COTTEL Thibault - CHAREF Sohel INSA 3A-AVM

COMPTE-RENDU TP INFOGRAPHIE 2023

Thème : La Plage

SOMMAIRE :

I. Objectifs

L'objectif de ce projet est de réaliser un programme infographique produisant une image sur le thème de la plage. Nous avons donc décidé de partir sur une image de taille 500x500 car cela nous semble être la meilleure solution pour avoir une image d'assez bonne qualité tout en ayant une réponse au calcul assez rapide. Vous pourrez voir ci dessous le dessin prototype qui nous a servi de référence pour notre dessin final

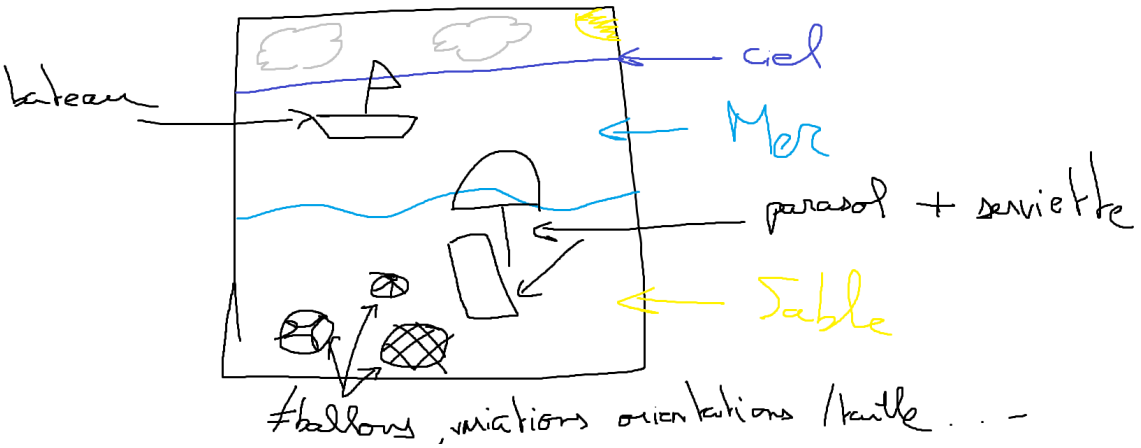

Figure 1 : 1er Dessin prototype

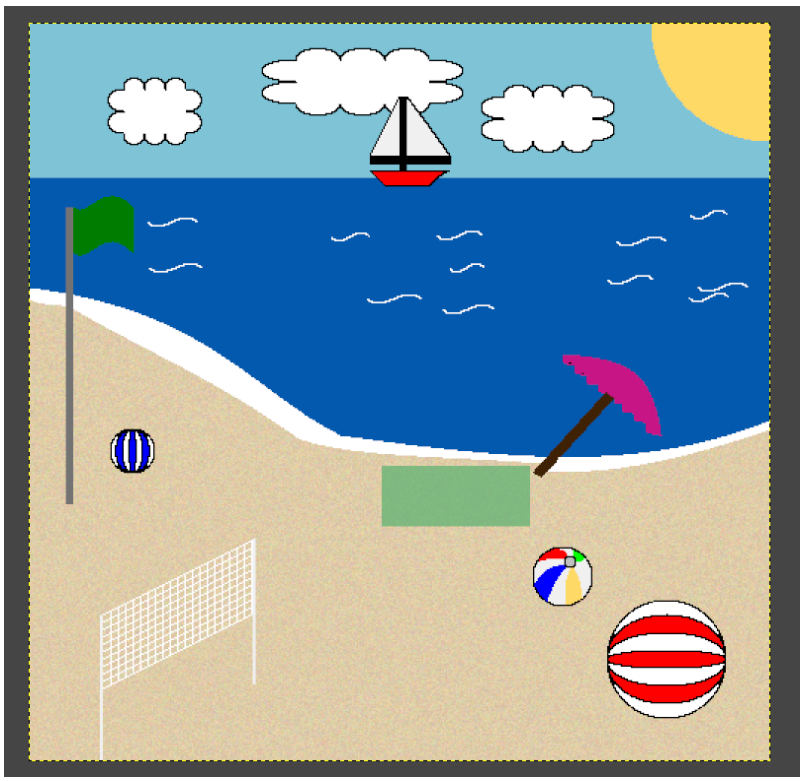

Figure : Dessin finale (les vagues sont générées de manière aléatoire)

II. Réalisation

A. Fonctions de dessins de formes

▌int rectangle(SURFACE *s,int Posx, int Posy, int lengthx, int lengthy, Color c) :

Fonction qui trace un rectangle. Les coordonnées d'entrée sont celles du point en haut à gauche de ce rectangle.

