

Partie pratique - Présentation

La partie pratique (T.P.) du projet logiciel est prévue sur dix semaines correspondant aux dix semaines civiles s04, s05, s06, s07, s08, s10, s12, s13, s14 et s15. Chaque semaine, un créneau de 3h est prévu, la première partie (1h30) est encadrée par votre enseignante ou enseignant de TP, la seconde partie (1h30) est en libre service.

Dans la mesure du possible, essayez de consacrer uniquement ce créneau de 3h à l'UE MAP401.

Documents à envoyer

Suivi de projet

Tout au long de ce projet, vous devrez remplir au fur et à mesure des semaines un document *tableur* nommé **SuiviProjet**. Ce fichier permettra de savoir où vous en êtes, aussi bien pour vous que pour l'enseignant.

Ce fichier disponible aux formats MSOffice-Excel (.xls) et LibreOffice-Calc (.ods) dans la section Documents du cours MAP401 sur le site Moodle de l'UGA (<https://cours.univ-grenoble-alpes.fr>)

Dans ce fichier, vous aurez à compléter chaque semaine les onglets **DiagrammeGanttReel** et **JournalDeBord** :

- dans l'onglet **DiagrammeGanttReel**, vous indiquerez pour chaque semaine la ou les tâches sur laquelle ou lesquelles vous avez travaillé.

Vous pourrez ainsi le comparer avec le **DiagrammeGanttPrevisionnel**.

- dans l'onglet **JournalDeBord**, vous indiquerez différentes informations sur les tâches sur lesquelles vous travaillez ou devriez travailler. Notamment par les actions en cours ou reportées, vous saurez d'une semaine sur l'autre quelles sont vos priorités.

Le tableau est composé de différentes colonnes :

1. *Réf* : cette colonne contient un numéro chronologique servant à référencer rapidement une ligne du tableau. Le numéro ne change pas pendant toute la durée de vie du document.
2. *Date* : cette colonne contient la date à laquelle un problème ou une information a été identifié. La date ne change pas pendant toute la durée de vie du document.
3. *Problème / Information* : cette colonne contient la description textuelle du problème ou de l'information.
4. *Action / Décision* : il s'agit de lister ici les actions ou les décisions engagées ou à engager dans le but de traiter le problème ou l'information correspondant.
Les actions engagées doivent être réalistes, révisables (il peut être nécessaire de les remettre en cause) et mesurables en termes d'estimation de coûts et de résultats.
5. *Date de réalisation prévue* : inscrire la date de réalisation prévue pour l'action considérée.
6. *Date de réalisation réelle* : inscrire la date réelle de réalisation de l'action considérée.
7. *Etat* : cette colonne permet d'indiquer dans quel "état" se trouve une action considérée.
Inscrire dans cette colonne : *en cours*, *en attente*, *reportée*, *annulée* ou *terminée* en fonction de l'état de l'action.

- l'onglet **JournalDeBord.Exemple** vous montre un exemple.

Lors de chaque semaine civile s04, s05, s06, s07, s08, s10, s12, s13, s14 et s15, à la fin de la séance de 3h de TP, le document **SuiviProjet** devra être complété et envoyé par e-mail à l'adresse de votre enseignant.e de TP.

Le sujet du message devra être [MAP401] - SP_{numero_semaine_civile} - noms_du_binome

Comptes rendus de tâche

Lors d'une séance de TP, quand vous avez terminé une tâche ou une partie d'une tâche, les codes sources C correspondants, fichier *Makefile*, et éventuellement d'autres fichiers annexes devront être mis à disposition de votre enseignant.e de TP, et éventuellement un compte-rendu au format PDF.

Il n'est pas demandé de terminer telle ou telle tâche lors d'une semaine particulière, l'important est de terminer correctement une tâche avant de passer à la suivante, avancez à votre rythme, en essayant cependant de respecter le diagramme de Gantt prévisionnel.

A la fin de chaque tâche (ou partie de tâche), envoyez un e-mail à l'adresse de votre enseignant.e de TP avec le compte-rendu demandé.

Le sujet du message devra être `[MAP401] - TacheX-Y - noms_du_binome`

Il faudra aussi donner avoir accès à vos codes sources :

- soit en indiquant le chemin complet sur le serveur turing où se trouvent vos codes sources.
- soit en joignant à votre message une archive compressée contenant un répertoire "nettoyé", c'est à dire un répertoire avec uniquement un fichier *Makefile*, les fichiers sources C (.c/.h), et éventuellement d'autres fichiers spécifiques (données ou résultats), mais sans fichier image PBM de grande taille, sans fichier objet (.o), et sans fichier exécutable.

