# CSL HW1 Report

Team Name: 我沒有頭緒

Team Members: B10902006 詹子慶、B10902067 黃允謙、B10902083 張程凱、B10902136 陳妍姍

Demo Video Link: https://www.youtube.com/watch?v=bGaYpATnB6o

**1. Summarize what you have learned in this lab**

[Theory]

- Understood the principle of total internal reflection in the design of touchpad technology

[Hardware]

- Understood the connection of basic circuits, including batteries, DuPont wires, resistors, LED lights, and a camera
- Understood how to add DuPont connectors on wires
- Understood how to use a heat gun to bend the arcylic sheet

[Software]

- Understood the basic usage of packages like OpenCV
- number detection:
  Understood how to extract features from handwritten digits and use a decision tree to classify them
- gesture detection:
  Used geometrical properties to identify each gesture. For example, we consider how the slope between two points changes over time for rotation, and we also consider the centroid of touch points to distinguish between similar gestures. We also used timers to calculate how many frames we should expect between double taps.
- finger detection:
  Understood how to use relative displacement between consecutive frames to identify the same finger ID.

**2-1. What you did**

[Hardware]

Mostly the same as what we learned. One special technique we employed was using a white paper as a liner inside a black plastic bag, which enhanced the reflection effect significantly.

[Software]

The structure of our codebase is listed as follows:

```
lab1
 |---- main.py
 |---- const.py
 |---- detector.py
 |---- number_detector.py
 |---- gesture_detector.py
 |---- finger_detector.py
```

`number_detector.py` performs real-time digit recognition, which means that the system predicts the results as the digits are written; `gesture_detector.py` detects which gesture (tap, double tap, long press, scroll, swipe, zoom in/out, and rotate) is written; and `finger_detector.py` identifies which detected points belong to the same finger in consecutive frames. The detectors in the three files are all inherited from a base class `Detector`.

In each iteration, `main.py` calls each of the three detectors and shows the predicted results in separate windows.

**2-2. How you can improve this device**

1. More accurate and robust digit recognition
   To ensure that our device can recognize different handwriting styles and strokes for universal usage, we can use machine learning to train a more advanced classifier. However, the current implementation is low-cost and can perform pretty well under regular scenarios already.
2. Higher sensitivity
   The sensitivity of the screen could still be improved to ensure smoother and more continuous frames without interruptions. We have tried using water or tapping / pressing harder, but enhancing the camera's resolution or refining certain aspects of the algorithm might provide better results.

**3. Some feedback for this lab**

- Thank you Professor and TAs! The ideas in class are clearly delivered and the instructions in TA sessions were also very helpful.