▌void cercle(SURFACE *s, int posX,int posY,int ray,Color c) :

Fonction qui trace le contour d'un cercle de rayon r donné en paramètre. Les coordonnées données en entrée désignent le centre du cercle.

▌void cercle_color(SURFACE *s, int x, int y, int ray,Color c) :

Fonction qui permet de tracer un cercle coloré. Pour remplir de couleur le cercle on trace des cercles de rayons inférieurs à l'intérieur de celui-ci. Les coordonnées données en entrée désignent le centre du cercle.

▌void courb(SURFACE *s, int x0, int y0, int x1, int y1, int x2, int y2, int x3, int y3, Color c) :
Fonction qui utilise les courbes de Bézier de degré 4 afin de tracer une courbe.

▌void ligne(SURFACE *s, int x1, int y1, int x2, int y2, Color c) :

Fonction qui permet de tracer des lignes en indiquant le point de départ et le point d'arrivée, ainsi qu'une couleur.

▌int filtre_rectangle(SURFACE *s,int Posx, int Posy, int lengthx, int lengthy, Color c) :

Fonction qui trace un rectangle coloré en y ajoutant du bruit.

▌void ballon(SURFACE *s, int centreX, int centreY ,int rayon, int rotation, Color contour, Color c1, Color c2, int nbrParties) :

Fonction qui dessine un ballon de plage pour des coordonnées x et y données et pour un couple de couleur données. La taille du ballon peut varier en fonction du rayon donnée. Le paramètre nbParties permet d'ajouter des courbes au ballon ou d'en retirer. Enfin vous pouvez aussi faire varier la rotation en degrés du ballon dans le plan.

▌void ballon2(SURFACE *s, Color Contour, Color c1, Color c2, Color c3, Color c4, Color interieur1, Color interieur2,int posX,int posY) :

Fonction qui permet de tracer un ballon composé de 6 couleurs + 1 contour. Ce ballon est placé selon des coordonnées données en entrée de la fonction.

▌void navire(SURFACE *s,int x, int y, Color contour, Color coque, Color voile) :

Fonction qui permet de dessiner un navire à une position donnée, pour des couleurs données (possibilité de personnaliser les contours, la coque ou le voile).

▌void drapeau(SURFACE *s,int x, int y, int longeurPoteau, Color colorPoteau, Color couleurDrapeau) :

La fonction drapeau permet de tracer un drapeau d'autorisation de baignade accompagné de son poteau à des coordonnées (le point en bas à gauche du poteau), une hauteur et des couleurs de poteau et de drapeau données. Le drapeau peut donc prendre seulement des couleurs uniforme, nous traçons d'abords le poteau sous forme d'un rectangle puis le drapeau

▌void vagues(SURFACE *s, int x, int y, int hauteurPoteau, Color couleurVagues) :

Cette fonction permet de tracer des vagues à l'aide des courbes de Bézier, on entre en paramètre la longueur des vagues ainsi que la couleur et les coordonnées du point de départ de la vague. Dans le main cette fonction est utilisée dans une boucle qui fait varier de façon semi-aléatoire les valeurs de x et y à chaque exécution du programme. Donc les vagues sont placé de manière différente à chaque exécution du programme.

▌void parasol(SURFACE *s, Color ColorMat, Color ColorToile) :

Cette fonction permet de tracer un parasol à l'intérieur d'une surface donnée en paramètre. La fonction prend aussi en entrée les couleurs du mât et de la toile du parasol. A noter que cette fonction ne prend pas de valeurs x et y, donc le parasol est fixe, ici la fonction sert simplement à dégager de l'espace dans le main.

▌void filet_volley(SURFACE *s,Point p1, Point p2, Color colorPoto, Color colorFilet) :

Fonction qui trace un filet de volley à l'aide des fonctions rectangle et draw_line_point.

▌void ecume(SURFACE *s, Color Couleur) :

Fonction qui dessine l'écume aux abords de la mer. Cette fonction dessine les courbes et remplit l'intérieur à l'aide de la fonction remplissageProp (voir ci-dessous).