Sites pédagogiques

Serveur turing (accès avec login et mot de passe AGALAN) :

Alias : `im2ag-turing.u-ga.fr` ou `im2ag-turing.univ-grenoble-alpes.fr`

Adresse IP : `152.77.81.50`

Site Moodle - UGA avec documents (accès avec login et mot de passe AGALAN) :

`https://cours.univ-grenoble-alpes.fr/`

puis choisir le cours MAP401

Conseils

Commencez par créer dans votre répertoire personnel, un repertoire nommé MAP401 dans lequel chaque sous-répertoire correspondra à chaque semaine civile, les répertoires devront être nommés S04 (pour la première semaine de TP), S05 (pour la deuxième semaine de TP), etc ...

Chaque semaine, si vous avez besoin de fichiers que vous avez créés lors des semaines précédentes, recopiez-les à partir de leur répertoire d'origine. A la fin d'une semaine, vous ne devrez plus modifier le contenu du répertoire correspondant, le document *SuiviProjet* vous permettra ultérieurement de reprendre ou corriger un fichier source en le recopiant.

et surtout, appropriiez-vous les documents et conseils qui vous sont donnés.

Images

Tout au long du projet, vous aurez à utiliser des fichiers images au format PBM texte.

Vous pourrez créer vos propres images, ou récupérer des images se trouvant dans le répertoire `/home/Public/401_MAP_Public/IMAGES_TESTS/` sur le serveur turing.

Ces images sont compressées au format `gzip`.

Pour décompresser une image nommée `nom.pbm.gz`, utilisez la commande `gunzip nom.pbm.gz`

Pour décompresser toutes les images, utilisez la commande `gunzip *.pbm.gz`

Ces images sont aussi disponibles sur le site de l'UE MAP401 de la plateforme Moodle-UGA dans Documents → images-tests.zip (archive compressée au format zip).

Tâche 1 - Image Bitmap

Récupérez sur les fichiers nécessaires à ce TP en copiant dans votre répertoire de travail, le répertoire suivant `/home/Public/401_MAP_Public/TACHE1/`

Ce répertoire est aussi disponible sur le site de l'UE MAP401 de la plateforme Moodle-UGA dans Documents → TACHE1

Il est fourni un fichier `Makefile` et un fichier `include types_macros.h` que vous pourrez utiliser et compléter au fur et à mesure du projet.

(A) Dans le fichier `image.c`, compléter les corps des fonctions `ecrire_image` et `negatif_image`, et ajouter éventuellement d'autres fonctions qui vous semblent utiles.

(B) Ecrivez un programme `test_image` afin de tester les différentes fonctions du module `image`, notamment les fonctions `lire_fichier_image`, `ecrire_image` et `negatif_image`.

(C) Afin de tester votre programme, vous utiliserez un fichier PBM texte que vous aurez créé (en plus du fichier `caractere2.pbm` déjà fourni).

IMPORTANT : pour les points (A) et (B), n'utilisez pas directement les champs du type `Image`, mais utilisez les fonctions déjà écrites pour créer, modifier ou accéder à une variable de type `Image`.

Compte-rendu

Par cette première tâche, le compte-rendu consiste uniquement au codes sources que vous aurez complété ou créé.

Lorsque vous avez terminé la tâche 1, envoyez par e-mail un message à votre enseignant.e de TP, soit en indiquant un chemin sur `turing`, soit en joignant une archive compressée (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE1 - noms_du_binome**

Tâche 2 - Géométrie 2D

A) Commencez l'écriture d'un module de calcul géométrique en dimension 2.

Ce module devra définir (au minimum) un type `Vecteur` et un type `Point`, chacun contenant deux coordonnées réelles de type `double`, ainsi que des opérations géométriques de base sur ces différents types.

IMPORTANT : vous devez implémenter au minimum les opérations géométriques présentées en cours : somme de deux points/vecteurs, création d'un vecteur à partir de deux points, multiplication des coordonnées d'un point/vecteur par un réel, produit scalaire de deux vecteurs, norme d'un vecteur, distance entre deux points.

B) Ecrivez un programme de test afin de tester les différentes fonctions de votre module. Vous recopierez et complétez le fichier `Makefile` de la tâche 1 en conséquence.

Compte-rendu

Lorsque que vous avez terminé les points A) et B), créez un document de type traitement de texte (Libre Office Writer, Word, LaTeX, ...) en y mettant :

- les programmes sources de votre module `geometrie2d`
(le fichier interface `.h`, le fichier implémentation `.c`, le fichier du programme principal de test)
- le résultat de l'exécution de votre programme.

et exportez ce document au format PDF.

Lorsque vous avez terminé la tâche 2, envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE2 - *noms_du_binome***

Tâche 3 - Extraction d'un contour externe

Récupérez sur les fichiers nécessaires à ce TP en copiant dans votre répertoire de travail, le répertoire suivant `/home/Public/401_MAP_Public/TACHE3/`.

Ce répertoire est aussi disponible sur le site de l'UE MAP401 de la plateforme Moodle-UGA dans Documents → TACHE3

Dans ce répertoire TACHE3 se trouvent différentes images PBM que vous utiliserez pour les tests, ainsi que le programme `exemple_sequence_points` correspondant à l'exemple du cours.

