

## Tache 8 Partie 1

### Courbe\_hilbert

#### Courbe\_hilbert\_7

Le test se termine sans aucun souci avec un temps user

=====

./test\_simplification-courbe\_hilbert\_7 39.09s user 24.90s system 89% cpu 1:11.48 total

#### Courbe\_hilbert\_8

Le test compte le nombre des contours mais la simplification ne se termine pas (killed)

=====

zsh: killed ./test\_simplification-courbe\_hilbert\_8

./test\_simplification-courbe\_hilbert\_8 44.85s user 28.77s system 82% cpu 1:28.89 total

Si on prend en compte la taille de l'image qui est 511x511, avec un nombre de segment total de 262144, pour d=0 et vu qu'il y a un seul contour, la fonction récursive de simplification est trop surchargée. En fait, la taille des listes chaines continue d'augmenter jusque qu'on arrive un moment pic de mémoire utilisé – définie par le system d'exploitation.

L'ordinateur prendre la décision d'arrêter le processus pour protéger son système central et les autres applications qui ont besoin de la RAM, par rapport au stack existant.

En architecture Mac, il n'y a pas de limite par rapport aux temps accumulé en CPU. Pourtant, il y a un mécanisme sur la machine qui dit que si une seule opération dépasse la limite de 80GB de mémoire sur une seule opération, on va faire un kill de l'opération cela pourrait être une opération qui voudrait endommager le système d'exploitation. On a observé que dans une période de 10 seconds, le programme a consommé 20GB de RAM qui était sauvegardé dans une partie du disque dure pour donner de la place dans la mémoire RAM pour le reste des opérations.

On peut conclure que le fait que le programme est arrêté, ne dépend pas par la taille de l'image, mais il dépend de nombre des segments à simplifier, qui est analogue au mémoire utilisé (par les listes chaines et les listes chaines des contours simplifiés).

#### Courbe\_hilbert\_9

On a observé le même comportement que pour Courbe\_hilbert\_8. Ici on a :

zsh: killed ./test\_simplification-courbe\_hilbert\_9

./test\_simplification-courbe\_hilbert\_9 46.02s user 29.62s system 90% cpu 1:23.93 total

#### Courbe\_hilbert\_10

On a observé le même comportement que pour Courbe\_hilbert\_8. Ici on a :

zsh: killed ./test\_simplification-courbe\_hilbert\_10

./test\_simplification-courbe\_hilbert\_10 48.87s user 29.84s system 92% cpu 1:25.13 total

## Zebres

### Zebres-2000x1500

Le test se termine sans aucun souci avec un temps user

```
=====
./test_simplification-zebres-2000x1500 36.70s user 20.66s system 86% cpu 1:06.20 total
```

### Zebres-1000x0750

Le test se termine sans aucun souci avec un temps user

```
=====
./test_simplification-zebres-1000x0750 6.25s user 3.72s system 87% cpu 11.434 total
```

### Zebres-3000x2250

Le test compte le nombre des contours mais la simplification ne se termine pas (killed)

```
=====
zsh: killed ./test_simplification-zebres-3000x2250
./test_simplification-zebres-3000x2250 57.92s user 28.64s system 89% cpu 1:36.78 total
```

On a observe que la mémoire qu'on a besoin pour tous les operations de simplification de cette image est un peu pres 63GB. Mais meme si on n'a pas arrive au pic de 80GB de protection d'ordinateur, comme explique pour l'image Courbe\_hilbert\_8.

