有機會再更進一步鑽研，結合更多不同的技術，希望能展示更不同的魚隻樣貌。

關鍵字：立體視覺、魚骨架偵測、DeepLabCut、Kalman 濾波器、深度學習

一、 簡介

（一） 研究動機

傳統的影像辨識是以二維的彩色影像辨識系統為主，但是缺乏深度資訊，無法辨識更為完整的物體資訊。因此近年來三維視覺計算是個崛起中的領域，亦是產業界高度看好的範疇，例如 Xiang 等人利用雙目立體視覺識別串收番茄[1]。身為海洋大學的一員，我們對建立三維的魚隻影像很感興趣，希望能透過電腦監控魚隻的行動，以降低養殖照顧的工作量，但是三維的座標通常很難在短時間內測量數值，尤以會動的物體為甚，而那些為數不多的工具通常只能專門針對特定動物，還可能受到其他方面因素的限制。

在過去的幾年中，深度學習的發展為研究動物行為帶來了新的工具，世界各地的科學家們都針對動物的行為開發了許多不同的計算模型。例如由 Cao 等人所做即時多人二維姿態估計 OpenPose[2]，使用深度學習偵測人體、手、面部和足部關鍵點。或是由美國哈佛大學的 Mathis、Bethge 等人開發的一款開源運動追蹤工具 DeepLabCut[3]，可在訓練數據相對較少的情況下追蹤其他動物獲得出色的結果，並且增加了實時性(DeepLabCut-live)和多動物(DeepLabCut_maDLC)支持。

除了二維影像辨識之外，在三維重建的方面，可以藉由相機校準、影像校正、三角測量等方法完成從二維至三維的轉換，例如，Zhang 在 2000 年改進了相機校準的演算法[4]；Aslam 和 Ansari 運用了三角測量的方式進行了雙鏡頭成像的研究[5]；Zou 與 Li 提出了以 OpenCV 實作 Bouguet 演算法的雙目立體視覺技術[6]；Kar 等人也提出了透過機器學習的三維重建方法[7]。而 DeepLabCut 結合上述所述之雙目立體視覺技術，從成對的相片中計算物體的深度資訊，以完成從二維到三維的轉換，使三維視覺計算從實驗室環境到現實世界的使用又向前邁進了一步。本專題選擇採用 DeepLabCut 來進行從雙鏡頭影片建立三維立體魚隻的研究。

（二） 研究目標

實驗過程中，我們先是對二維影像進行標點，將已標點資料集合並訓練，以初步得出二維影像辨識之模型，再藉由接續訓練作業重複訓練以獲得更加精準之二維辨識模型。爾後拍攝棋盤方格之雙鏡頭相片(checkerboard pictures)，經雙目視覺之立體校正，通過三角測量進行重投影，建構出三維空間，得出該雙鏡頭在固定相對角度與距離之狀況下，由二維座標轉換至三維座標的通式。之後再輸入該雙鏡頭所拍攝之魚影像，利用二維辨識模型辨識出二維座標，再透過通式轉換至三維座標。最終，依據此三維座標做出每一個時刻的三維圖像，再將圖像製作成影片。

由於人員與時間有限，本專題所獲取之標點資料與訓練資料也相對有限，因此在辨識上的精確度未到達預設理想值。為改善此情況，我們利用 Kalman 濾波器(Kalman filter)來修正三維座標的狀態預測值[8]，以獲得更加穩定精確之三維座標資料，並利用魚眼估計找出未辨識到之魚眼，使魚隻輪廓更為完整。

（三） 設備應用

1. 硬體設備
   - Intel Core i5-9600KF 處理器(圖 1)
   - RAM 64G
   - NVIDIA GeForce RTX 2080 顯示卡(圖 2)

圖 1、
9th Generation
Intel® Core™ i5 Processor

圖 2、
NVIDIA GeForce RTX 2080 Graphics Card

2. 軟體設備
   - Ubuntu 20.04(圖 3)
   - Anaconda 2021.05(圖 4)
   - Tensorflow 2.6(圖 5)
   - DeepLabCut 2.2(圖 6)