Cette tâche consiste en l'écriture d'un module de calcul d'un contour d'une image bitmap noir et blanc, et d'un programme pour le tester. Ce module devra utiliser le module de manipulation d'image de la tâche 1, et si vous le jugez nécessaire, le module de géométrie 2D de la tâche 2.

IMPORTANT : cette tâche est en deux parties, et devra faire l'objet de deux comptes-rendus distincts.

Partie 1 - écriture du contour à l'écran

Commencez l'écriture d'un module de calcul de contour afin de pouvoir à partir d'une image PBM, calculer et écrire *à la volée* à l'écran le contour externe correspondant : pour cela la fonction `memoriser_position` consistera à écrire la position à l'écran.

Indication : pour la fonction de calcul du contour de l'image, il est conseillé de donner en paramètre d'entrée le point initial du contour (qui aura été déterminé au préalable par une autre procédure ou fonction).

Pour tester votre module, vous devrez créer des fichiers PBM texte avec des images-tests (de petites dimensions) pour lesquelles il est facile de déterminer le contour *à la main* afin de vérifier que les contours calculés par votre programme sont corrects, et permettant de couvrir l'ensemble des configurations possibles.

Partie 2 - stockage du contour dans une séquence de points

Complétez votre module avec différentes procédures ou fonctions afin de pouvoir :

1. récupérer le contour dans une **séquence** de points (la fonction `memoriser_position` consistera à ajouter la position dans une **séquence** de points qui, au préalable, aura été initialisée correctement).
Pour la **séquence** de points, il est conseillé d'utiliser des listes chaînées; pour cela, vous pouvez utiliser le code source `exemple_sequence_points.c` fourni dans le répertoire **TACHE3**.
2. calculer le contour, puis écrire à l'écran le nombre de segments composant le contour (on rappelle que pour un contour, le nombre de segment est égal au nombre de points moins 1),
3. écrire le contour (déjà stocké dans une **séquence** de points) dans un fichier texte au format suivant :
 - première ligne avec le nombre de contours (pour l'instant 1)
 - suivi des différents contours, chaque contour étant décrit par son nombre de points suivi des différents points au format **réel**, c'est à dire avec chaque coordonnée écrite avec décimale.Le fichier `image_ex_poly.contours` est un exemple de fichier de contours correspondant à l'exemple du polycopié de cours (`image_ex_poly.pbm`).

Compte-rendu

• Partie 1

Lorsque vous avez terminé la partie 1 de la tâche 3, créez un document PDF avec vos images-tests et les contours obtenus par votre programme.

Envoyez par e-mail un message à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE3-1 - noms_du_binome**

• Partie 2

Pour chaque image PBM du répertoire *Tache3*, exécutez votre programme pour créer un fichier-contour par image.

Dans un fichier nommé `resultats-tache3-2.txt`, indiquez pour chaque image testée :

- son nom,
- ses dimensions (largeur et hauteur),
- et le nombre de **segments** du contour calculé.

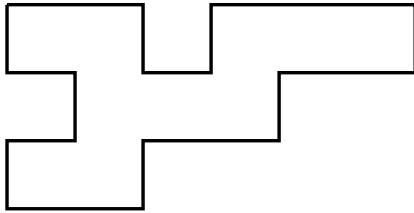
Envoyez un e-mail à votre enseignant.e de TP en joignant le fichier `resultats-tache3-2.txt`, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE3-2 - noms_du_binome**

Tâche 4 - Sortie fichier au format EPS

Modifiez/complétez votre programme de la tâche 3 afin de lire un fichier image PBM, en extraire le contour et créer un fichier EPS correspondant au contour.

Pour un contour fermé, il faudra notamment avoir la possibilité de le tracer sous l'un des deux modes suivants :



Mode 1 :
Tracé du contour (stroke)



Mode 2 :
Remplissage du contour (fill)

Note : pour le mode **stroke**, mettez l'épaisseur de tracé à 0.

Compte-rendu

- a) Pour chacune des deux images `chat.pbm` et `image_ex_poly.pbm`, créez deux fichiers EPS correspondants au contour de l'image pour les deux modes de tracé.
- b) Pour chacune des sept autres images `coq.pbm`, `elephant.pbm`, `gymnaste.pbm`, `labyrinthe.pbm`, `lettreZ.pbm`, `map401.pbm` et `tete.pbm`, créez le fichier EPS correspondant au contour de l'image uniquement avec le mode 2 (remplissage).

Lorsque vous avez terminé la tâche 4, envoyez par e-mail un message à votre enseignant.e de TP, en joignant les fichiers EPS créés, et en donnant accès aux codes sources de cette tâche (cf. page 2). Le sujet du message doit être : **[MAP401] - TACHE4 - noms_du_binome**

Tâche 5 - Extraction des contours d'une image

Récupérez sur les fichiers nécessaires à ce TP en copiant dans votre répertoire de travail, le répertoire suivant `/home/Public/401_MAP_Public/TACHE5/`.

