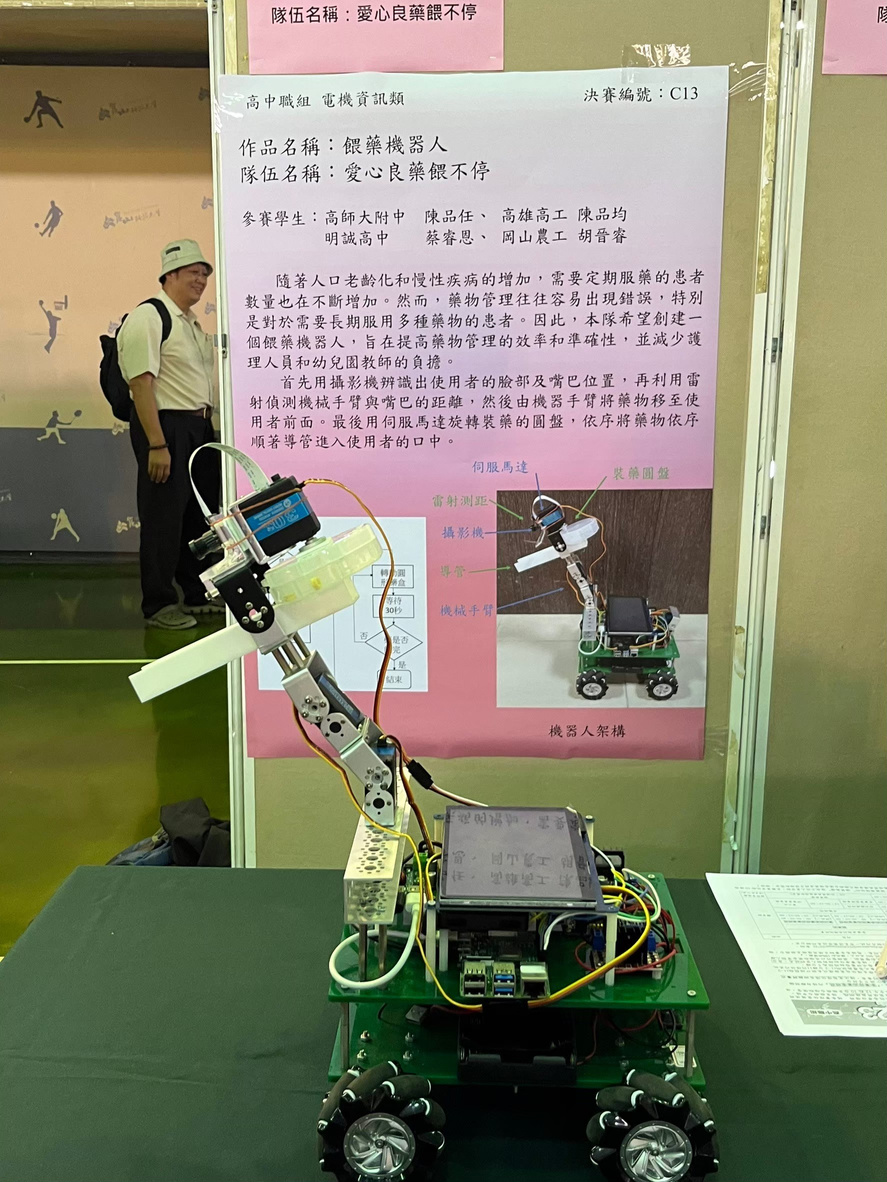
**醫療科技化及臨床照護自動化**

**餵藥機器人專題學習歷程**



**一、專題起點與靈感**

**我的專題靈感源自於醫療照護領域的實際需求，尤其是針對行動不便或記憶力受限者的服藥挑戰。服藥過程若能自動化，不僅能減少人為疏忽，還能提升生活便利性。我決定設計一個餵藥機器人，能自動偵測使用者的嘴巴位置，並精準投餵藥物，結合影像辨識、機械控制和通訊技術，實現技術與醫療價值的結合。我們的願景是讓機器人自動到達患者身邊，為病床患者投餵藥物並控制劑量，以提升醫療照護的效率與安全性。**