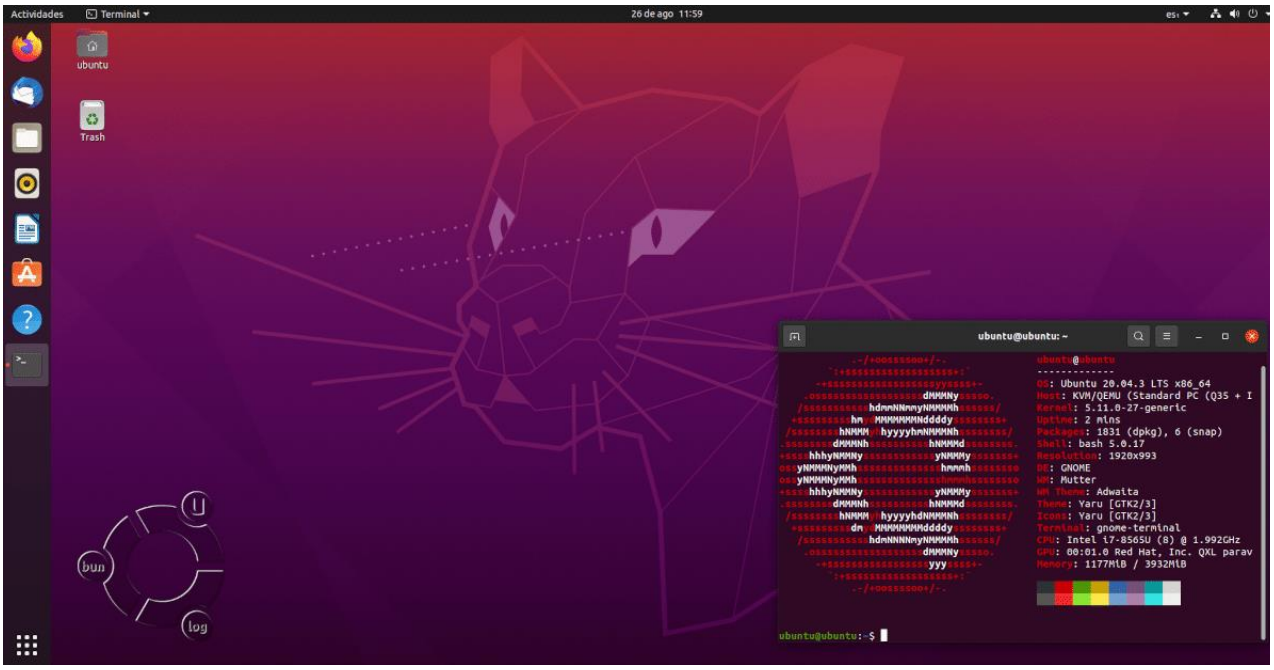

圖 3、Ubuntu 20.04 desktop

圖 4、Anaconda

圖 5、TensorFlow

圖 6、DeepLabCut

二、 理論推導

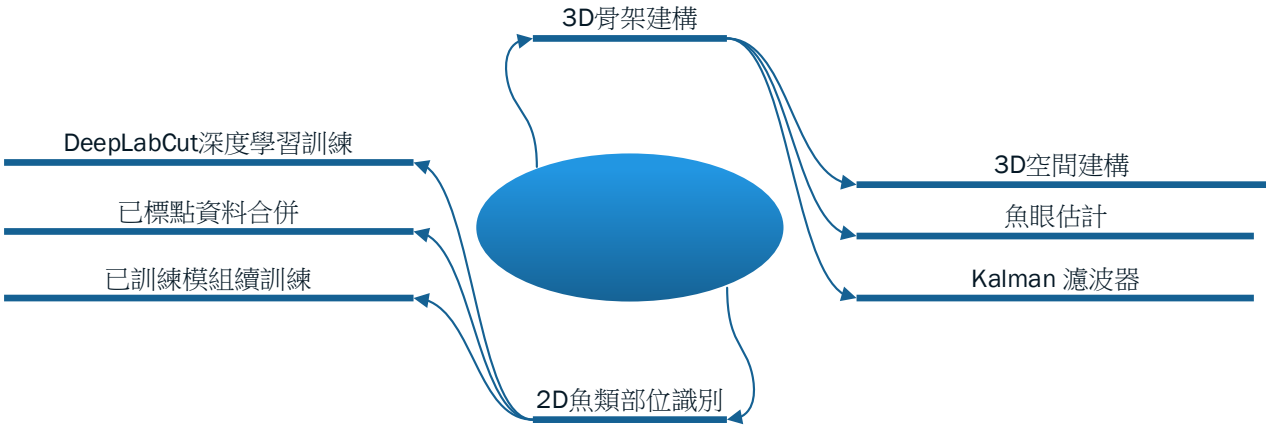

圖 7、專案細部分割圖

本專題主要可分兩大部分，一是做出可以識別魚部位座標的深度學習模型，二是以雙鏡頭所攝影出的影片，藉由模型辨識出二維座標後轉換為三維座標，由此建構出三維之骨架。每個部份又依據本專題遇到的需求，分出六個小部分依序說明。(圖 7)

（一） DeepLabCut 深度學習訓練

DeepLabCut 是由美國哈佛大學的 Mathis、Bethge 及同事利用機器學習開發的一款開源運動追蹤工具。[3]

DeepLabCut 利用了 Insafutdinov 等人提出的一種最先進的人體姿勢估計算法的特徵檢測器（ResNets + 讀出層），稱為 DeeperCut[9]，啟發了工具箱的名稱，並添加了速度更快、性能更高的變體，包括 MobileNetV2s、EfficientNets[10] 和自己的 DLCRNet 主幹，提高了推理速度，並提供了額外的新的增強方法，增加了實時性和多動物支持。為動物姿態估計提供最先進的性能。

（二） 已標點資料合併作業

因應疫情與設備造成的遠端協作模式，並考量標記資料的效率，隔一段時間就會將所有組員的二維標點資料合併成為一份資料集，完成後接續後續作業。

在 Labeling Project 中，標點完成的資料儲存在 project folder/labeled-data/video_name 資料夾中，以.h5 檔案形式儲存，並有易讀性高的.csv 檔案做為副本。又 Project 是以 config.yaml 中 video_sets 參數定位出 video_name 資料夾，因此只要在 Training Project 內，調整 video_sets 的路徑，並確保 h5 檔案中路徑正確，即使標點資料增加了可攜性，同時也可依此特性合併 Labeling Project。(參照圖 8、圖 9)

（三） 已訓練模型續訓練作業

考量到視訊記憶體等硬體的限制，以 1080p 的影片而言，每次訓練的極限約在 3000 張標點資料左右。為了突破該限制，須沿用前一次訓練的結果到下一次的訓練當中，以達到接續訓練的效用。

在 Training Project 中，訓練出的模型儲存在 project folder/dlc-models 中的 train 資料夾，訓練相關的參數則儲存於該資料夾的 pose_cfg.yaml 內，其中 init_weights 參數決定訓練開始時的初始模型，由於模型內儲存的是各部位的特徵值，因此替換 init_weights 路徑即可以以不同的初始特徵值開始訓練，將其值改成前次訓練完後的模型即可延續前次訓練內容繼續訓練。(參照圖 8、圖 9)

