

3D Computer Graphics Lab 3  code quality  A

code quality A

ity A

系級：資工4B

學號：105502042

姓名：張博堯

# STDOUT 輸出

輸出處理 `display` 指令花的時間，以毫秒(*ms*)計算。

# 截圖

Lab3A.in

2.

```

File Edit Selection View Go Debug Terminal Help
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3A.in
inst all six planes
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3B.in
Press any key to continue . .
display takes: 4ms
Press any key to continue . .
display takes: 4ms
[c1 >= 0 && c2 >= 0) {
    Press any key to continue . .
    / (c1 - c2) * (p - s));
t2in

```

```

File Edit Selection View Go Debug Terminal Help
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3A.in
inst all six planes
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3B.in
Press any key to continue . .
display takes: 4ms
Press any key to continue . .
display takes: 4ms
[c1 >= 0 && c2 >= 0) {
    Press any key to continue . .
    / (c1 - c2) * (p - s));
t2in

```

## Lab3B.in

1.

```

File Edit Selection View Go Debug Terminal Help
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3A.in
inst all six planes
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3B.in
Press any key to continue . .
display takes: 4ms
Press any key to continue . .
display takes: 4ms
[c1 >= 0 && c2 >= 0) {
    Press any key to continue . .
    / (c1 - c2) * (p - s));
t2in

```

```

File Edit Selection View Go Debug Terminal Help
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3A.in
inst all six planes
D:\3D-Computer-Graphics-Lab3\2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe Lab3B.in
Press any key to continue . .
display takes: 4ms
Press any key to continue . .
display takes: 4ms
[c1 >= 0 && c2 >= 0) {
    Press any key to continue . .
    / (c1 - c2) * (p - s));
t2in

```

The screenshot shows the Microsoft Visual Studio Code interface with the following details:

- Title Bar:** DrawKit.hpp - 3D-Computer-Graphics-Lab3 - Visual Studio Code
- Left Panel (Code Editor):** Shows `Lab3.cpp` with code for clipping polygons. It includes functions like `clip_plane()`, `clip_polygons()`, and `clip_polygons_inplace()`. The code handles perspective division and uses `std::lock_guard` for thread safety.
- Right Panel (Terminal):** Shows the output of the program. It starts with the command `g++ -o Lab3B Lab3.cpp`, followed by several frames of output from the program itself:
  - Frame 1: Press any key to continue . . .
  - Frame 2: display takes: 1ms
  - Frame 3: Press any key to continue . . .
  - Frame 4: display takes: 1ms
  - Frame 5: Press any key to continue . . .
  - Frame 6: Press any key to continue . . .
  - Frame 7: Press any key to continue . . .
  - Frame 8: display takes: 22ms
  - Frame 9: Press any key to continue . . .
  - Frame 10: display takes: 21ms
  - Frame 11: Press any key to continue . . .

2.

Lab3C.in

1.

