資工系 100062236 林修安

多媒體技術概論 HW4 Report

Part 1

圖一為 Cubic Bezier Curve 畫出來的兔子圖，其中紅色圓圈為 Points.txt 中的每個點。

將 0~1 分割成 100 個區間當作 t，組成矩陣 T

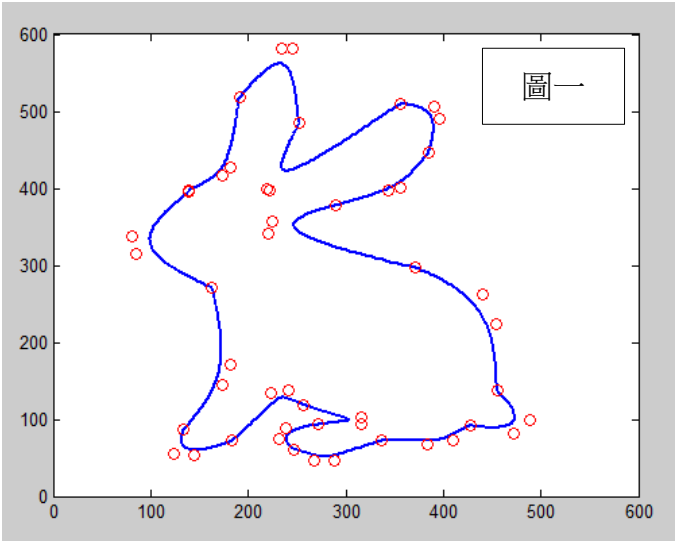
每四個控制點，算出 100 個 Bezier Curve 的點，並將所有的點串連在一起作圖。

```
P = textread('Points.txt');
P(end+1, :) = P(1, :);
M = [-1 3 -3 1; 3 -6 3 0; -3 3 0 0 ; 1 0 0 0 ];
n = 100;
t = linspace(0, 1, n)';
T = [t.^3 t.^2 t ones(n, 1)];
XY = [];

for i = 1:3:50
    if i+3<=50
        XY = cat(1, XY, T * M * P(i:i+3, :));
    end
end
```

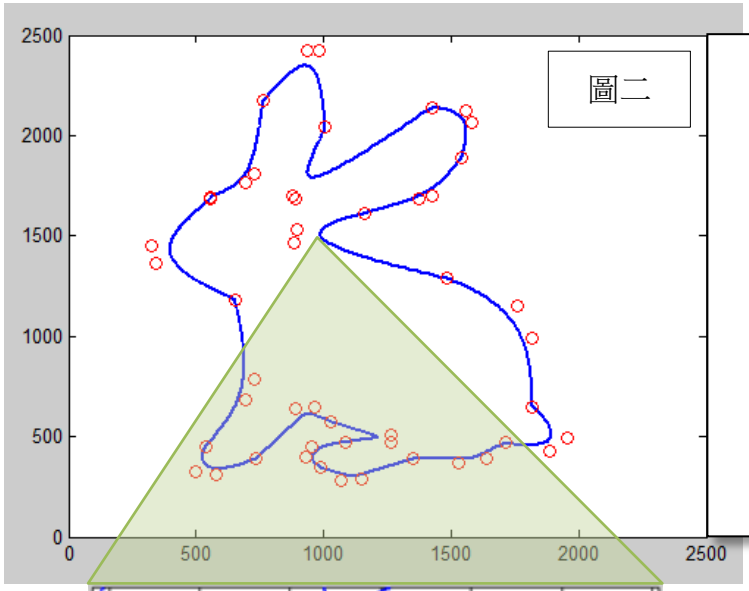
因為原本的圖畫出來是上下顛倒的，所以我把所有的 Y 取負號加上 600。

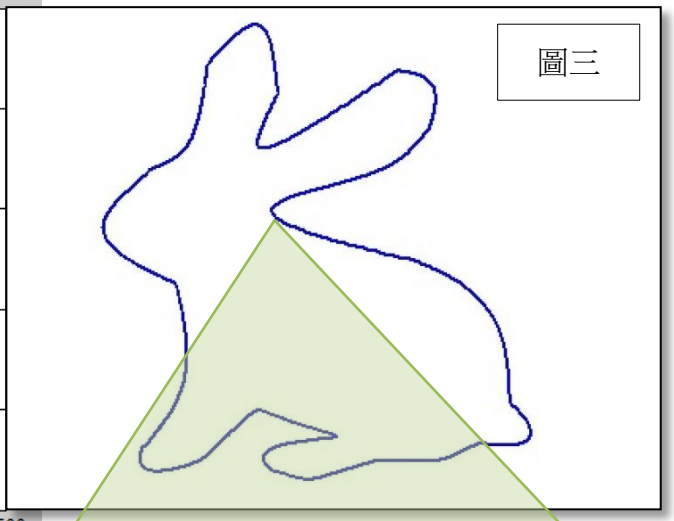
```
XY(end, :) = P(1, :);
figure, plot(XY(:, 1), -XY(:, 2)+600, 'b-', 'lineWidth', 2); hold on;
plot(P(:, 1), -P(:, 2)+600, 'ro');
axis ([0 600 0 600]);
```