（四） 3D 空間建構

固定兩攝影機間距，以多個不同角度與距離拍攝棋盤方格，對這些相片進行校準：計算每個相機的內在和外在參數，使用這些參數計算重投影誤差，估計兩個相機之間的轉換，並對相機進行立體校準。通過計算立體校正將兩相機圖像(棋盤方格)平面帶到同一平面。

使用相機矩陣對校準圖像和角點進行去畸變(undistortion)，並將這些去畸變的點投影到去畸變的圖像上以檢查它們是否正確對齊，以檢查立體校正的效果。

如果沒有錯誤，便可以對來自兩個相機的姿勢進行三角測量以獲得三維坐標。

（五） 魚眼估計

由於青萬隆體型結構上並不圓潤，若非正面面對鏡頭，通常取得的影像僅能包含一隻眼睛，而魚隻的行為難以控制，僅有少數片段能夠同時偵測到雙眼，此演算法便是為了改善此現象以利於實現 3D 化。

已知 a,b,c 點位於同一平面 A 上，可得該平面 A 之點斜式及法向量。又一不在平面 A 上之點 d，可得 d 對 A 之對稱點。基於上述概念，加上由於青萬隆脊椎的結構限制了擺動方向，使其水平方向擺動大於垂直方向，取魚嘴、魚腹鰭前緣、魚背鰭前緣分別可視為 a,b,c 點，由該三點可大致估量出魚頭的縱切面，並透過該平面可大致推導出另一側魚眼的位置。

（六） Kalman 濾波器

Kalman 濾波器是一種自回歸濾波器，可以從一連續的測量中，平滑化該測量中的雜訊，其特色之一，是可以估計對過去、現在及未來的測量值。

因訓練部分可能有空值，或可能有偏離理想值太多的個別點，因此用 Kalman 濾波器將極端雜訊平滑化，同時估計未測量/計算到的空值。我們使用 Kalman 濾波器的過程中，曾經參考了 Welch 和 Bishop 在 1995 年對 Kalman 濾波器進行的一系列理論上的推導以及應用實例[8]。

對某個時段 k 的估計值（k>=2），可以依時段 k-1 的濾波結果經線性運算得出，然後可依設定的比例，由時段 k 的估計值和時段 k 的測量值權衡，得出時段 k 的濾波結果。並以該結果進入下一循環。若測量值為空（未測得），則取估計值為濾波結果。

三、 架構與演算法則

（一） 專案流程

使用 DLC 深度學習工具，先由研究人員創建標點用專案(labeling project)對二維影像分別進行標點，再利用已標點資料合併作業建立訓練用專案(training project)，將多位人員建立之資料集合併並訓練，初步得出可用於辨識之模型(network model)，後續可用於二維圖像辨識，並可藉由對訓練資料集之比較得出誤差值。若該模型無法一次訓練至研究人員所需之精準度，可藉由已訓練模型續訓練作業重複訓練成更加精準的二維辨識模型，以進行三維建構作業。

待二維辨識模型的標點結果符合研究人員期待，建立三維專案(3D project)，利用事先攝影之棋盤格雙鏡頭照片(checkerboard pictures)，藉由 3D 空間建構，經雙目視覺之立體校正，透過三角測量進行重投影，建構出三維空間，得出該雙鏡頭在固定相對角度與距離之狀況下，由二維座標至三維座標的轉換方法。

將該雙鏡頭所拍攝之魚影片(camera videos)，利用二維辨識模型辨識出二維座標，再透過上述通式轉換為三維座標。可利用 Kalman 濾波器修正狀態預測值，確保能得到穩定的三為座標資料，並利用魚眼估計找出未辨識到之魚眼，使魚隻輪廓更加完整且具立體感。最後再依據此三維座標做出每一個時刻的三維圖像，再將圖像製作成影片。(圖 8)