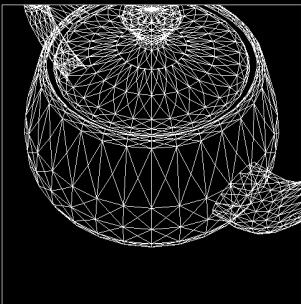


```

1 File Edit Selection View Go Debug Terminal Help DrawKit.hpp - 3D-Computer-Graphics-Lab3 - Visual Studio Code
2 h- DrawKit.hpp x h- MatrixWith.hpp C- Lab3_105502042.cpp h- Object.hpp h- Observer.hpp h- viewport.hpp
3 2019CG_Lab3_105502042\2019CG_Lab3_105502042> h- DrawKit.hpp
4 #include <iostream>
5 #include <vector>
6 #include <cmath>
7 #include <algorithm>
8 #include <functional>
9 #include <array>
10 #include <map>
11 #include <list>
12 #include <stack>
13 #include <queue>
14 #include <set>
15 #include <random>
16 #include <thread>
17 #include <mutex>
18 #include <atomic>
19 #include <condition_variable>
20 #include <chrono>
21 #include <future>
22 #include <atomic>
23 #include <functional>
24 #include <vector>
25 #include <array>
26 #include <cmath>
27 #include <vector>
28 #include <functional>
29 #include <array>
30 #include <vector>
31 #include <array>
32 #include <functional>
33 #include <array>
34 #include <vector>
35 #include <array>
36 #include <functional>
37 #include <array>
38 #include <vector>
39 #include <array>
40 #include <functional>
41 #include <array>
42 #include <vector>
43 #include <array>
44 #include <functional>
45 #include <array>
46 #include <vector>
47 #include <array>
48 #include <functional>
49 #include <array>
50 #include <vector>
51 #include <array>
52 #include <functional>
53 #include <array>
54 #include <vector>
55 #include <array>
56 #include <functional>
57 #include <array>
58 #include <vector>
59 #include <array>
60 #include <functional>
61 #include <array>
62 #include <vector>
63 #include <array>
64 #include <functional>
65 #include <array>
66 #include <vector>
67 #include <array>
68 #include <functional>
69 #include <array>
70 #include <vector>
71 #include <array>
72 #include <functional>
73 #include <array>
74 #include <vector>
75 #include <array>
76 #include <functional>
77 #include <array>
78 #include <vector>
79 #include <array>
80 #include <functional>
81 #include <array>
82 #include <vector>
83 #include <array>
84 #include <functional>
85 #include <array>
86 #include <vector>
87 #include <array>
88 #include <functional>
89 #include <array>
90 #include <vector>
91 #include <array>
92 #include <functional>
93 #include <array>
94 #include <vector>
95 #include <array>
96 #include <functional>
97 #include <array>
98 #include <vector>
99 #include <array>
100 #include <functional>
101 if (polygon.size()) {
102     for (auto& a : polygon) // perspective division
103         if (a[3] != 1)
104             a = Vector4{ {a[0] / a[3], a[1] / a[3], a[2] / a[3], 1.0} };
105     std::lock_guard l{ lock_clipped_polys };
106     clipped_polys.push_back(std::move(polygon));
107 }
108 }
109 return clipped_polys;

```

2.



```

1 File Edit Selection View Go Debug Terminal Help DrawKit.hpp - 3D-Computer-Graphics-Lab3 - Visual Studio Code
2 h- DrawKit.hpp x h- MatrixWith.hpp C- Lab3_105502042.cpp h- Object.hpp h- Observer.hpp h- viewport.hpp
3 2019CG_Lab3_105502042\2019CG_Lab3_105502042> h- DrawKit.hpp
4 #include <iostream>
5 #include <vector>
6 #include <cmath>
7 #include <algorithm>
8 #include <functional>
9 #include <array>
10 #include <map>
11 #include <list>
12 #include <stack>
13 #include <queue>
14 #include <set>
15 #include <random>
16 #include <thread>
17 #include <mutex>
18 #include <atomic>
19 #include <condition_variable>
20 #include <chrono>
21 #include <future>
22 #include <atomic>
23 #include <functional>
24 #include <vector>
25 #include <array>
26 #include <cmath>
27 #include <vector>
28 #include <functional>
29 #include <array>
30 #include <vector>
31 #include <array>
32 #include <functional>
33 #include <array>
34 #include <vector>
35 #include <array>
36 #include <functional>
37 #include <array>
38 #include <vector>
39 #include <array>
40 #include <functional>
41 #include <array>
42 #include <vector>
43 #include <array>
44 #include <functional>
45 #include <array>
46 #include <vector>
47 #include <array>
48 #include <functional>
49 #include <array>
50 #include <vector>
51 #include <array>
52 #include <functional>
53 #include <array>
54 #include <vector>
55 #include <array>
56 #include <functional>
57 #include <array>
58 #include <vector>
59 #include <array>
60 #include <functional>
61 #include <array>
62 #include <vector>
63 #include <array>
64 #include <functional>
65 #include <array>
66 #include <vector>
67 #include <array>
68 #include <functional>
69 #include <array>
70 #include <vector>
71 #include <array>
72 #include <functional>
73 #include <array>
74 #include <vector>
75 #include <array>
76 #include <functional>
77 #include <array>
78 #include <vector>
79 #include <array>
80 #include <functional>
81 #include <array>
82 #include <vector>
83 #include <array>
84 #include <functional>
85 #include <array>
86 #include <vector>
87 #include <array>
88 #include <functional>
89 #include <array>
90 #include <vector>
91 #include <array>
92 #include <functional>
93 #include <array>
94 #include <vector>
95 #include <array>
96 #include <functional>
97 #include <array>
98 #include <vector>
99 #include <array>
100 #include <functional>
101 if (polygon.size()) {
102     for (auto& a : polygon) // perspective division
103         if (a[3] != 1)
104             a = Vector4{ {a[0] / a[3], a[1] / a[3], a[2] / a[3], 1.0} };
105     std::lock_guard l{ lock_clipped_polys };
106     clipped_polys.push_back(std::move(polygon));
107 }
108 }
109 return clipped_polys;