**二、設計與規劃**

**專題設計了一個由兩部分組成的系統：**

**樹莓派（Raspberry Pi）與上半部手臂（由我負責）：**

* **使用 USB 攝影機與 OpenCV 進行 AI 臉部與嘴巴偵測。**
* **計算嘴巴座標 (mouth\_x, mouth\_y)，並透過 UART 傳給 Arduino。**
* **顯示即時影像，方便測試與調整。**

**Arduino 與下半部硬體（由其他組員負責）：**

* **接收座標，控制 MG995 伺服馬達移動機械手臂。**
* **操作另一伺服馬達驅動藥盤，釋放藥物並控制劑量。**

**硬體清單：**

* **樹莓派（主控板）**
* **USB 攝影機（影像輸入）**
* **MG995 伺服馬達（機械手臂）**
* **標準伺服馬達（藥盤旋轉）**
* **Arduino（動作控制）**
* **透明藥盤與導管（藥物傳輸）**

**軟體工具：**

* **Python 3（搭配 OpenCV，負責人臉辨識部分）**
* **Arduino IDE（編寫 C++ 程式，負責機械控制部分）**

**由於時間限制，原本計畫中的機體行動功能（使用萬向輪實現自動移動）最終未能實現，但這不影響我們專注於核心功能：精準的人臉與嘴巴偵測，以及藥物投餵與劑量控制。我負責上半部的人臉辨識與樹莓派開發，專注於 AI 影像處理技術。**

**三、製作過程**

**製作過程分為軟體開發（我負責人臉辨識部分）與硬體組裝（由其他組員負責）兩個階段，並記錄實作中可能遇到的問題與解決方案。**

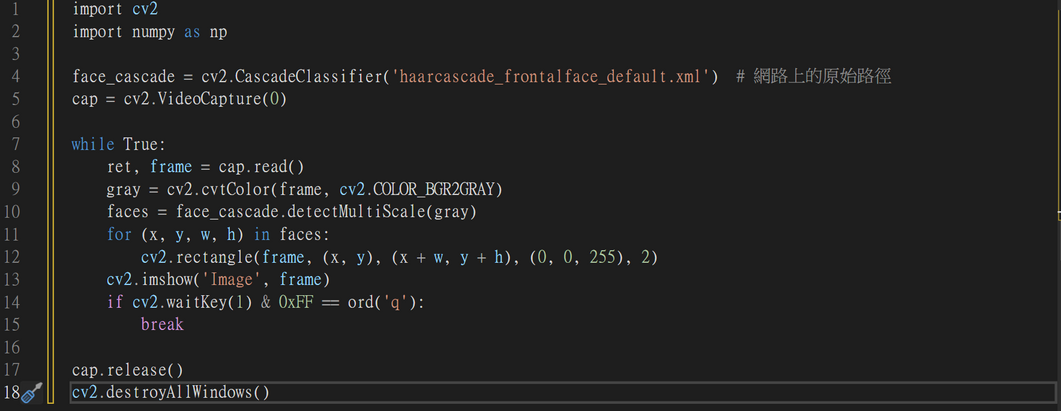