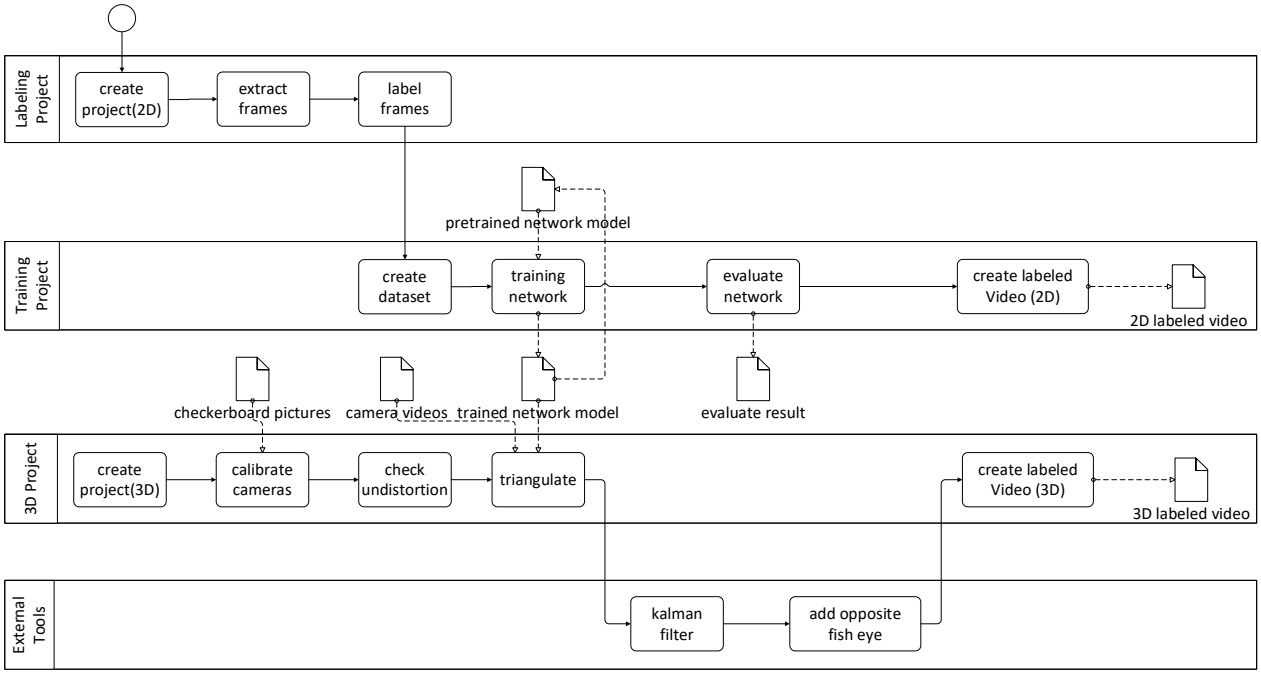

圖 8、專案流程圖

（二） 操作方法

本專題大多數之執行程序皆為 python 函式，參考下述模組描述之函式設計，並依據專案流程圖(圖 8)執行即可。

維須注意之步驟為前述之**已標點資料合併作業**(表 1)與**已訓練模型續訓練作業**(表 2)。

表 1、已標點資料合併作業

複製 Labeling Project 的 labeled-data 資料夾與 videos 資料夾 至 Training Project 的專案目錄。	(步驟 1)
將 Labeling Project 中，config.yaml 的 video_sets 部分 複製至 Training Project 的 config.yaml。	(步驟 2)
使用 check_labels(config_path, visualizeindividuals)確認是否成功。	(步驟 3)
執行 Training Project 的 create_training_dataset(config_path) 建立訓練資料集。	(步驟 4)

例外狀況	有重名影片	重新命名影片，更改 labeled-data 資料夾中 csv 檔案內之路徑，使用 convertcsv2h5(config,scorer)重建 h5 檔案。
	scorer 不一致	使用 convertcsv2h5(config,scorer)重建 h5 檔案。

表 2、已訓練模型續訓練作業

從已訓練專案的 project folder/dlc-models/…/train/路徑找到以下檔案： snapshot-XXX.data ???-of-??? snapshot-XXX.index snapshot-XXX.meta XXX 是最後一次 model 快照的 iteration，???為不特定數字。	(步驟 1)
從待訓練專案的 project folder/dlc-models/…/train/路徑找到 pose_cfg.yaml	(步驟 2)
以步驟(1)檔案的絕對路徑， 刪去副檔名後替換待訓練專案中 pose_cfg.yaml 的 init_weight 值。	(步驟 3)
將待訓練專案重新執行 train_network(config_path, displayiters, saveiters, maxiters) 訓練。	(步驟 4)

四、 模組設計描述

（一） 檔案架構

本專題之主要架構可分為 2D Project(圖 9)和 3D Project(圖 10)，專案流程圖(圖 8)中之 Labeling Project 和 Training Project 皆為 2D Project 之架構。該架構中只會顯示研究人員依據