Ce répertoire est aussi disponible sur le site de l'UE MAP401 de la plateforme Moodle-UGA dans Documents → TACHE5

Dans ce répertoire TACHE5 se trouvent différentes images PBM que vous utiliserez pour les comptes-rendus des deux parties.

• Partie 1 - Extraction des contours

Complétez les modules que vous avez déjà écrits afin de pouvoir extraire l'ensemble des contours d'une image.

Prévoyez une fonction ou procédure afin d'écrire l'ensemble des contours dans le format de fichier décrit dans la tâche 3 (comme exemple, cf. le fichier `image2_poly.contours.txt`)

Prévoyez une fonction ou procédure afin d'écrire à l'écran pour un ensemble de contours, le nombre de contours ainsi que la somme des nombres de **segments** de chaque contour.

Par exemple pour l'image `image2_poly.pbm`, il y a 3 contours et un total de 59 points soit un total de 56 segments.

• Partie 2 - Sauvegarde d'une suite de contours au format EPS

En reprenant ce que vous avez fait dans la tâche 4 :

- complétez votre projet en écrivant une procédure pour créer un fichier EPS correspondant à une séquence de contours,
- puis écrivez un programme lisant une image dans un fichier image PBM et créant un fichier EPS contenant la séquence des contours de l'image ; votre programme devra aussi écrire le nombre de contours de l'image ainsi que le nombre total de segments.

Compte-rendu

• Partie 1

Créez un document PDF contenant :

- les fichiers contours correspondant aux images `image1_poly.pbm`, `image2_poly.pbm`,
- les fichiers au format PBM et les fichiers contours correspondant à deux images tests que vous aurez créées,
- les infos (nombre de contours et somme des nombres de segments) pour les images suivantes :

| | | |
|---------------------------------|-------------------------------|-------------------------------|
| <code>Bugs_Bunny.pbm</code> | <code>Charlot.pbm</code> | <code>Pink_Panther.pbm</code> |
| <code>animaux.pbm</code> | <code>damier_4_5_1.pbm</code> | <code>deux-des.pbm</code> |
| <code>dessin-deliuss.pbm</code> | <code>gai-luron.pbm</code> | <code>papillon2.pbm</code> |

Envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE5-1 - noms_du_binome**

• Partie 2

Pour cette partie 2 de la tâche 5, il est demandé de créer les fichiers EPS (**mode remplissage** uniquement) obtenus à partir des images suivantes :

| | | | |
|------------------------------|------------------------------|---------------------------------|------------------------------|
| <code>image1_poly.pbm</code> | <code>image2_poly.pbm</code> | <code>France_Regions.pbm</code> | <code>Droopy_Wolf.pbm</code> |
|------------------------------|------------------------------|---------------------------------|------------------------------|

Envoyez un e-mail à votre enseignant.e de TP, en joignant les quatre fichiers EPS créés, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE5-2 - noms_du_binome**

Tâche 6 - Simplification de contours par segments

Pour la partie 2 de cette tâche, vous aurez besoin de six images PBM. Ces images font partie des images-tests (cf. page 2 de la partie Présentation).

• Partie 1 - Distance point-segment

Complétez votre module de géométrie 2D afin de pouvoir calculer la distance entre un point P et un segment $S = [A, B]$.

Ecrivez un programme permettant à un utilisateur d'entrer - au clavier ou par arguments du programme - trois points P , A et B puis de calculer et écrire à l'écran la distance entre le point P et le segment $S = [A, B]$.

• Partie 2 - Simplification de contour

Créez un module de simplification de contours et un programme l'utilisant.

À partir d'un fichier image PBM ou d'un fichier contenant une séquence de contours, et d'une distance-seuil d (d réel positif ou nul), le programme devra simplifier la séquence de contours suivant la distance-seuil d et écrire différents résultats (cf. compte rendu).

Compte-rendu

• Partie 1

Créez un document PDF contenant :

- le code source C de la fonction de calcul de la distance point-segment,
- le code source C du programme test,
- un jeu de tests couvrant l'ensemble des cas possibles et les distances *point-segment* correspondantes.

Envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE6-1 - noms_du_binome**

• Partie 2

Pour cette partie 2 de la tâche 6, il est demandé :

- les fichiers sources C + le fichier *Makefile* de la tâche 6,
- les fichiers EPS (mode *remplissage*) correspondant aux simplifications des images suivantes :

| | | |
|---------------------------|---------------------|----------------|
| image_poly_tache6.pbm | elephant-gotlib.pbm | goudyini-A.pbm |
| JoaquimHock-LesArbres.pbm | cheval.pbm | papillon2.pbm |

Pour chaque image, créez deux fichiers EPS, un pour la simplification avec la distance-seuil $d = 1$ et un pour la simplification avec la distance-seuil $d = 2$.