**1. 軟體開發（我負責）**

**樹莓派程式（Python）與 AI 人臉辨識**

**我使用 OpenCV 的 Haar Cascade 進行人臉與嘴巴偵測，開發過程中遇到以下實際問題，並展現我在人臉辨識領域的技術能力與成長：**

**問題 1：程式碼語法錯誤與攝影機無法開啟（初步測試程式）**

**我一開始測試時，螢幕上什麼都沒顯示，也沒有偵測的外框。原程式為：**

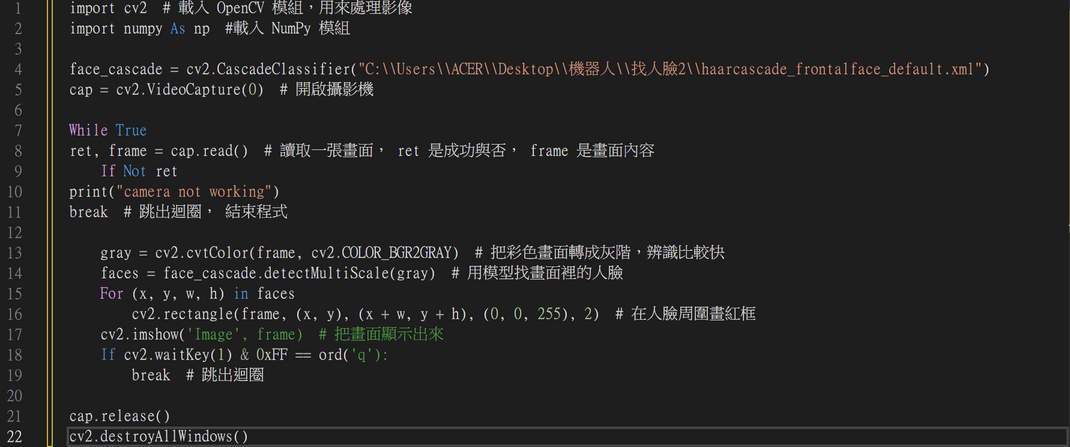


**原因**：我直接從網路上抄了一段程式碼下來，為 haarcascade\_frontalface\_default.xml

是別人電腦的路徑，自己電腦找不到這個檔案，應該要用我自己的路徑

"C:\\Users\\ACER\\Desktop\\機器人\\找人臉2\\haarcascade\_frontalface\_default.xml"。

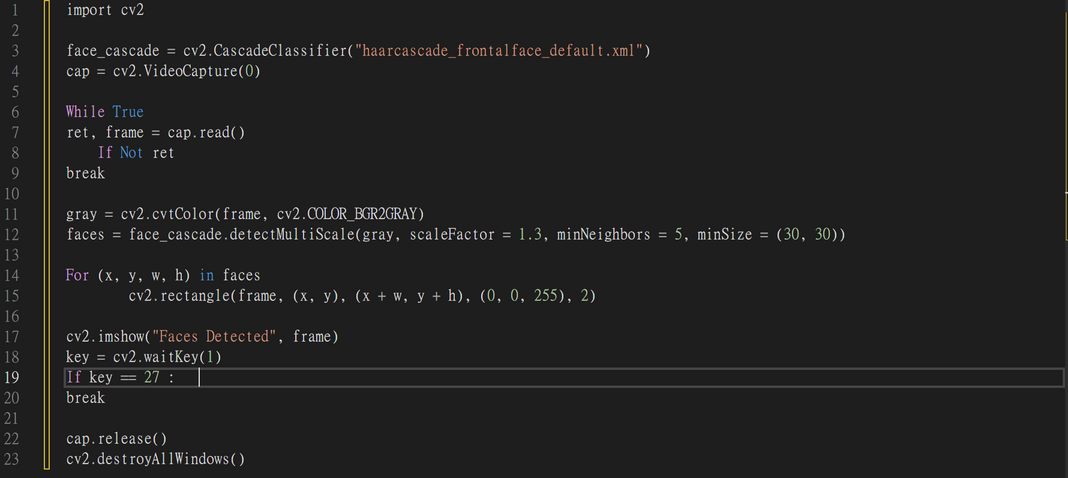
**解決方案**：查閱 OpenCV 官方文件（[https://docs.opencv.org/）與技術論壇，修正程式並檢查硬體：](https://docs.opencv.org/%EF%BC%89%E8%88%87%E6%8A%80%E8%A1%93%E8%AB%96%E5%A3%87%EF%BC%8C%E4%BF%AE%E6%AD%A3%E7%A8%8B%E5%BC%8F%E4%B8%A6%E6%AA%A2%E6%9F%A5%E7%A1%AC%E9%AB%94%EF%BC%9A)