```

## Lab3D.in

File Edit Selection View Go Debug Terminal Help DrawKit.hpp - 3D-Computer-Graphics\Lab3 - Visual Studio Code

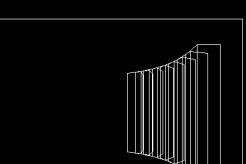
2019CG\_Lab3\_105502042 > 2019CG\_Lab3\_105502042.cpp h+ Matrix.hpp h+ Lab3.hpp h+ Object.hpp h+ Observer.hpp h+ Viewport.hpp

inline auto project\_<clip\_plane> const polygons<> polygons, const matrix<> &matrix) {

inst Vector<4> v) { return v[3] + v[0]; }, // w = x, w + x  
 inst Vector<4> v) { return v[3] + v[1]; }, // w = y, w + y  
 inst Vector<4> v) { return v[2]; })); // w = z, w

}, [&](Polygon\_u<4> polygon) {

Lab3Window



Press any key to continue . . .  
 display takes: 280ms  
 Press any key to continue . . .  
 display takes: 257ms  
 Press any key to continue . . .

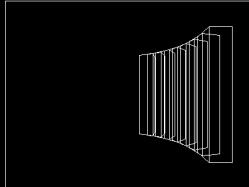
D:\3D-Computer-Graphics\Lab3\2019CG\_Lab3\_105502042\Debug>2019CG\_Lab3\_105502042.e  
xe Lab3D.in  
Press any key to continue . . .  
display takes: 1ms  
Press any key to continue . . .  
display takes: 1ms  
Press any key to continue . . .  
display takes: 1ms  
Press any key to continue . . .  
display takes: 2ms  
Press any key to continue . . .  
display takes: 2ms  
Press any key to continue . . .  
display takes: 3ms  
Press any key to continue . . .  
display takes: 4ms  
Press any key to continue . . .  
display takes: 4ms  
Press any key to continue . . .  
display takes: 5ms  
Press any key to continue . . .

relay.clear();

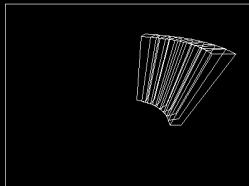
};

if (polygon.size()) {  
 for (auto& a : polygon) // perspective division  
 if (a[3] != 1)  
 a = Vector<4>{a[0] / a[3], a[1] / a[3], a[2] / a[3], 1.0};  
 std::lock\_guard l{lock\_clipped\_polys};  
 clipped\_pots.push\_back(std::move(polygon));  
}

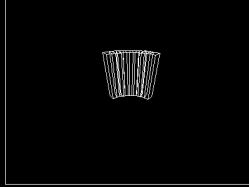
});  
return clipped\_pots;



1.



2.