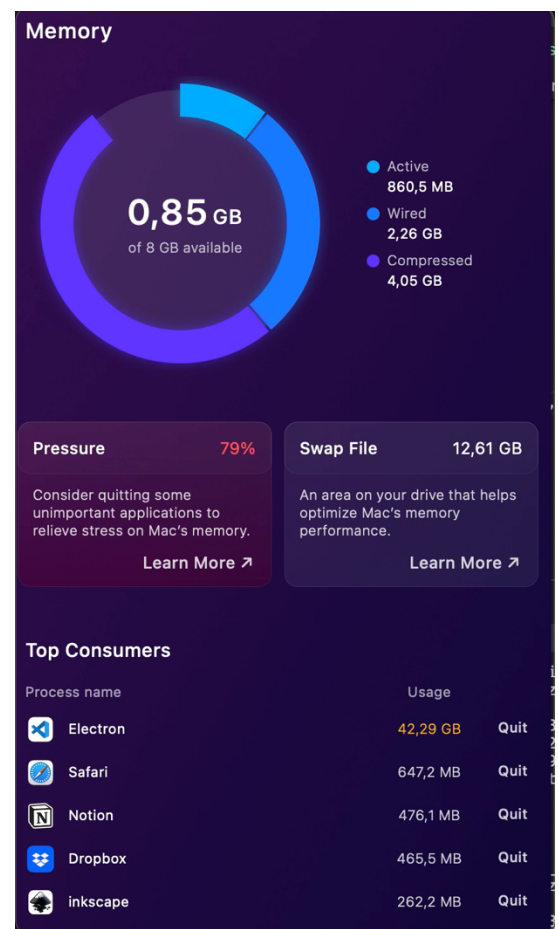
Dans le cas de cette image, le stress mémoire a arrivé à 100% et c'est pourquoi le programme était arrete. En fait, plus il y a de "stress", moins la mémoire traite efficacement les données.

### Zebres-4000x3000

Le test compte le nombre des contours mais la simplification ne se termine pas (killed)

On a observé le même comportement que pour Courbe\_hilbert\_8. Ici on a :

```
=====
zsh: killed ./test_simplification-zebres-4000x3000
./test_simplification-zebres-4000x3000 66.76s user
30.00s system 84% cpu 1:54.34 total
```



## Tache 8 Partie 2

### Résultats des méthodes de simplification

Cnt -> Contours

Seg -> Segments

Bezier2	Original	D=0	D=0.5	D=1	D=2	D=4	D=8	D=16
Asterix3	Cnt: 32 Seg:12926	8343	4968	966	371	253	183	119
Lettre-L-cursive	Cnt: 3 Seg: 4228	2777	1717	255	46	34	29	21
ColombesDeLaPaix	Cnt: 106 Seg:21764	13989	8440	1599	717	505	332	229

\* Les nombres corresponds au nombre des courbes de Bézier de degré 2

Bezier3	Original	D=0	D=0.5	D=1	D=2	D=4	D=8	D=16
Asterix3	Cnt: 32 Seg:12926	7603	3470	648	298	210	155	85
Lettre-L-cursive	Cnt: 3 Seg: 4228	2460	1215	157	40	28	25	16
ColombesDeLaPaix	Cnt: 106 Seg:21764	12861	5957	1155	563	385	259	180

\* Les nombres corresponds au nombre des courbes de Bézier de degré 3

Simplification	Original	D=0	D=0.5	D=1	D=2	D=4	D=8	D=16
Asterix3	Cnt: 32 Seg:12926	7698	6172	1314	635	394	264	153
Lettre-L-cursive	Cnt: 3 Seg: 4228	2380	1904	271	103	74	51	36
ColombesDeLaPaix	Cnt: 106 Seg:21764	13338	10474	2452	1133	769	497	323

\* Les nombres corresponds au nombre des segments après l'application de l'algorithme de simplification par segments

### Commentaires sur les méthodes de simplification

On peut remarquer que la méthode de simplification par courbes de Bézier de degré 3 nous donne comme résultat le plus petit nombre des courbes, par rapport la méthode de courbe de Bézier de degré 2 et la méthode de simplification par segments.

Effectivement, si on passe directement par courbes de Bézier de degré 3 et pas par la méthode de courbes de Bézier de degré 2, on n'a pas besoin de transformer les courbes de degré 2 en courbes de degré 3 pour la construction du postscript. De plus, vu que l'approximation par courbe de Bézier de degré 3 donne 4 points à chaque itération, en comparaison avec l'approximation de courbes de Bézier de degré 2, on lit le total des points plus efficacement, tout en les simplifiant.