- un fichier texte nommé **resultats-tache6-2.txt** contenant pour chacune des six images :
 - . le nombre de contours,
 - . le nombre total de segments des contours de l'image initiale (avant simplification),
 - . le nombre total de segments des contours de la simplification avec $d = 1$,
 - . le nombre total de segments des contours de la simplification avec $d = 2$.

Envoyez un e-mail à votre enseignant.e de TP, en joignant le fichier **resultats-tache6-2.txt** et les fichiers EPS créés, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE6-2 - noms_du_binome**

Tâche 7 - Simplification de contours par Bézier

Pour cette tâche, vous aurez besoin de trois images PBM. Ces images font partie des images-tests (cf. page 2 de la partie Présentation).

- Afin de préparer les parties 1 et 2, complétez les modules que vous avez déjà écrits avec :
 - la définition de types `Bezier2` et `Bezier3` à partir du type `Point`,
 - le calcul du point $C(t)$ d'une Bézier de degré 2 ou 3, pour une valeur de t (entre 0 et 1),
 - la conversion d'une Bézier de degré 2 en Bézier de degré 3,
 - ...

• Partie 1 - Simplification par courbes de Bézier de degré 2

Complétez votre projet afin de permettre la simplification de contour(s) à l'aide de courbes de Bézier de degré 2, la sortie sous forme de fichiers PostScript et une information sur le nombre total de courbes de Bézier.

1.1 Ecrivez une fonction `approx.bezier2` calculant $\mathcal{B}(C_0, C_1, C_2)$ la Bézier de degré 2 approchant $(P_{j1}, P_{j1+1}, \dots, P_{j2})$ partie d'un contour polygonal entre les indices $j1$ et $j2$ avec $n = j2 - j1 > 0$. Pour vérifier que la fonction donne le résultat correct, les tests suivants devront être faits à l'aide d'un programme :

- (a) testez le cas $n = 1$: créez un contour polygonal avec 2 points $\{P_0, P_1\}$, et vérifiez que la courbe de Bézier calculée avec `approx.bezier2` donne $C_0 = P_0$, $C_1 = (P_0 + P_1)/2$, et $C_2 = P_1$.
- (b) testez le cas $n \geq 2$ de la manière suivante :
 - choisir n entier supérieur ou égal à 2, et trois points (Q_0, Q_1, Q_2) ,
 - à partir de la courbe de Bézier de degré 2 $\mathcal{B}(Q_0, Q_1, Q_2)$, créez le contour polygonal (P_0, P_1, \dots, P_n) tel que

$$P_i = C\left(\frac{i}{n}\right), \quad \forall 0 \leq i \leq n$$

c'est à dire choisir les $n + 1$ points P_i sur la courbe $\mathcal{B}(Q_0, Q_1, Q_2)$,

- calculez la Bézier de degré 2 $\mathcal{B}(C_0, C_1, C_2)$ approchant le contour polygonal (P_0, P_1, \dots, P_n) ,
- vérifiez que $C_0 = Q_0$, $C_1 = Q_1$, $C_2 = Q_2$.

- (c) testez l'exemple du cours avec 9 points (cf. page 51).

1.2 Implémentez l'algorithme de Douglas-Peucker pour simplifier un contour polygonal par une séquence de Bézier de degré 2 (en reprenant le canevas de l'algorithme de Douglas-peucker de la tâche 6).

1.3 Ajoutez une sortie au format EPS d'une (séquence de) séquence de Bézier de degré 2.

1.4 Ecrivez un programme qui, à partir d'une distance-seuil d et d'une image PBM ou d'un fichier séquence de contours correspondant à une image PBM, effectue la simplification par courbes de Bézier 2, et effectue les sorties similaires à la partie 2 de la tâche 6.

• Partie 2 - Simplification par courbes de Bézier de degré 3

Complétez votre projet afin de permettre la simplification de contour(s) à l'aide de courbes de Bézier de degré 3, la sortie sous forme de fichiers PostScript et une information sur le nombre total de courbes de Bézier.

2.1 Ecrivez une fonction `approx_bezier3` calculant $\mathcal{B}(C_0, C_1, C_2, C_3)$ la Bézier de degré 3 approchant $(P_{j1}, P_{j1+1}, \dots, P_{j2})$ partie d'un contour polygonal entre les indices $j1$ et $j2$ avec $n = j2 - j1 > 0$. Pour vérifier que la fonction donne le résultat correct, les tests suivants devront être faits à l'aide d'un programme :