# Lab3E.in

```
File Edit Selection View Go Debug Terminal Help
D:\3D-Computer-Graphics-Lab3\Visual Studio Code
Lab3E.in -> 2019CG_Lab3_105502042\Debug>2019CG_Lab3_105502042.exe
Lab3E.in
1. 
2. 

3. 
4. 
```

```
1. 
2. 
```

```
3. 
```

```
4. 
```

## 影片

[https://youtu.be/FgfSL\\_YjRl8](https://youtu.be/FgfSL_YjRl8)

## 6 Source files

1. **Lab3\_105502042.cpp** ➔ 處理讀檔和每個指令。
2. **Viewport.hpp** ➔ 定義存放viewport資訊的object，包含了aspect ratio。
3. **Object.hpp** ➔ 定義了每個 .asc 檔讀進來的資料結構，也提供了幾個member function方便讀入 .asc 檔使用。在 `set_vertex` 中每個頂點在讀入時即會左乘當時的TM。`to_polygons()` 則是會回傳多個多邊形 (of type `Polygons<4>` , which is a `std::vector< of Polygon_u<4>` , which is yet another `std::vector< of Vector<4>` , which is a `std::array<double, 4>` )。
4. **Observer.hpp** ➔ 讀到observer指令時會instantiate一個observer物件，提供了一個member function回傳  $projection matrix \times eye matrix$  的 $4 \times 4$ 矩陣。

5. **DrawKit.hpp** ➡ 定義了畫線、畫多邊形、畫多邊形們和 `project_clip_pd` 這幾個函式。
6. **MatrixKit.hpp** ➡ 定義了各種和矩陣有關的運算，包含但不限於矩陣乘法、內積、外積、矩陣和向量乘法。有個對多邊形們的每個頂點左乘矩陣的函式也運用了上述的 `std::execution::par_unseq` execution policy。

footnote: [A Tour of C++ \(Second edition\)](#) is a concise, insightful, well-written book on C++17, authored by the creator of C++, Bjarne Stroustrup. A lot of the language constructs that I use come from this book.

## More on the rendering pipeline

### Indside of `project_clip_pd`

這個函式把多邊形們從world space轉到projection space (line 80)，再clipping (lines 82 to 90)，再做perspective divide (lines 102 to 104)，然後回傳處理好的多邊形們給 `process_display` 留待nobackfaces處理。

這個函式利用了 `std::for_each` (line 79)這個higher-order function，我把我要做的事情寫成一個lambda expression (lines 80 to 108)當成first-class function成為他的參數，參數部分再多加入 `std::execution::par_unseq` (parallel and unsequenced，但MSVC似乎只實作了parallel，也就是multi-threading的部分)，使得整件事可以平行處理，但如此便要加lock 🔒 (lines 77 and 105)否則會crash。

最外層的 `std::for_each` 會iterate over每個polygon，內層處理clipping的部分 `for (const auto& c : codes)` 則是iterate through 6個functors (lines 73 to 75)對應到老師筆記的6個不等式，比如說 `c(S)` 可能就會回傳 $S$ 這個點的 $S_w - S_x$ 。如此一來便不用寫6個長得差不多的for迴圈了👉。

```

70
71 inline auto project_clip_pd(const Polygons<4>& polygons, const Matrix<4>& pmXem) {
72     using Codes = std::array<const std::function<double(const Vector<4>&), &>; // 6 planes
73     const Codes codes{{[] (const Vector<4>& v) { return v[3] - v[0]; }, [] (const Vector<4>& v) { return v[3] + v[0]; }, // w - x, w + x
74                         [] (const Vector<4>& v) { return v[3] - v[1]; }, [] (const Vector<4>& v) { return v[3] + v[1]; }, // w - y, w + y
75                         [] (const Vector<4>& v) { return v[3] - v[2]; }, [] (const Vector<4>& v) { return v[2]; }}}; // w - z, w
76     Polygons<4> clipped_polys;
77     std::mutex lock_clipped_polys;
78
79     std::for_each(std::execution::par_unseq, polygons.begin(), polygons.end(), [&](Polygon_u<4> polygon) {
80         polygon = pmXem * polygon; // project a polygon to projection space
81
82         Polygon_u<4> relay;
83         for (const auto& c : codes) { // clip against all six planes
84             const auto sz = polygon.size();
85             for (size_t i = 0; i < sz; ++i) {
86                 const auto& s = polygon[i];
87                 const auto& p = polygon[(i + 1) % sz];
88                 if (const double c1c(s), c2c(p)); (c1 >= 0 && c2 >= 0)) { // in2in
89                     relay.push_back(p);
90                 } else if (const auto new_s = (s + c1 / (c1 - c2) * (p - s)); (c1 >= 0 && c2 < 0)) { // in2out
91                     relay.push_back(new_s);
92                 } else if (c1 < 0 && c2 >= 0) { // out2in
93                     relay.push_back(new_s);
94                     relay.push_back(p);
95                 }
96             }
97             polygon = relay;
98             relay.clear();
99         }
100
101         if (polygon.size()) {
102             for (auto& a : polygon) // perspective division
103                 if (a[3] != 1)
104                     a = Vector<4>{{a[0] / a[3], a[1] / a[3], a[2] / a[3], 1.0}};
105             std::lock_guard l{lock_clipped_polys};
106             clipped_polys.push_back(std::move(polygon));
107         }
108     });
109 }
110 }
```