本專題架構與流程操作會使用到的檔案。

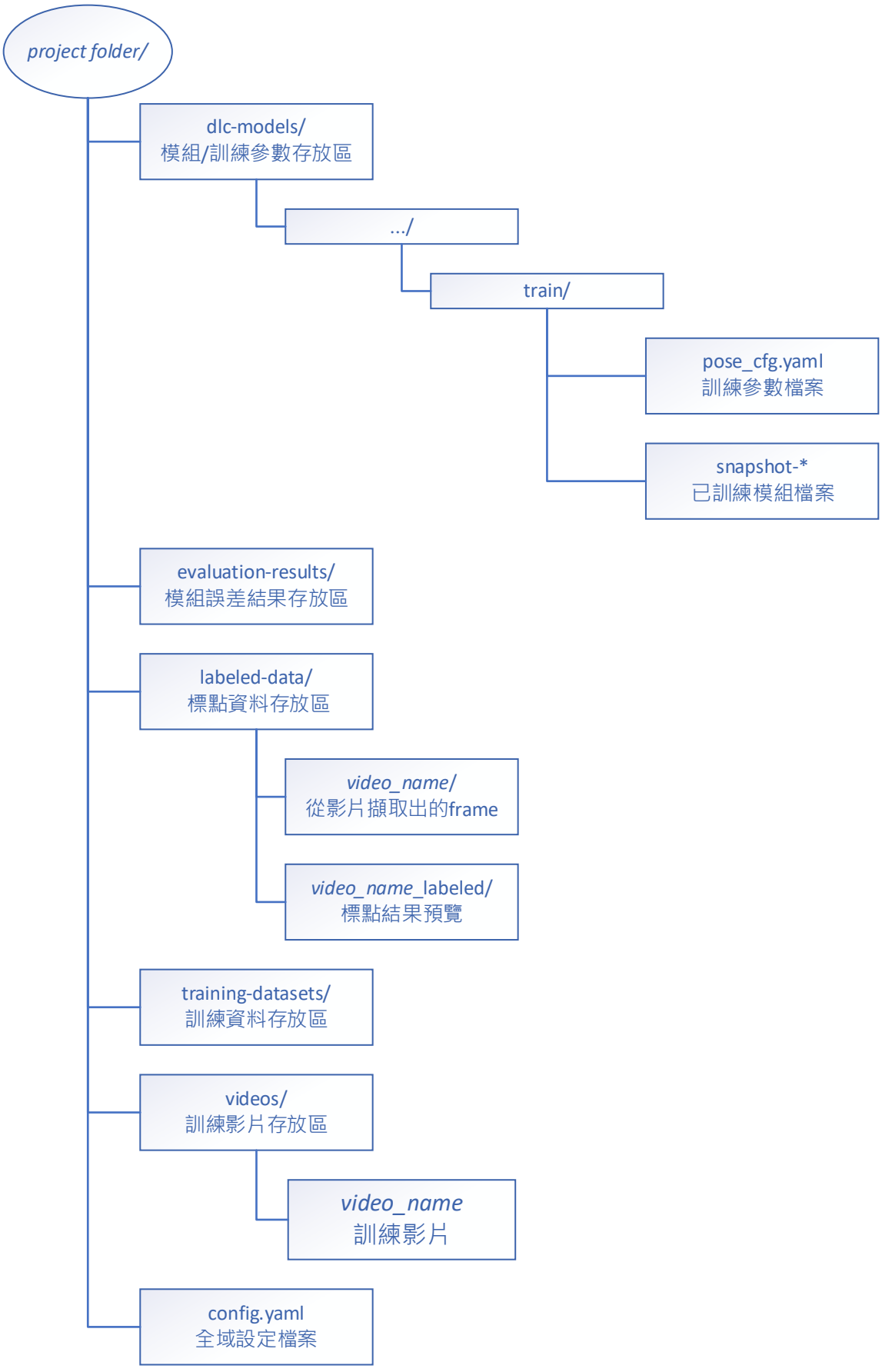

圖 9、2D Project 專案架構圖

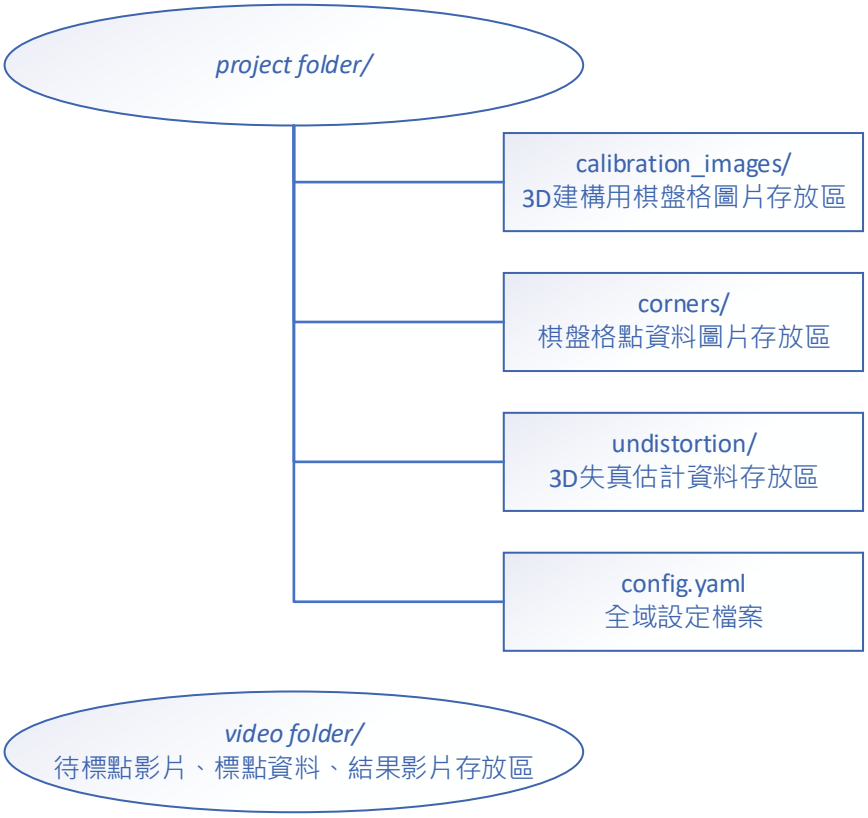

圖 10、3D Project 專案架構圖

（二） 檔案參數

本專題之各項參數可分為二類，一為置放於檔案內之固定參數，二為呼叫函式時可由需求調整之可動參數(見下節(三))。固定參數只存在於下列三檔案之中，可參照專案架構圖(圖9、10)之位置改動。本節僅顯示研究人員依據本專題架構與流程操作會使用到的參數。(表 3~5)

表 3、config.yaml(2D Project)

project path	專案路徑
video_sets	影片路徑及切割資料 (影片路徑包含 video name/影片名稱)
bodyparts	座標點名稱
numframes2pick	切割幀數
skeleton	骨架
default_net_type	預設網路模型

表 4、pose_cfg.yaml

batch_size	tensorflow 資料集分割大小 (大幅度影響 training 速度，但依硬體、影片大小、資料集大小影響最大值)

dataset	訓練/測試資料集路徑
init_weights	初始訓練權重路徑
net_type	網路模型種類

表 5、config.yaml(3D Project)

Project path	專案路徑
skeleton	自定義骨架顯示資料
num_cameras	攝影機數目
camera_names	自定義攝影機名稱
scorername_3d	3D 專案負責人
trainingsetindex_*camera_names*	訓練資料集編號 camera_names 部分需與上述 camera_names 同步
config_file_*camera_names*	訓練用專案(Training Project)全域設定檔案路徑 camera_names 部分需與上述 camera_names 同步