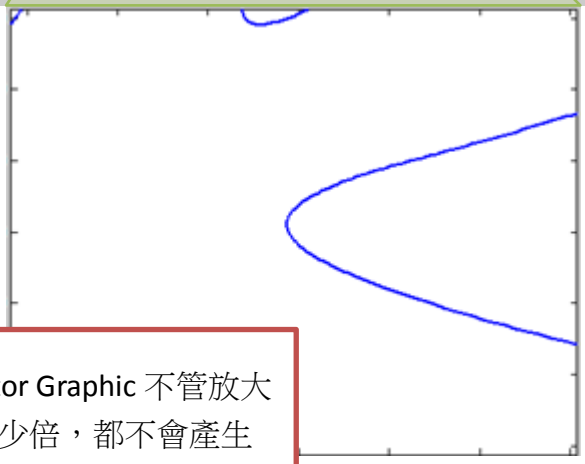


圖二則為 Bezier Curve 放大四倍的圖，將原本的點全部乘以四，再依照上述方法作出放大的圖。

圖三為用 Nearest Neighbor 將圖一放大四倍的結果。

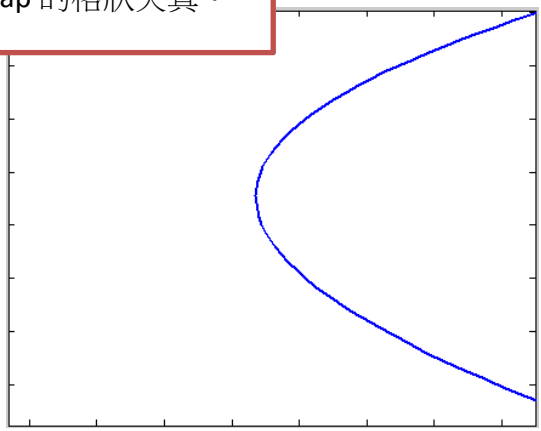


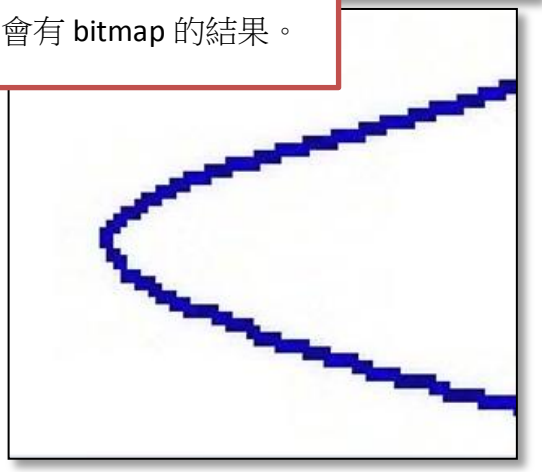




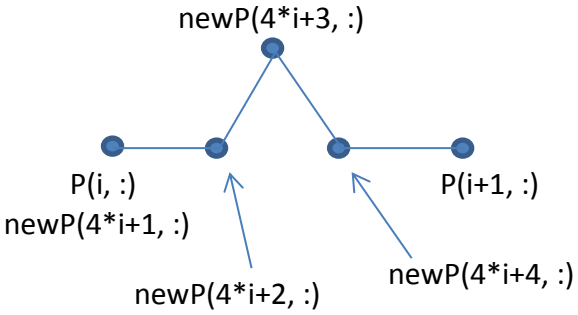
Vector Graphic 不管放大多少倍，都不會產生 bitmap 的格狀失真。