### Call site of `project_clip_pd` (from world space to screen space)

lines 75–80 ➡ 把所有讀進來過的objects都轉成多邊形們等下一次處理。

line 82 ➡ 所有多邊形從world space到image space

lines 84–88 ➡ nobackfaces處理，移除所有法向量和Z方向夾角  $\theta \leq 90^\circ$  的多邊形

lines 90–93 ➡ 清除畫面並畫出viewport邊界，其中 `vp.get_borders()` 回傳一個tuple可以透過structured binding 賦值給左側的四個變數

line 94 ➡ 轉到screen space後畫在螢幕上，其中最右邊的 `*` 平行地對多邊形們(`ps`)左乘左側的矩陣

參數中的 `pmXem` 為 *projection matrix × eye matrix* =

$$\begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & A_R & 0 & 0 \\ 0 & 0 & \frac{y}{y-H} \tan \theta & \frac{H \times y}{H-y} \tan \theta \\ 0 & 0 & \tan \theta & 0 \end{pmatrix} \times \text{Rotate}_{(-Tilt)} \times \text{Mirror}_{(x)} \times \text{GRM} \times \text{Translate}_{(-E_x, -E_y, -E_z)}$$

```
72 auto process_display(const Viewport& vp, const std::vector<Object>& objects, const Matrix<4>& pmXem) {
73     auto t0 = std::chrono::high_resolution_clock::now();
74     // dump all faces of all objects to Polygons<4>
75     Polygons<4> ps;
76     for (const auto& obj : objects) {
77         const auto& a = obj.to_polygons();
78         ps.insert(ps.end(), a.begin(), a.end());
79     }
80
81     ps = project_clip_pd(ps, pmXem); // performs projection, clipping, and perspective division in parallel
82
83     // nobackfaces
84     if (nobackfaces)
85         ps.erase(std::remove_if(std::execution::par_unseq, ps.begin(), ps.end(),
86                                [] (const auto& a) { return cross(a[1] - a[0], a[2] - a[1])[2] >= 0; }),
87                                ps.end());
88
89     glClearColor(0.0, 0.0, 0.0, 0.0);
90     glClear(GL_COLOR_BUFFER_BIT);
91     const auto [vx1, vxr, vyb, vyt] = vp.get_borders();
92     draw_polygon(Polygon<4>{{{vx1, vyb}, {vxr, vyb}, {vxr, vyt}, {vx1, vyt}}}); // this function will do the flushing
93     draw_polygons(translation_m(vx1, vyb) * scaling_m((vxr - vx1) / 2.0, (vyt - vyb) / 2.0) * translation_m(1.0, 1.0) * ps);
94
95     auto t1 = std::chrono::high_resolution_clock::now();
96     std::cout << "display takes: " << std::chrono::duration_cast<std::chrono::milliseconds>(t1 - t0).count() << "ms\n";
97     system("pause");
98 }
99
100
```

## Requirement

1. the latest Visual Studio 2017
2. the latest Windows SDK
3. C++17  (structured binding, `std::string_view`, parallel algorithms)

## .clang-format settings

```
BasedOnStyle: LLVM
PointerAlignment: Left
ColumnLimit: 0
SpaceAfterTemplateKeyword: false
```

## .editorconfig settings

```
root = true

[*]
indent_style = space
indent_size = 2
end_of_line = lf
charset = utf-8
trim_trailing whitespace = true
insert_final_newline = true

[*.md]
trim_trailing whitespace = false
```