\*加上了自己對於程式的理解及注釋，並請Chatgpt修改部分內容

**問題 2：人臉偵測不穩定（實時人臉偵測程式）**

**在實現實時人臉偵測時，程式偶爾無法偵測到人臉，或誤將背景識別為人臉。原程式如下：**



**原因**：路徑設定錯誤、光線不足或參數不適合當前環境，退出按鍵與專題需求不一致。

**解決方案**：修正路徑為 cv2.data.haarcascades + 'haarcascade\_frontalface\_default.xml'，

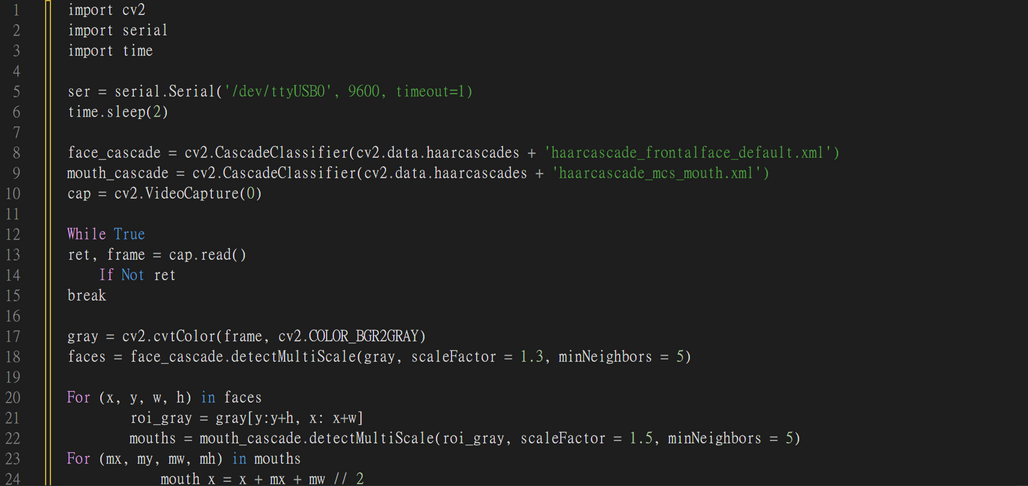
調整參數（scaleFactor=1.1, minNeighbors=3, minSize=(50, 50)），並統一退出按鍵為 'q'：

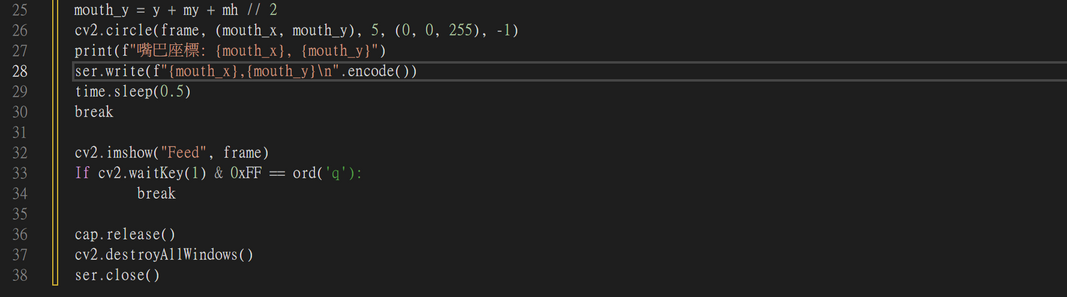


\*第8和12行因為尚未理解全，請Chatgpt修飾並補充加註，經過細膩解釋，我也理解了這兩行的程式邏輯，判斷是否成功並將照片轉為方便機器辨識的灰階數據影像。

問題 3：嘴巴偵測失敗與 UART 傳輸錯誤（最終專題程式）

在整合人臉與嘴巴偵測並傳送座標給 Arduino 時，嘴巴偵測不穩定，且 UART 傳輸偶爾失敗。原程式如下：





\*UART（通用非同步收發器）是一種 **串列通訊** 方式，適合短距離數據交換。

可能接收失敗原因: UART 緩衝區溢出或者是硬體連接問題。

**原因**：Haar Cascade 參數設定過於嚴格

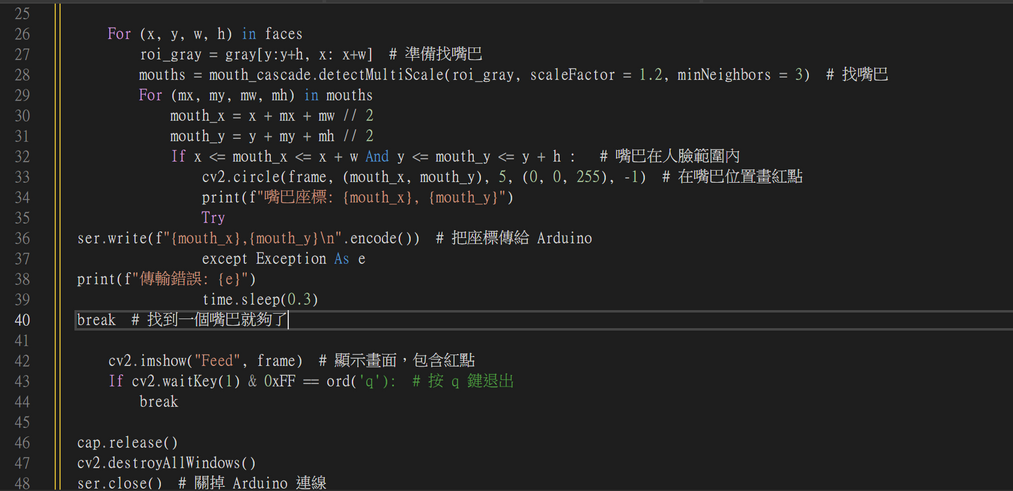
（scaleFactor=1.5, minNeighbors=5），UART 路徑 /dev/ttyUSB0 可能錯誤，