用 nearest neightbor method 放大 JPG 圖檔則會有 bitmap 的結果。





Part 2



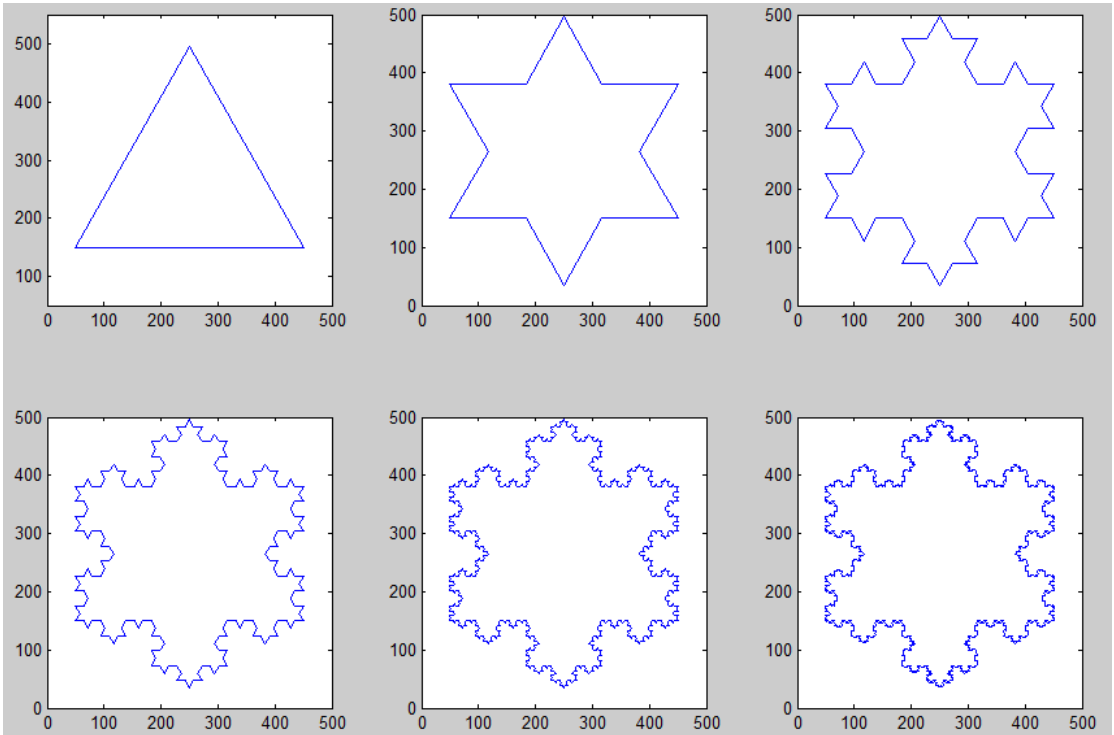
對每組 P(i, :)及 P(i+1, :)做內分點，
算出 newP(4*i+2, :)及 newP(4*i+4, :)；

算出從 newP(4*i+2, :)到 newP(4*i+3, :)
的長度(邊長三分之一)以及需要旋轉的角度θ，

$tan\theta = \frac{P(i+1)_y - P(i)_y}{P(i+1)_x - P(i)_x}$ ，即可找出 newP(4*i+3, :)。

如此重複步驟即可畫出 Koch's snowflake。

```
P = [50 150;450 150;50+400*cos(pi/3) 150+400*sin(pi/3); 50 150];
subplot(2, 3, 1); plot(P(:,1), P(:,2));
hold on; axis([0 500 50 550]);
for iter = 1:5
    newP = zeros(size(P,1)*4+1, 2);
    for i=1:size(P,1)-1
        newP(4*i+1,:) = P(i,:);
        newP(4*i+2,:) = (2*P(i,:) + P(i+1,:))/3;
        vector = P(i+1,:)-P(i,:);
        theta = atan2(vector(2), vector(1));
        vectorLen = sqrt(sum(vector.^2));
        newP(4*i+3,:) = newP(4*i+2,:) + (vectorLen/3)*[cos(theta-pi/3), sin(theta-pi/3)];
        newP(4*i+4,:) = (P(i,:) + 2*P(i+1,:))/3;
    end
    newP(4*size(P,1)+1,:) = P(size(P,1),:);
    nonzero = newP(:, 1)>0;
    P = cat(3, newP(nonzero, 1), newP(nonzero, 2));
    subplot(2, 3, iter+1); plot(P(:,1), P(:,2));
end
```



Execution

Part1:

hw4_1 畫出原圖以及用 vector 放大後的結果。

nearestNeighbor 畫出用 nearest neighbor 放大後的結果。

Part2:

hw4_2 畫出 Koch's snowflake (0~5 stages)