Comme attendue, la méthode de simplification des segments est la moins efficace. En fait, cette méthode nous permet de simplifier des contours mais il ne permet pas de minimiser au maximum les points qu'on a besoin tout en maintenant l'image « lisible » et sans beaucoup des modifications qui impactent sa qualité après la simplification (voir section 'Commentaires par rapport la qualité des images'). C'est pourquoi, si on compare les résultats avec la méthode de courbe de Bézier de degré 2, on constate une diminution de 45 % des segments/courbes.

### **Commentaires par rapport la qualité des images**

*Ceux commentaires appliquent aux visualisations des simplifications que vous pouvez trouver à la section correspondante ci-dessous.*

On constate pour une valeur de distance de seuil assez grand (ex : 16), la seule méthode qui est capable de maintenir le maximum des détails de l'image original est celle de la simplification par courbe de Bézier de degré 3.

Si on faisait une comparaison entre la méthode de courbe de Bézier de degré 2 et de courbe de Bézier de degré 3, on peut constater que pour une distance seuil moyenne (ex : 4), on ne perde pas un grand nombre des informations essentiels, en comparaison de l'image original.

Par contre, on constate très rapidement que la méthode par simplification des segments commence à modifier la nature de l'image original à partir d'une distance seuil moyenne. Pour des distances seuils assez grandes, on constate qu'il est impossible de représenter des courbes au niveau des pixels et les points restants après la simplification. Ça s'explique par la manière de création du fichier EPS et l'algorithme de Douglas qu'on utilise pour minimiser les points qu'on a besoin pour représenter l'image en fonctionne de la distance seuil donne en argument.

En même temps, pour des distances seuils assez grandes, la méthode de courbes de Bézier de degré 2, même s'il est capable de représenter de courbes (manière de création du fichier EPS), elle est aussi incapable de maintenir toutes les informations de l'image originale. Ça s'explique par le fait qu'on a besoin de transformer les courbes de Bézier degré 2 aux courbes de Bézier de degré 3, et si les points construit qui complètent la base de la courbe de degré 3 ne font partie du contour original, ceux points sont considère comme "artificielles" qui nous amené à perdre la précision de la représentation de ceux points.

On peut conclure que pour des distances seuil assez petit (ex : 0, 0.5, 1 ou 2), l'efficacité des trois méthodes, par rapport leur efficacité de maintenir la qualité de l'image, est un peu près la même.







La seule méthode qui est capable de maintenir tous les détails pour n'importe quelle distance seuil est celle de courbe de Bézier de degré 3, pour des raisons qu'on explique en partie 'Commentaires sur les méthodes de simplification'.







Visualisations des simplifications

On a inclus la totalité des images simplifie pour tous les trois images testé et comparé.

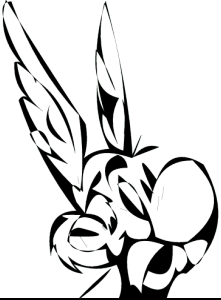




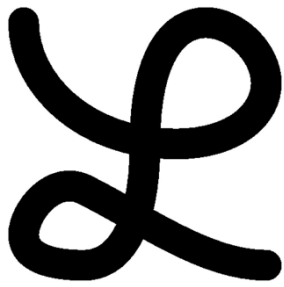
Asterix3	D=0	D=0.5	...
Bezier2			...
Bezier3			...
Simplification			...

	D=1	D=2	...
Bezier2			...
Bezier3			...
Simplification			...

	D=4	D=8	...
Bezier2			...
Bezier3			...
Simplification			...

