

嵌入式系統實驗

實驗二報告

Tessel 2

電機三 童 寬 B03901039

電機三 李元顯 B03901093

教授：鄭振牟

繳交日期：2017/5/5

壹、功能

1. 水平儀數值讀取

此為本程式的基礎功能。利用 Tessel 2 的加速規模組，我們的程式可以讀取當下開發板的方向、位置及加速度狀態。

2. 攝影機畫面讀取

利用與 Tessel 2 相容的 USB 網路攝影機，我們的程式以網頁形式呈現攝影機所拍攝到的實時畫面。搭配水平儀以及網頁的訊息提示使用則可以協助使用者校正開發板的位置。

3. 調節水平儀靈敏度

在使用水平儀校正時，使用者可以從網頁介面自行設定水平儀的靈敏度：靈敏度愈高則發出校正警告的偏移容許值愈低，反之亦然。

4. 視覺校正協助提示

由於水平儀可以搭配攝影機使用，當開發板的並非水平而有偏移時，網頁中會出現以攝影機相同的視角而言，應向哪個方向旋轉才可恢復水平狀態。當完全恢復水平狀態（亦即偏移值低於自訂之靈敏度時）後，校正協助提示便會消失。

5. 燈號校正協助提示

由於 Tessel 2 開發板上即有可開關的 LED 燈號，除了 4.所述的網頁介面提示外，使用者也可以直接觀察開發板上的燈號得知目前的水平狀態。若 LED 燈號熄滅則代表目前未有用戶端連線使用，或是目前偏離水平狀態較大，需先以目測大致調正；若 LED 燈號閃爍則代表目前 x 或 y 的其中一個方向仍需繼續校正；若 LED 燈號長亮則代表目前開發板為水平狀態（依自行設定之靈敏度而定）。

6. 自行設定校正基準值

使用者可以透過網頁介面設定水平校正的基準值，即當需要參考的基準並非實際水平時，可以將開發板事先旋轉至所欲校正的基準後按下「Set」按鈕，水平儀便會以該狀態為基準歸零。如需將基準值重設為實際水平，則可按下「Reset」按鈕。

7. 輔助照明燈號

部分市面上的網路攝影機附有前照明設備，若水平儀於環境光線較昏暗的情況下使用時可以透過網頁介面上的按鈕開啟輔助使用。由於實驗室所提供的網路攝影機並未具此功能，我們在程式中以控制 Tessel 2 開發板上內建的藍色 LED 燈號取代；若插入具照明功能的網路攝影機則只需稍微更改程式碼即可使用。