- (a) testez le cas $n = 1$: créez un contour polygonal avec 2 points $\{P_0, P_1\}$, et vérifiez que la courbe de Bézier calculée avec la fonction `approx_bezier3` donne $C_0 = P_0, C_1 = (2 P_0 + P_1)/3, C_2 = (P_0 + 2 P_1)/3$, et $C_3 = P_1$.
- (b) testez le cas $n = 2$: créez un contour polygonal avec 3 points $\{P_0, P_1, P_2\}$, et vérifiez que la courbe de Bézier calculée avec la fonction `approx_bezier3` donne $C_0 = P_0, C_1 = (4 P_1 - P_2)/3, C_2 = (4 P_1 - P_0)/3$, et $C_3 = P_2$.
- (c) testez le cas $n \geq 3$ de la manière suivante :
 - choisir n entier supérieur ou égal à 3, et quatre points (Q_0, Q_1, Q_2, Q_3) ,
 - à partir de la courbe de Bézier de degré 3 $\mathcal{B}(Q_0, Q_1, Q_2, Q_3)$, créez le contour polygonal (P_0, P_1, \dots, P_n) ainsi

$$P_i = C \left(\frac{i}{n} \right), \quad \forall 0 \leq i \leq n$$

- c'est à dire choisir les $n + 1$ points P_i sur la courbe $\mathcal{B}(Q_0, Q_1, Q_2, Q_3)$,
 - calculez la Bézier de degré 3 $\mathcal{B}(C_0, C_1, C_2, C_3)$ approchant le contour polygonal (P_0, P_1, \dots, P_n) ,
 - vérifiez que $C_0 = Q_0, C_1 = Q_1, C_2 = Q_2, C_3 = Q_3$
- (d) testez l'exemple du cours (page 55).

2.2 Implémentez l'algorithme de Douglas-Peucker pour simplifier un contour polygonal par une séquence de Bézier de degré 3 (en reprenant le canevas de l'algorithme de Douglas-peucker de la partie 1 avec les Bézier de degré 2).

2.3 Ajoutez une sortie au format EPS d'une (séquence de) séquence de Bézier de degré 3.

2.1 Ecrivez un programme qui à partir d'une distance-seuil d et d'une image PBM ou d'un fichier séquence de contours correspondant à une image PBM, effectue la simplification par courbes de Bézier 3, et effectue les sorties similaires à la partie 1 de la tâche 7.

Compte-rendu

• Partie 1

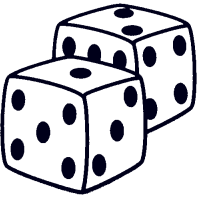
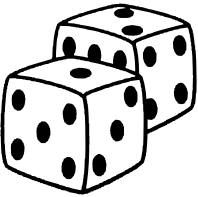
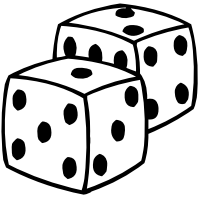

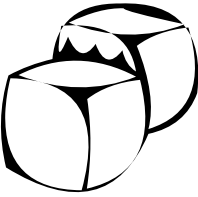
Créez un document PDF contenant :

- les ajouts (et/ou modifications) que vous aurez effectués dans vos différents modules : mettez dans le rapport les codes sources commentés des ajouts (et/ou modifications) faits dans les différents fichiers *spécification* (.h) .
- le code source du programme pour les tests unitaires de la partie 1.1.
- pour chacune des images suivantes

| | | |
|--------------|----------------------|----------------------|
| Asterix3.pbm | lettre-L-cursive.pbm | ColombesDeLaPaix.pbm |
|--------------|----------------------|----------------------|

un tableau contenant l'image EPS (mode remplissage) des contours initiaux, les images EPS (mode remplissage) des contours simplifiés par courbes de Bézier de degré 2 et pour les distances-seuils $d = 1$, $d = 3$, $d = 10$ et $d = 30$. Indiquez pour chaque image simplifiée, le nombre total de courbes de Bézier.

Exemple de tableau récapitulatif pour une image

| | | | | |
|---|---|---|--|---|
|  |  |  |  |  |
| Image initiale (20 contours) 9466 segments | Simplification pour $d = 1$ 746 Bézier | Simplification pour $d = 3$ 129 Bézier | Simplification pour $d = 10$ 86 Bézier | Simplification pour $d = 30$ 38 Bézier |

Envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : [MAP401] - TACHE7-1 - noms_du_binome

• Partie 2

Créez un document PDF contenant :

- les ajouts (et/ou modifications) que vous aurez effectués dans vos différents modules : mettez dans le rapport les codes sources commentés des ajouts (et/ou modifications) faits dans les différents fichiers *spécification* (.h) .
- le code source du programme pour les tests unitaires de la partie 2.1.
- pour chacune des images suivantes

| | | |
|--------------|----------------------|----------------------|
| Asterix3.pbm | lettre-L-cursive.pbm | ColombesDeLaPaix.pbm |
|--------------|----------------------|----------------------|

un tableau contenant l'image EPS (mode remplissage) des contours initiaux, les images EPS (mode remplissage) des contours simplifiés par courbes de Bézier de degré 3 et pour les distances-seuils $d = 1$, $d = 3$, $d = 10$ et $d = 30$. Indiquez pour chaque image simplifiée, le nombre total de courbes de Bézier.

Envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : [MAP401] - TACHE7-2 - noms_du_binome

Tâche 8 - Tests de robustesse et performance

Comparatif des simplifications

Récupérez sur les fichiers nécessaires à ce TP en copiant dans votre répertoire de travail, le répertoire suivant `/home/Public/401_MAP_Public/TACHE8/`

Ce répertoire est aussi disponible sur le site de l'UE MAP401 de la plateforme Moodle-UGA dans Documents → TACHE8

Dans ce répertoire TACHE8 se trouvent différentes images PBM que vous utiliserez pour les comptes-rendus des deux parties.

La partie 1 doit être faite uniquement avec les programmes de la tâche 6.

La partie 2 doit être faite avec les programmes de la tâche 6, et éventuellement complétée avec les programmes de la partie 1 de la tâche 7, et la partie 2 de la tâche 7.

• Partie 1 - Tests de robustesse et performance

Le but de cette partie est d'utiliser vos programmes avec certaines images complexes (dimensions des images, nombre de contours et nombre total de segments de l'image initiale) afin de tester leur robustesse (fonctionnent-ils correctement ou non ?) et observer leur temps d'exécution.

Pour chaque image, effectuer l'extraction des contours puis la simplification par segments avec la distance-seuil $d = 0$.

1) Testez pour chaque image si le processus complet s'effectue correctement, c'est à dire si votre (ou vos) programme(s) ne plante(nt) pas.

2) Observez pour chaque image le temps d'exécution du processus complet : pour cela, utiliser la commande système `time` suivie de la commande correspondant à votre programme.

Par exemple, si votre commande de simplification est :

```
simplification zebres-2000x1500.pbm 0
```

alors exécutez la commande :

```
time simplification zebres-2000x1500.pbm 0
```

et à la fin du programme, notez le temps indiqué avec `u` (pour `user`) : ce temps est le temps réel d'exécution (utilisation du processeur) de votre programme.

Les images-tests sont :

A) série *zebres* : image – d'un groupe de zèbres – scannée à différentes tailles.

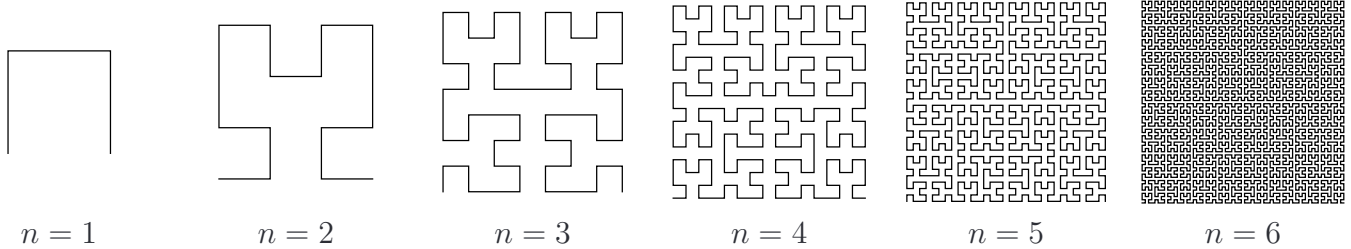
| | | | |
|-----------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|
| <code>zebres-1000x0750.pbm</code> | <code>zebres-2000x1500.pbm</code> | <code>zebres-3000x2250.pbm</code> | <code>zebres-4000x3000.pbm</code> |
|-----------------------------------|-----------------------------------|-----------------------------------|-----------------------------------|

B) série *courbe de Hilbert* : images de courbes de Hilbert

| | | | |
|-----------------------------------|-----------------------------------|-----------------------------------|------------------------------------|
| <code>courbe_hilbert_7.pbm</code> | <code>courbe_hilbert_8.pbm</code> | <code>courbe_hilbert_9.pbm</code> | <code>courbe_hilbert_10.pbm</code> |
|-----------------------------------|-----------------------------------|-----------------------------------|------------------------------------|

La courbe de Hilbert d'ordre n est une courbe polygonale contenue dans le carré unité $[0, 1] \times [0, 1]$ et passant par tous les points $\left(\frac{2i+1}{2^{n+1}}, \frac{2j+1}{2^{n+1}}\right)$ avec $0 \leq i, j \leq 2^n - 1$

Courbes de Hilbert d'ordre 1 à 6



L'image correspondant à la courbe de Hilbert d'ordre n est une image carrée de dimensions $L = H = 2^{n+1} - 1$ avec un seul contour passant par tous les points soit un contour initial avec 4^{n+1} segments.

• Partie 2 - Comparatif des simplifications

Le but de cette partie est de comparer les différentes simplifications en terme de résultat visuel et de taille des contours simplifiés.