（三） 函式設計

各模組即為可執行之函式，可對照架構與演算法則中專案流程圖(圖 8)及操作方法執行，函數定義依原始碼列出，但參數說明只顯示研究人員依據本專題架構與流程操作會使用到的參數。

模組1. 創建一個新的 2D 專案，自動創造專案架構及全域設定檔案

create_new_project(project,experimenter,videos,working_directory=None,copy_videos=False,videotype=".avi",multianimal=False)

參數說明：

參數名稱	型態	說明
project	string	專案名稱
experimenter	string	專案負責人
videos	list	標點影片的路徑集合
videotype	string	標點影片的副檔名
working_directory	string	專案路徑(可選)
copy_videos	bool	複製影片 or 創建捷徑(可選)

模組2. 用設定檔案的參數切割標點用的 frames

extract_frames(config_path)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑

模組3. 開啟標點用的 GUI

label_frames(config_path)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑

模組4. 確認標點資料的正確性、預覽已標的點

check_labels(config_path, visualizeindividuals=True)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑
visualizeindividuals	bool	預覽標點

模組5. 以 csv 檔案重新建立 h5 檔案

convertcsv2h5(config,scorer)

參數說明：

參數名稱	型態	說明
config	string	全域設定檔案路徑
scorer	string	新專案負責人

模組6. 用已標的點資料作出訓練資料集與測試資料集

create_training_dataset(config_path)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑

模組7. 將資料集以機器學習之方式訓練出模型

train_network(config_path, displayiters=100, saveiters=15000, maxiters=30000)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑
displayiters	integer	Terminal 顯示的訓練週期
saveiters	integer	snapshot 存檔的訓練週期
maxiters	integer	最大訓練週期值

模組8. 估量現有模型的成效(誤差值)

evaluate_network(config_path, plotting=True)

參數說明：

參數名稱	型態	說明
config_path	string	全域設定檔案路徑
plotting	bool	是否以圖像顯示誤差

模組9. 創建一個新的 3D 專案，自動創造專案架構及全域設定檔案

create_new_project_3d(project, experimenter, num_cameras=2, working_directory=None)

參數說明：

參數名稱	型態	說明
project	string	全域設定檔案路徑
experimenter	string	專案負責人
num_cameras	integer	攝影機數量
working_directory	string	專案路徑(可選)

模組10. 執行 3D 校正

若 calibrate 為 False，則將辨識出的棋盤格點描繪在 corner 資料夾內以調整校正建構用圖片；若為 True 則產出校正檔案

calibrate_cameras(config, cbrow=8, cbcol=6, calibrate=False, alpha=0.4)

參數說明：

參數名稱	型態	說明
config	string	全域設定檔案路徑
cbrow	integer	棋盤格的列數
cbcol	integer	棋盤格的行數
calibrate	bool	是否執行校正

模組11. 確認上一步建構空間的失真程度

check_undistortion(config, cbrow=8, cbcol=6, plot=True)

參數說明：

參數名稱	型態	說明
config	string	全域設定檔案路徑
cbrow	integer	棋盤格的列數
cbcol	integer	棋盤格的行數
plot	bool	是否描繪失真結果

模組12. 使用全域設定檔案中所定義的 2D 專案，對影片進行辨識，並建立三維的辨識結果

triangulate(config,video_path,videotype="avi",destfolder=None,save_as_csv=False)

參數說明：

參數名稱	型態	說明
config	string	全域設定檔案路徑
video_path	list of list/string	辨識影片的資料夾或全路徑
videotype	string	標點影片的副檔名
destfolder	string	儲放結果的資料夾(可選)
save_as_csv	bool	是否將結果以 csv 檔案儲存(可選)

模組13. 將三維辨識結果以轉換成影片並輸出

create_labeled_video_3d(config,path,videofolder=None,start=0,end=None,videotype="avi",view=[-113, -270],xlim=[None, None],ylim=[None, None],zlim=[None, None])

參數說明：

參數名稱	型態	說明
config	string	全域設定檔案路徑
path	list	辨識結果的資料夾路徑
videofolder	string	儲放 3D 影片的位置
videotype	string	標點影片的副檔名
view	list	觀看結果的角度
xlim	list	x 座標邊界值
ylim	list	y 座標邊界值
zlim	list	z 座標邊界值

模組14. 在辨識點的資料當中套用 Kalman 濾波器

apply_kalman_filter(h5_file_path)

參數說明：

參數名稱	型態	說明
h5_file_path	string	h5 辨識點檔案路徑

模組15. 估量未辨識的魚眼

estimate_opposite_eyes(h5_file_path)

參數說明：

參數名稱	型態	說明
h5_file_path	string	h5 辨識點檔案路徑

五、 實驗結果

（一） 辨識模型訓練成果

本專題採深度學習模型 mobilenet_v2_1 作為初始模型，以 10,000 張標點資料，訓練達 1,200,000 次迭代，得出可進行辨識之模型。每次訓練完成後，能以模型與訓練資料集生成誤差評估資料，並在每 50,000 次迭代時做出誤差值之紀錄，以估量準確程度。

誤差評估資料分為誤差評估圖和平均誤差數值。藉由對比使用者提供之部位座標(圓點)與現有模型所提出之部位預測座標(十字點)，可以看出現有模型對於部位偵測的準確度。(圖 11)，平均誤差數值則是以兩種座標取距離，並在各部位、各圖片平均計算而得出。

藉由誤差值之紀錄，可統計出誤差與迭代次數之關係圖(圖 12、13)，由橫軸-疊代次數（單位：百萬）與縱軸-誤差（單位：像素）所作，並且得出相當準確的 2D 的標點辨識結果。