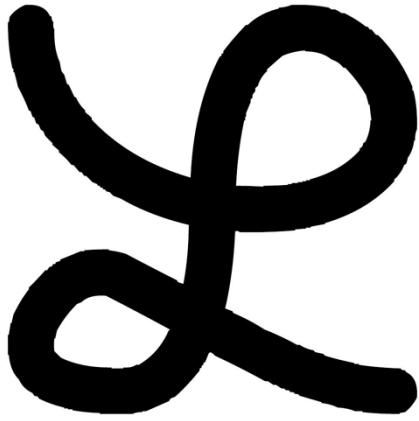
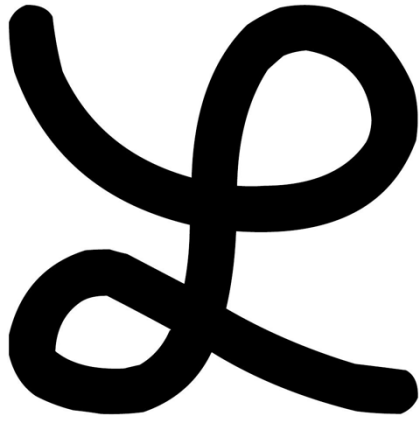
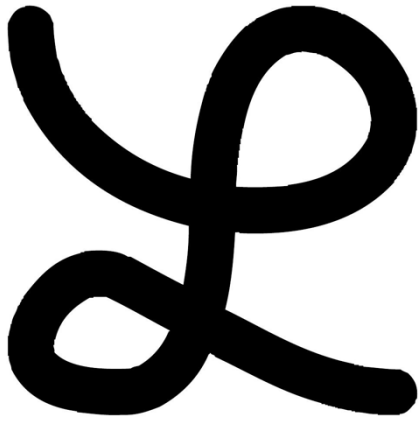
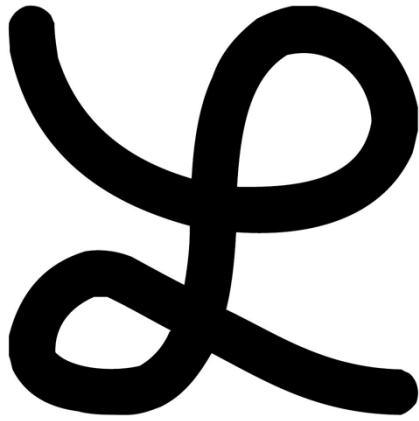
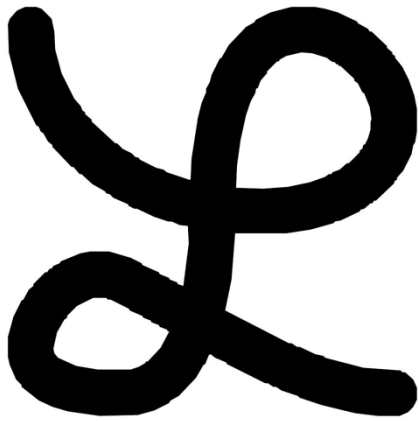
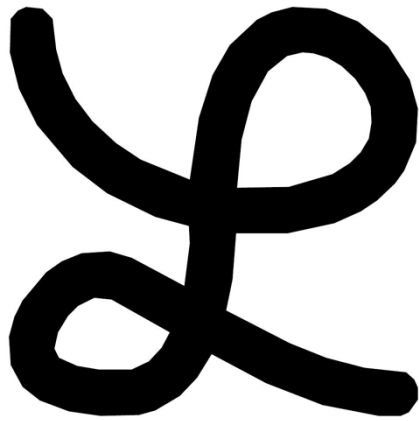
D=16