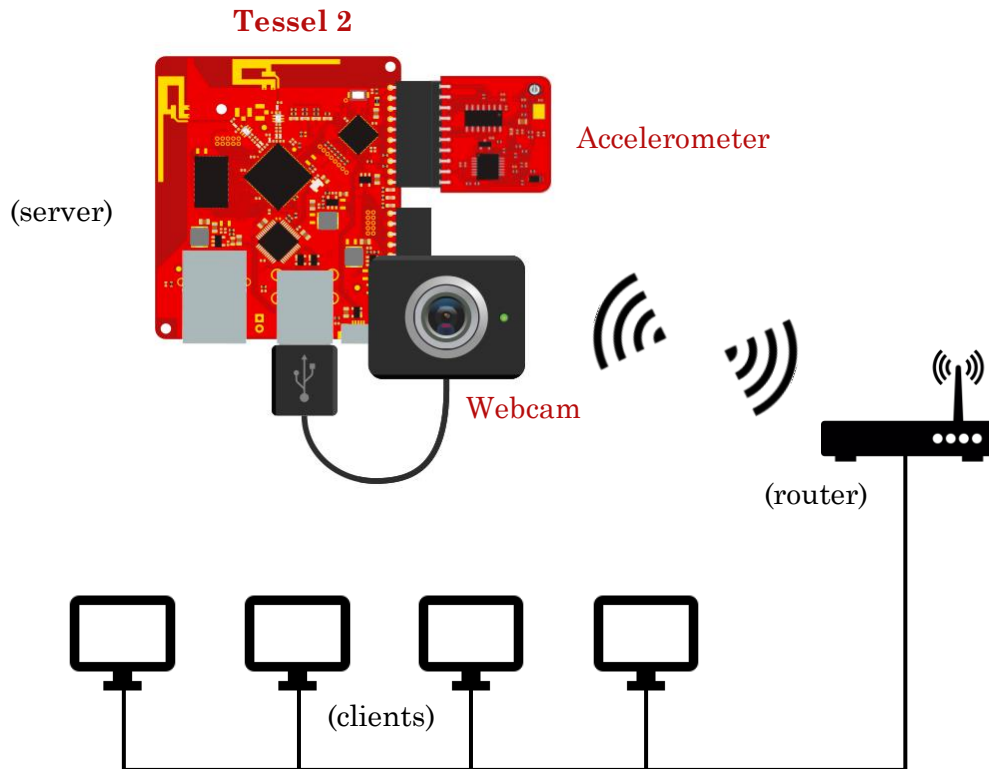
貳、架構

如下圖所示，本程式的架構非常簡潔明瞭。

我們以 Tessel 2 開發板作為硬體中樞，利用有線方式連接加速規模組以及網路攝影機，所讀取的數值即拍攝的影像即直接由 Tessel 2 獲得；同時，它也是程式的伺服器端（server），負責將讀取的資料送至用戶端（client）或是將用戶端傳來的設定值記錄並儲存。

我們使用網頁形式作為主要的使用者介面，當使用者與 Tessel 2 位於同一區域網路內並自本機瀏覽器連接至開發板的 IP 位置時，用戶端網頁便會顯示並提供視覺化操作介面。

在本次實驗中，我們將 Tessel 2 連至區域網路內的無線路由器而未使用 AP 模式。



1. 伺服器與用戶端間的資料傳輸

我們延續 Lab1 的結構，使用 socket.io 來實現伺服器與用戶端之間的資料傳輸。

```
1. // in server.js
2.
3. var express = require('express');
4. var app = express();
5. var http = require('http').Server(app);
6. var io = require('socket.io')(http);
7.
```

```

8. app.use(express.static(`${__dirname}/public`));
9.
10. ...
11.
12. http.listen(3000, function() {
13.     console.log('listening on *:3000');
14. });

```

如此一來，我們即可透過熟悉的 `socket.on` 以及 `socket.emit` 等函數將伺服器端（Tessel 2）所測得的資料傳輸至用戶端、或是將用戶端的設定值傳輸至伺服器端。

2. 加速規數值讀取

我們所設計的水平儀最重要的資料來源便是 Tessel 2 所讀取到來自加速規模組量測得到的結果。加速規共可測得三個維度的值，分別為 x 、 y 、 z ：在本次實驗中，我們只考慮靜止狀態下的水平測量，不須使用到加速規的 z 值（即測量上下垂直移動的加速度）。有關加速規的程式碼略列如下：

```

1. // in server.js
2.
3. accel.on('ready', function() {
4.     accel.on('data', function(xyz) {
5.         socket.emit('newdata', {
6.             x: xyz[0],
7.             y: xyz[1],
8.             z: xyz[2],
9.         });
10.    });
11. });

```

`accel` 為 Tessel 2 之加速規模組內建的函數，而當有讀值產生時，`accel.on('data')` 便會被呼叫並傳入讀到的三個值（存於變數 `xyz`）。同時，我們將讀到的值透過 `socket.emit('newdata')` 傳至用戶端以供分析。

3. 輔助照明燈號開閉

使用者從用戶端可以控制輔助照明燈的開與關。在本實驗中，我們以 Tessel 2 內建的藍色 LED 燈號（即 `tessel.led[3]`）代替。伺服器端的程式碼略列如下：

```

1. // in server.js
2.
3. socket.on('ledon', function() {
4.     tessel.led[3].on();
5. });
6.
7. socket.on('ledoff', function() {
8.     tessel.led[3].off();
9. });


```

用戶端的程式碼略列如下：

```

1. // in main.js
2.
3. handleFlashToggle = () => {
4.   this.setState({ flashed: !this.state.flashed });
5.   if (!this.state.flashed) this.context.socket.emit('ledon');
6.   else this.context.socket.emit('ledoff');
7. }