圖 11、誤差評估圖

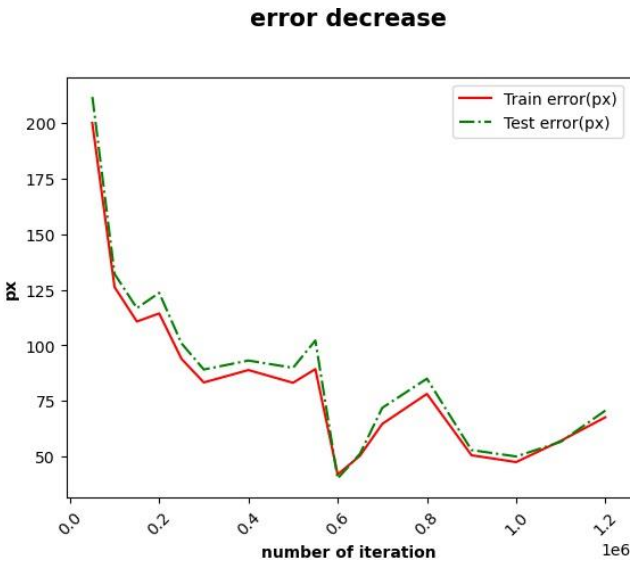

圖 12、誤差-迭代關係圖

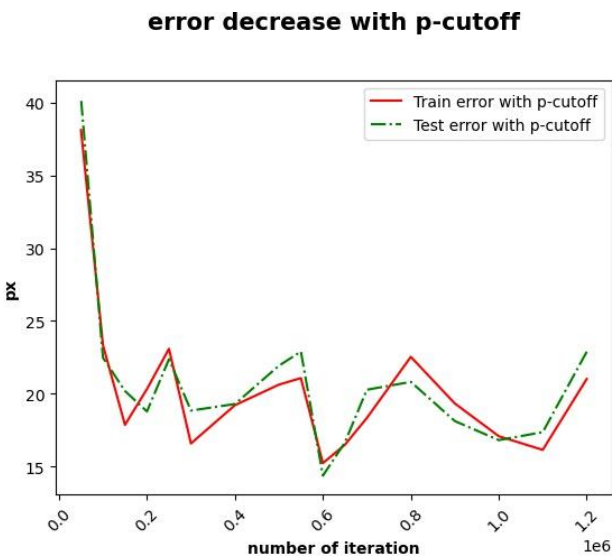

圖 13、誤差-迭代關係圖
(以 p-cutoff 扣除集端值)

（二） 三維輪廓建構成果

利用實驗(一)產生之二維辨識模型進行專案流程圖(圖 8)的 3D Project 程序，產生立體的魚輪廓影片。

多數時間 2D 的標點辨識結果正確，3D 可見立體架構。(圖 13)但部分幀會因 2D 標點結果偏移而產生 3D 偏移，或 2D 其中一張圖未辨識到點造成 3D 未產生點的狀況。