且未加入傳輸錯誤處理。

**解決方案**：調整參數為 scaleFactor=1.2 和 minNeighbors=3，限制嘴巴在人臉範圍內，

並檢查 UART 路徑（發現應為 /dev/ttyUSB1）。修正後程式：





**2. 硬體組裝（由其他組員負責）**

由於專注於人臉辨識，我並未直接參與硬體組裝，但與其他成員合作確認系統整合。我了解硬體部分（樹莓派、Arduino、伺服馬達、藥盤）與軟體的協作，確保人臉辨識數據能順利傳輸與應用。

**問題1：**手臂跟藥盤裝不穩，老是搖晃

開始組裝時，發現藥盤表面光滑，無法直接將機械手臂和藥盤接合，稍微動一下就搖搖晃晃，藥還會掉出來。

<https://down-tw.img.susercontent.com/file/tw-11134207-7r98t-lmh0y8p46ahl09>

\*我們使用**分隔式旋轉藥盤**作為機器手臂前端來投餵藥品

**原因:** 手臂跟藥盤的接面不平整，使馬達帶動轉盤時，藥丸容易卡住或掉出去。

**解決方案：**我們直接在轉盤上打上螺絲孔，還加了幾個小墊片讓藥盤固定得更牢，也使馬達可直接帶動轉盤，解決因震動而造成的不穩定。

**反思:** 雖然將手臂與藥盤以螺絲鎖死能夠解決手臂穩定度的問題，我們在後續的操作卻發現了弊端 :

1. 無法方便拆除藥盤和進行清理

2. 假想醫院實際應用情況，更換藥品過程太過繁瑣，效率低下，反而增加護理人員的工作量和麻煩。

**問題：**機械手臂與藥盤動作不穩定

測試時，機械手臂與藥盤的動作偶爾不穩，藥物釋放失敗。

**原因：**考慮到軟體和硬體方面，我們設想可能是伺服馬達角度或馬達的供電不足。

**解決方案：**我們試著調整 servoArm.write(145) 和 servoDisk.write(100) 的角度，並確保 MG995 伺服馬達使用獨立 5V 電源，使穩定性提升。

**反思:** 雖然嘗試使MG995 伺服馬達使用獨立 5V 電源，但實際測量時，電壓卻無法達到5V，經過組員們討論和上網查詢相關資料，發現不只我們遇到相關問題，可能是因為樹梅派的電路不穩定，導致馬達的動力大小不一，導致有抖動相關動作。

**四、測試與改進**

我負責人臉辨識部分的測試，觀察系統在不同光線、距離和使用者位置下的表現，並記錄以下結果：

**成功之處：**

機器人能夠正確偵測臉部與嘴巴，顯示即時影像並標記目標，數據精準傳至 Arduino，實現投餵藥物的設定功能。

從照片中的展示場景（展覽看板與實際操作）可以看到人臉辨識部分穩定運行，獲得正面回饋。

**AI 人臉辨識的挑戰與改進：**

* 光線變化：在昏暗環境下，Haar Cascade 偵測失敗率高。我調整環境光線並優化參數（scaleFactor=1.1, minNeighbors=3），但仍考慮升級至深度學習模型（如 MTCNN )
* 角度與遮擋：側臉或遮擋時，嘴巴偵測失敗。我限制嘴巴在人臉範圍內，穩定性提升。
* 計算效率：Haar Cascade 在樹莓派上運行尚可，但若加入複雜功能，可能需要優化程式或使用 Edge TPU/GPU 加速。(由chatgpt建議)