```

如要開關，只需點選網頁介面中的按鈕（）。

4. 水平儀靈敏度調整

使用者從用戶端可以自訂水平儀的靈敏度（存於變數 `thres`），我們在網頁介面中設置了可拖曳的按鈕，可設置的靈敏度範圍介於 0 至 0.5 之間。因各用戶端使用的靈敏度可能不同，其值是儲存於用戶端的一個變數中，並未傳送至伺服器；同理，水平狀態的判斷也是於用戶端執行。用戶端的程式碼略列如下：

```

1. // in main.js
2.
3. class Tessel extends Component {
4.   constructor(props, context) {
5.     this.state = {
6.       thres: 0.045,
7.     };
8.   }
9.
10.  ...
11.
12.  handleSlider = (ev, val) => {
13.    this.setState({ thres: val });
14.  }
15. }

```

5. 網路攝影機影像顯示

將網路攝影機固定於開發板並將鏡頭朝向加速規所指的前方之後，便可以做為水平儀校準的視覺輔助。我們使用 Tessel 2 官方網站的 `stream` 函數將網路攝影機攝得的畫面串流至用戶端的網頁中。伺服器端的程式碼略列如下：

```

1. // in server.js
2.
3. var camera = new av.Camera();
4.
5. app.get('/stream', (request, response) => {
6.   response.redirect(camera.url);
7. });

```

6. 水平狀態判斷與顯示

在網頁介面中，我們設有兩個狀態顯示區，分別代表 x 、 y 兩個方向是否達到水平：若達到水平，則顯示「nor」，反之則依讀值為正偏向或負偏向分別顯示「pos」或「neg」。另外，水平時顯示區之底色為綠，反之為紅。

由於我們選擇完全在用戶端決定靈敏度、偏移校準值而不牽涉伺服器，因此水平狀態顯示的判斷式也是位於用戶端。我們將判斷的結果以字串形式分別儲存至 `x_orient` 以及 `y_orient` 兩變數中。其程式碼略列如下：

```
1. // in main.js
2.
3. if (Math.abs(this.state.x) > thres) {
4.   if (this.state.x > 0) this.setState({ x_orient: "pos" });
5.   else this.setState({ x_orient: "neg" });
6. }
7. if (Math.abs(this.state.y) > thres) {
8.   if (this.state.y > 0) this.setState({ y_orient: "pos" });
9.   else this.setState({ y_orient: "neg" });
10. }
```

此外，我們也有在開發板端設置水平狀態的 LED 燈號提示（在此使用綠色燈號，即 `tessel.led[2]`）。這部分需要將判斷結果透過 `socket.io` 傳送至伺服器端，以另 Tessel 2 開發板的燈號閃爍、開啟或關閉。其程式碼略列如下：

```
1. // in main.js
2.
3. if (this.state.x_orient === 'nor' && this.state.y_orient === 'nor')
4.   this.context.socket.emit('greenon');
5. else if (this.state.x_orient !== 'nor' && this.state.y_orient !== 'nor')
6.   this.context.socket.emit('greenoff');
7. else
8.   this.context.socket.emit('greenblink');
9.
10. // in server.js
11.
12. socket.on('greenon', function() {
13.   tessel.led[2].on();
14. });
15.
16. socket.on('greenoff', function() {
17.   tessel.led[2].off();
18. });
19.
20. socket.on('greenblink', function() {
21.   ledBlink();
22. });
```

7. 設定以及重設校正基準值

在網頁介面的下方，使用者可以分別透過點選「Set」以及「Reset」按鈕來設定和重設水平儀的校正基準值。當「Set」被按下後，當下的傾斜角即會被存入 `x_cali` 和 `y_cali` 兩變數中，在將來判斷水平狀態時便會扣除此基準值。若要重設為預設水平，則可按下「Reset」按鈕，`x_cali` 和 `y_cali` 便會重設為初始值 0。其位於用戶端的程式碼略列如下：

```
1. // in main.js
2.
```