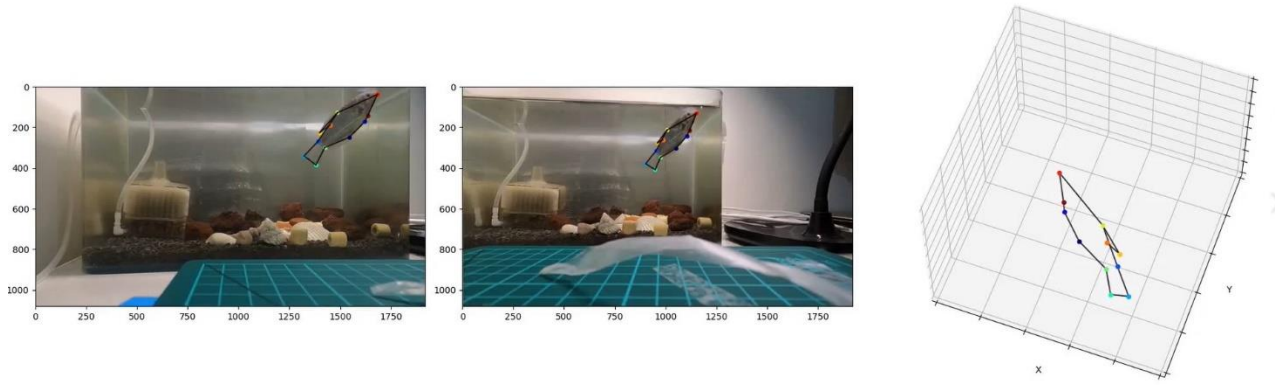

圖 14、2D 辨識圖與 3D 架構圖

（三） Kalman 濾波器修正

對實驗(二)的三維座標套用 Kalman 濾波器，將偏離理想值過多的值平滑化，同時估計未測量到的空值。

對比實驗(二)的辨識結果，未辨識出點的部分有明顯改善(圖 14)。然而在某一點長期偏移的情況下，仍可能有標點位置的錯誤產生。

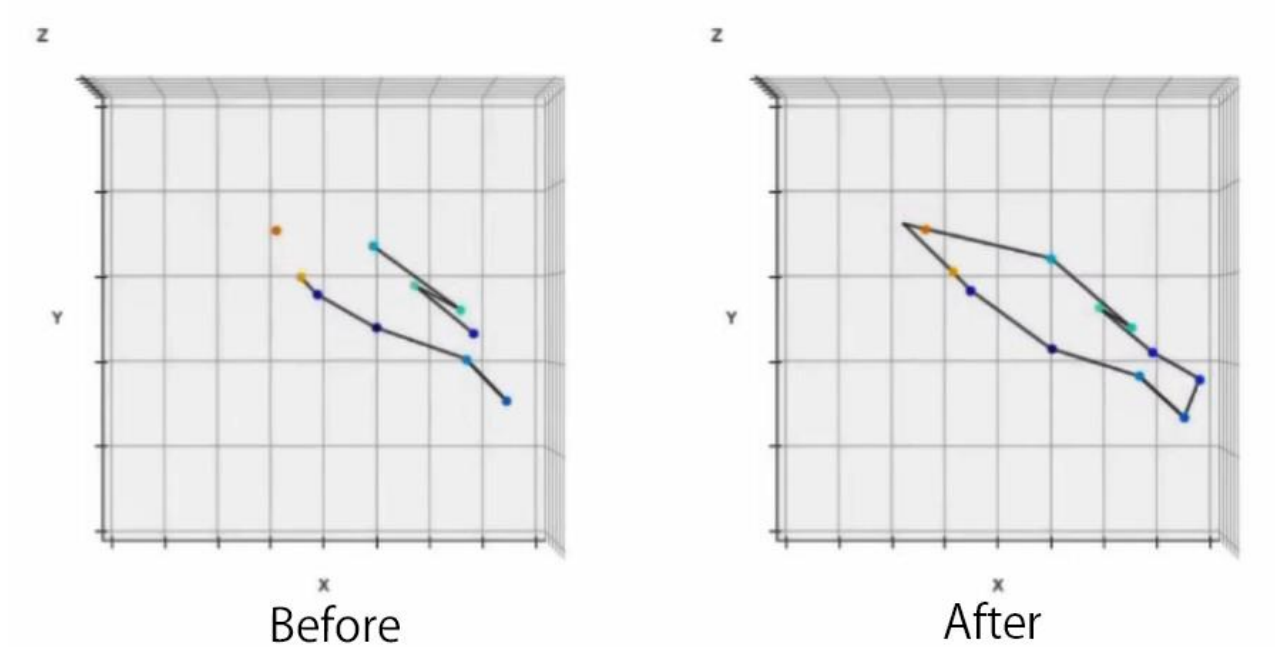

圖 14、Kalman 濾波器的前後對比圖

六、 討論

根據上述的實驗結果，我們可以看到穩定呈現的魚的輪廓，在實驗一時常出現的辨識錯誤、失敗等問題，利用 Kalman 濾波器皆得到很好的修正。

但有時還是會出現未標示及偏移的問題，我們認為是由於訓練的迭代次數不足所造成的，因為此情況在 2D 標點影片中同樣不甚理想。此問題可以透過增加訓練時間、改善硬體設備或是將作業分散到不同電腦進行，以增加訓練的效率。因為 DLC 具 conda 環境可攜性，可自訂 CPU/GPU 環境，能通用於各電腦。

同時在 3D 呈現的結果中，有時仍會出現較明顯的不對稱或是不自然的模樣，我們可以透過修改魚眼估計的取點依據做改善。

另外，受限於疫情的影響，我們只能自己養魚拍攝影片，在有限的情況下改變環境，資料多樣性不足，換一個魚缸環境錯誤率便會大幅增加，並且希望接下來能夠嘗試辨識青萬隆以外更多的魚種。

七、 結論

本專題以三維追蹤魚隻位置來發想，透過像是圖像特徵檢測、相機校準、Kalman 濾波器等等深度學習與立體視覺原理及工具，來完成這次主題，通過這些我們了解到 3D 技術中的基礎，未來若是有機會再更進一步鑽研，結合不同類型的技術，如加上不同魚種的建模，或如 3D 投影、3D 列印等等技術應用來延伸創作，把成果化為實品，又能打造不同的成果，未來發展性可期。

專題相關資料：https://github.com/Yukimi27043816/Sky-eye-of-Bikini-Bottom

展示影片：https://youtu.be/Ha3jAxk8U48

參考文獻

[1] Xiang, R., Jiang, H., & Ying, Y. (2014). Recognition of clustered tomatoes based on binocular stereo vision. Computers and Electronics in Agriculture, 106, 75-90.

[2] Cao, Z., Hidalgo, G., Simon, T., Wei, S. E., & Sheikh, Y. (2019). OpenPose: realtime multi-person 2D pose estimation using Part Affinity Fields. IEEE transactions on pattern analysis and machine intelligence, 43(1), 172-186.

[3] Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. Nature neuroscience, 21(9), 1281-1289.

[4] Zhang, Z. (2000). A flexible new technique for camera calibration. IEEE Transactions on pattern analysis and machine intelligence, 22(11), 1330-1334.

[5] Aslam, A., & Ansari, M. (2019). Depth-map generation using pixel matching in stereoscopic pair of images. arXiv preprint arXiv:1902.03471.

[6] Zou, L., & Li, Y. (2010, November). A method of stereo vision matching based on OpenCV. In 2010 International Conference on Audio, Language and Image Processing (pp. 185-190). IEEE.

[7] Kar, A., Häne, C., & Malik, J. (2017). Learning a multi-view stereo machine. arXiv preprint arXiv:1708.05375.

[8] Welch, G., & Bishop, G. (1995). An introduction to the Kalman filter.

[9] Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., & Schiele, B. (2016, October). Deepercut: A deeper, stronger, and faster multi-person pose estimation model. In European Conference on Computer Vision (pp. 34-50). Springer, Cham.

[10] Mathis, A., Biasi, T., Schneider, S., Yuksekgonul, M., Rogers, B., Bethge, M., & Mathis, M. W. (2021). Pretraining boosts out-of-domain robustness for pose estimation. In Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (pp. 1859-1868)