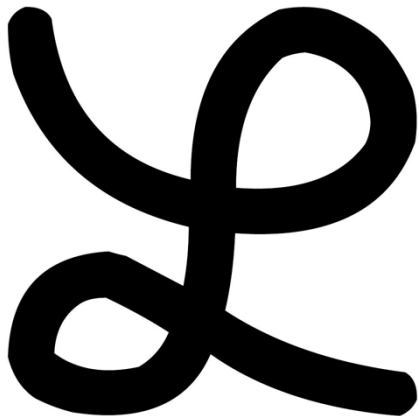
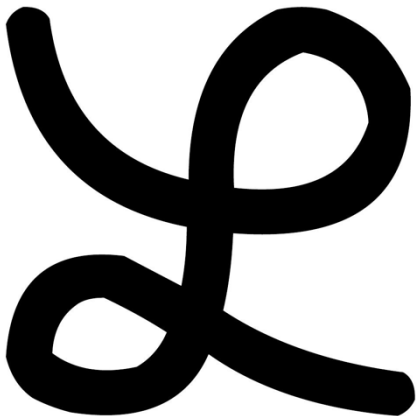
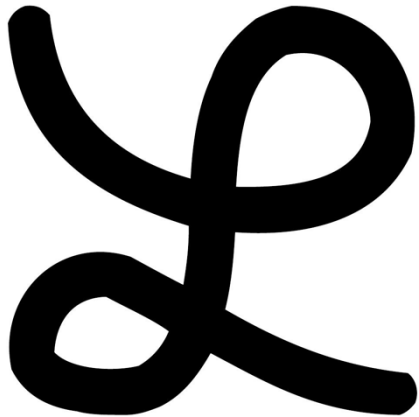
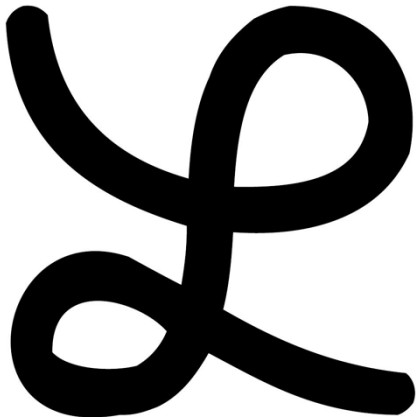
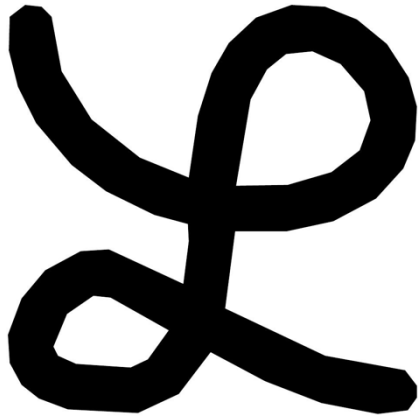

Bezier2	Bezier3	Simplification
		



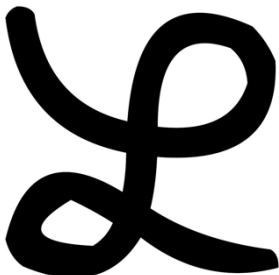


Lettre-L-c...	D=0	D=0.5	...
Bezier2			...
Bezier3			...
Simplification			...

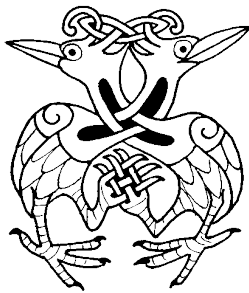








	D=1	D=2	...
Bezier2			...
Bezier3			...
Simplification			...

	D=4	D=8	...
Bezier2			...
Bezier3			...
Simplification			...







D=16

Bezier2	Bezier3	Simplification
		




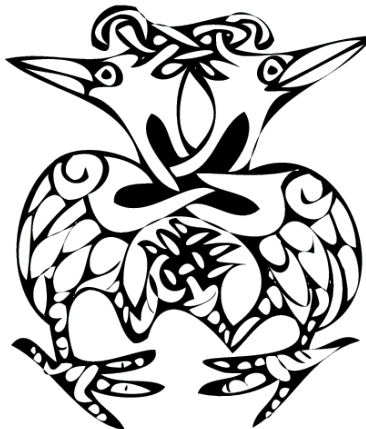

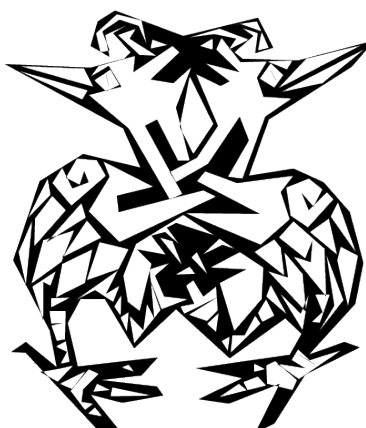


Colombes...	D=0	D=0.5	...
Bezier2			...
Bezier3			...
Simplification			...

	D=1	D=2	...
--	-----	-----	-----

Bezier2			...
Bezier3			...
Simplification			...

	D=4	D=8	...
--	-----	-----	-----

Bezier2			...
Bezier3			...
Simplification			...

D=16

Bezier2	Bezier3	Simplification