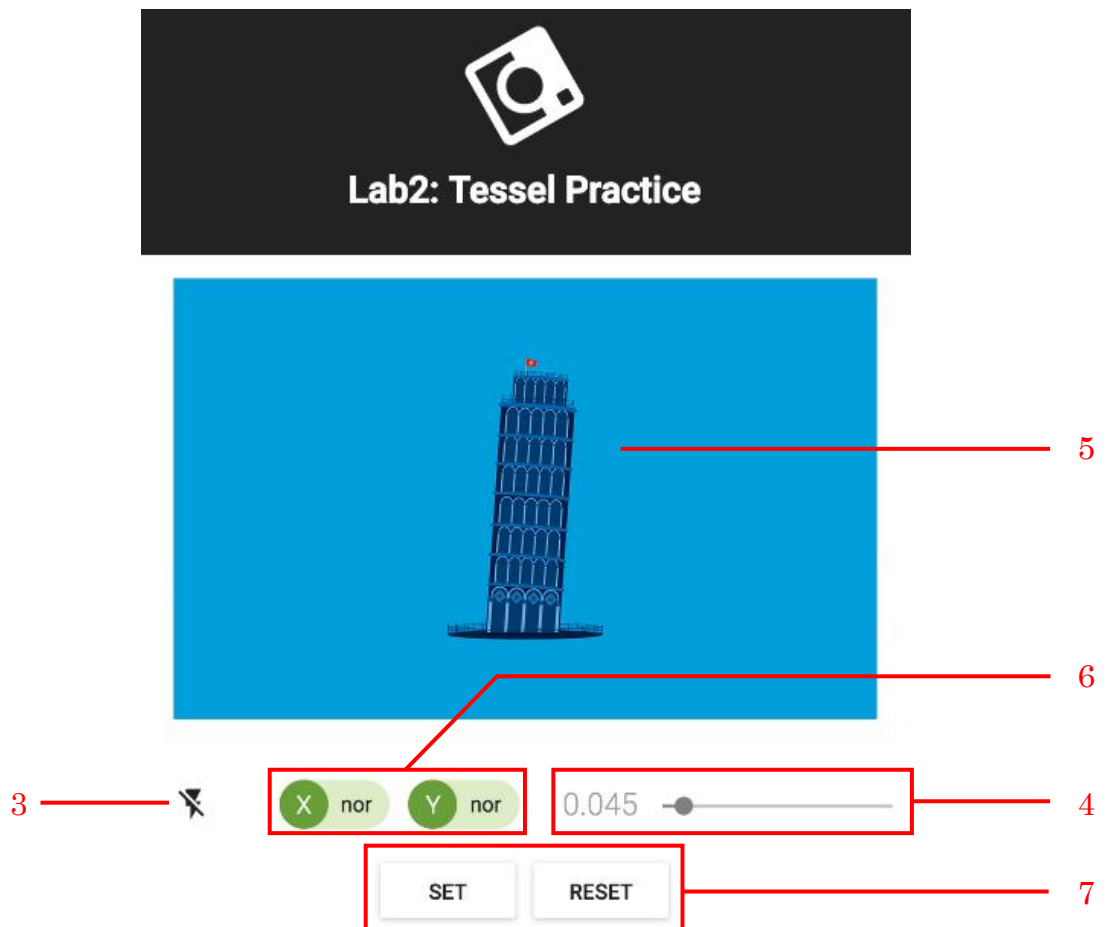
```

3. handleClick = () => {
4.   this.setState({
5.     x_cali: this.state.x + this.state.x_cali,
6.     y_cali: this.state.y + this.state.y_cali
7.   })
8. }
9.
10. handleResetClick = () => {
11.   this.setState({
12.     x_cali: 0,
13.     y_cali: 0
14.   })
15. }

```

參、使用者介面

下圖即我們的網頁介面，其中個別元素的標號即對應第貳節所詳列的功能。



我們以 React、ReactDOM 搭配 material-ui 完成前端的部分。

React 是由 Facebook 推出，專注於 UI 的 JavaScript 函式庫。在 React 我們會以 JSX 的格式來實作每個元素，並透過 state、props 的設定與傳送，去控制網頁上每個元素的行為。

ReactDOM 一樣是 Facebook 推出的 JavaScript 函式庫，裡面有很多處理 DOM 的函式，例如在特定的 DOM node 插入 JSX。我們就是在 `index.html` 裡設一個特別的 `div`，並使用 ReactDOM 的 `render` 函式插入整個介面的 JSX 內容。

我們使用 `create-react-app` 來設立開發環境，它幫我們整合了 `babel`、`webpack` 等功能。

肆、參考資料

1. “socket.io docs”, <https://socket.io/docs/>, 2017/03
2. “Tooltipster”, <http://iamceege.github.io/tooltipster/>, 2017/03
3. “React - A JavaScript library for building user interfaces”, <https://facebook.github.io/react/>, 2017/04
4. “Tessel 2 docs”, <http://tessel.github.io/t2-start/index.html>, 2017/04
5. “Material-UI”, <http://www.material-ui.com/>, 2017/04