Pour chacune des images suivantes (de dimension 500×500) :

| | | |
|--------------|----------------------|----------------------|
| Asterix3.pbm | lettre-L-cursive.pbm | ColombesDeLaPaix.pbm |
|--------------|----------------------|----------------------|

1. effectuez l'extraction de contour(s) et notez le nombre de contours et nombre total de segments,
2. effectuer la simplification :

— pour chacune des méthodes de simplification :

| | | |
|---------|-------------------|-------------------|
| Segment | Bézier de degré 2 | Bézier de degré 3 |
|---------|-------------------|-------------------|

— et pour chacune des distances-seuils suivantes :

| | | | | | | |
|---------|-----------|---------|---------|---------|---------|----------|
| $d = 0$ | $d = 0.5$ | $d = 1$ | $d = 2$ | $d = 4$ | $d = 8$ | $d = 16$ |
|---------|-----------|---------|---------|---------|---------|----------|

Pour chaque test (1 test = 1 méthode + 1 distance-seuil), notez le nombre total d'éléments après simplification.

Compte-rendu

- Créez un document PDF pour l'ensemble de la tâche 8, et contenant :

Partie 1 : pour chaque image-test, indiquez si le processus complet (extraction de contour puis simplification) s'est fait correctement.

Si oui, indiquez le temps **user** donné par la commande **time**.

Sinon, indiquez l'erreur rencontrée et éventuellement la cause.

Partie 2 : pour chaque image-test, faites un tableau récapitulatif avec pour chaque test (méthode + distance-seuil), le nombre total d'éléments après simplification, et écrivez un commentaire sur les résultats obtenus.

Pour l'image **Asterix3.pbm**, faites un tableau supplémentaire sur une page avec les différentes images EPS (mode remplissage) obtenues pour les méthodes de simplification et les distances-seuils $d = 1$, $d = 2$, $d = 4$ et $d = 8$, et écrivez un commentaire sur la qualité visuelle des images obtenues. Envoyez un e-mail à votre enseignant.e de TP en joignant le document PDF, et en donnant accès aux codes sources de cette tâche (cf. page 2).

Le sujet du message doit être : **[MAP401] - TACHE8 - noms_du_binome**

Evaluation finale

Rapport, codes sources et soutenance orale

L'évaluation finale de l'UE MAP401 consiste en trois éléments : un rapport écrit, les codes sources et une soutenance orale.

Le rapport écrit et les codes sources devront être fournis à votre enseignant.e de TP au moins deux jours ouvrables avant la soutenance finale.

Pour cela, envoyez un e-mail à votre enseignant.e de TP en joignant le rapport écrit au format PDF, et en indiquant le chemin complet sur `turing` pour récupérer les codes sources de votre programme et le fichier `Makefile`.

Le sujet du message doit être : **[MAP401] - TACHE_FINALE - noms_du_binome**

Rapport final

Dans ce rapport, pour chaque tâche, vous décrirez les solutions que vous avez envisagées, les structures de données que vous avez utilisées, la structure de chaque programme (module unique ou différents modules), et les problèmes éventuels que vous avez rencontrés.

Vous ferez aussi un manuel utilisateur pour chaque programme des taches 5, 6 et 7, c'est à dire en indiquant le ou les fichiers sources (fichiers `.c` / `.h`), comment faire la compilation, comment exécuter le programme, quelles sont les données en entrée et quels sont les résultats en sortie.

Vous mettrez aussi dans ce rapport votre diagramme de Gantt final et votre journal de bord final.

Le rapport final devra être envoyé au format PDF, puis imprimé et apporté lors de la soutenance.

Codes source

Préparez un répertoire final avec l'ensemble des fichiers de votre projet (programmes sources `.c` et `.h`, `Makefile`, scripts shell, ...) qui permettent de créer les fichiers exécutables.

Indiquez le chemin de ce répertoire si vous l'avez mis sur `turing`, ou envoyez-le sous forme d'une archive à votre enseignant.e.

La qualité des codes sources sera principalement évaluée : on prendra en compte la modularité, et la lisibilité du code (commentaires, indentation, noms des fonctions/variables, ...).

Soutenance orale

Les soutenances orales sont prévues entre 29 avril et le 7 mai 2024 (semaines civiles 18 et 19).

Pendant votre soutenance, vous aurez une démo à faire qui devra durer entre 5 et 8 minutes, qui consistera à présenter vos réalisations logicielles sur les parties "Extraction de contour(s) à partir d'image PBM" et "Simplification de contour(s)". Après votre démo, vous aurez à répondre à quelques questions.

Veillez à préparer votre démo afin de ne pas perdre de temps lors de celle-ci : par exemple, préparez un script ou un fichier texte avec les différentes commandes d'exécution de votre programme (ou vos programmes).

Ne présentez pas vos codes sources, et ne modifiez pas vos codes sources pendant votre démo.

Utilisez des exemples pertinents (images / distances-seuils) lors votre démo.