**硬體與動作的挑戰與改進（由其他組員負責）：**

* 伺服馬達角度校正：機械手臂與藥盤角度需多次測試，最終定為 145 度和 100 度。
* 傳輸延遲：縮短 time.sleep(0.3) 後，動作更連貫，但偶爾有資料丟失，計畫加入緩衝機制。 \*增加緩衝區（Buffer）機制，確保即使程式短暫延遲，也不會錯過 UART 傳來的數據。
* 藥物釋放問題：調整藥盤角度與導管設計後，藥物釋放穩定性提升。

**五、心得與反思**

**作為專題中負責人臉辨識（上半部手臂相關功能）的成員，在這次的專案設計中我學會如何整合 AI 影像辨識（OpenCV Haar Cascade）、通訊協作（UART）與硬體系統的技術。從設計草圖到程式開發，再到測試與優化，深刻體會到影像處理的複雜性與樂趣。而其中最大的挑戰是實作中遇到的技術問題（例如程式語法錯誤、參數調整困難、光線影響），但這些挑戰也成為未來研發挑戰的經驗。**

**透過查閱技術文件（OpenCV 官方文件、Stack Overflow）、測試與優化，逐步克服這些困難，學會如何 debug 與改進系統。雖然 Haar Cascade 效率高，但其在速度與傳輸方面的限制讓我意識到未來需要探索深度學習技術（如 MTCNN），提升系統在醫療場景中的效率和多元應用面。**

**從照片中看到機器人在比賽中的表現，我感到非常有成就感，但也意識到人臉辨識仍有改進空間。**

**例如，系統應更穩健地應對複雜環境（光線、角度變化），並進一步提升精準度。展示時使用者的回饋（光線影響、辨識速度等）提醒我，未來需要考慮更多醫療場景的需求。**

**未來優化方向：**

* **升級 AI 模型至深度學習方案（如 MTCNN，**[**https://github.com/ipazc/mtcnn），提升在光線與角度下的穩定性。**](https://github.com/ipazc/mtcnn%EF%BC%89%EF%BC%8C%E6%8F%90%E5%8D%87%E5%9C%A8%E5%85%89%E7%B7%9A%E8%88%87%E8%A7%92%E5%BA%A6%E4%B8%8B%E7%9A%84%E7%A9%A9%E5%AE%9A%E6%80%A7%E3%80%82)
* **加入實時人臉追蹤功能（OpenCV KCF/CSRT** [**https://docs.opencv.org/master/d9/df8/tutorial\_root.html），改善連續動作的精準度。**](https://docs.opencv.org/master/d9/df8/tutorial_root.html%EF%BC%89%EF%BC%8C%E6%94%B9%E5%96%84%E9%80%A3%E7%BA%8C%E5%8B%95%E4%BD%9C%E7%9A%84%E7%B2%BE%E6%BA%96%E5%BA%A6%E3%80%82)
* **優化樹莓派的運算效率，可能使用 Edge TPU 或 GPU 加速模組。**
* **開發更智能的劑量控制邏輯，結合人臉辨識與藥物管理系統，實現精準醫療。**

**照片中的技術細節與應用價值（聚焦人臉辨識）**

* **主板與電路板：照片中的綠色電路板顯示了樹莓派與 Arduino 的整合，線路整齊但需要進一步束縛，這反映了在電路設計中的學習與改進。樹莓派負責人臉辨識，與電機工程中的穩定性與可靠性的要求一致，特別是在醫療設備中至關重要。**
* **USB 攝影機與 AI：攝影機與 OpenCV 的結合實現了人臉與嘴巴偵測，這是醫療監控的核心技術。在這次的專題設計中，因參數設定不熟悉導致偵測失敗，但透過測試與學習逐步優化，展現了影像處理能力。**
* **上半部手臂（我負責）：照片中的機械手臂結構顯示了 USB 攝影機與樹莓派的整合，我負責確保人臉辨識數據精準傳輸至下半部硬體，這與醫療自動化的需求高度相關。**
* **下半部硬體（其他組員負責）：照片中的藥盤與伺服馬達由其他成員負責，實現藥物投餵與劑量控制，與我的人臉辨識部分協作完成系統功能。**