▌void nuage(SURFACE *s, int xSet, int ySet, int length, Color colorLineNuage,Color colorInsideNuage, int isBehind) :

Fonction qui trace des nuages dans le ciel. Cette fonction vérifie la position du nuage à dessiner et elle le trace en fonction, c'est-à-dire qu'elle ajuste si le nuage doit se retrouver au premier plan ou au second.

B. Fonctions de remplissage de forme

▌void surface(SURFACE *s, int width, int height) :

Fonction qui initialise une surface vide avec la taille de celle ci (longeur et largeur) alloué en mémoire sous la forme d'un tableau à 2 dimension d'entier

- La couleur que nous voulons modifier (0 = Rouge | 1 = Vert | 2 = Bleu)

- L’indice du pixel de la surface (de gauche à droite, et de haut en bas)

▌void fill(SURFACE *s, double valueR, double valueG, double valueB) :

Rempli entièrement la surface d’une couleur R G B voulue.

▌int pixel_set(SURFACE *s, int x, int y, Color c) :

Met un pixel de position (x,y) de la surface à une couleur (c) donnée.

▌void remplissageProp(SURFACE *s, int x, int y, Color c) :

Remplissage 4-connexe d’une couleur donnée (c) à partir d’un point (x,y).

▌void remplissagePropAvecBruit(SURFACE *s, int x, int y, Color c, double degradation) :

Remplissage 4-connexe d’une couleur donnée (c) à partir d’un point (x,y) avec du bruit dont nous pouvons donner le taux de dégradation.

▌void free_surface(SURFACE *s) :

Libère totalement toute les cases de la surface alloué dans la case mémoire

C. Autres fonctions

▌int pgm_write(SURFACE *s, FILE *f) :

Fonction qui écrit une surface donnée sur un fichier donné.

D. Structure du programme

Concernant la structure du programme, nous l’avons réalisé de manière progressif. Nous avons d’abord écrit les fonctions de base pour notre dessin, que ce soit la création de la surface, la création des piles puis les fonctions un peu plus complexes tel que le remplissage 4-connexe de la surface avec une pile. Pour enfin écrire les fonctions des objets de plage tel que le navire par exemple.

Toute la stratégie de dessin a surtout été réalisée à la fin dans le main() notamment l’ordre d’appel des fonctions. Ceci nous a permis de résoudre beaucoup de problèmes liés au remplissage des formes. Après avoir mis en place une surface de dessin remplie d’une couleur hors de nos limites pour déterminer la partie non coloriée (256,256,256). Nous avons commencé par dessiner les objets présents dans notre plage, puis nous avons rempli le ciel, la mer et la plage pour enfin finir avec les vagues.

E. Répartition du travail et difficultés rencontrées

Une grande partie du travail a été réalisé lors des séances de TP, c'est le cas pour les fonctions permettant de tracer un segment, un cercle, un rectangle par exemple. Pour ce qui est du reste, la partie remplissage à été principalement réalisée par Thibault et la question de l'agrandissement et de la rotation d'objet à été traitée par Sohel. Aussi, nous nous sommes répartis égalitairement les objets à dessiner.

Concernant les difficultés rencontrées par Thibault:

La fonction remplissageProp qui effectue un remplissage par propagation a été plutôt complexe. En effet, nous avons souvent rencontré des problèmes de remplissage avec des formes non remplies ou alors partiellement remplies. De plus, certaines formes (notamment les courbes avec une pente élevée) ne ferment pas entièrement l'espace. La décision prise a été de corriger manuellement les pixels non dessinés.

Concernant les difficultés rencontré par Sohel:

La fonction vagues avait 2 problèmes: Lorsque la vague touchait un autre objet, la vague dessinait au-dessus, et aussi lorsque la vague était dessinée avant le remplissage de la mer, si 2 vagues créait un espace accessible en 8-connexe, cela créait une case vide. Pour remédier à cela, le placement des vagues varie dans un espace prédéfini et est généré après le remplissage de la mer.

Aussi la fonction ballon a été très compliquée à mettre en place à cause de problèmes d'écriture liée au calculs approximatifs des valeurs de positions sur la surface de sortie. Mais, après une correction avec une approximation des entier à l'aide de la fonction math.round(), nous pouvons le faire fonctionner pour des valeurs de k*90° de manière fiable.

III. Conclusion

Pour conclure, nous trouvons que nous avons fait une bonne adaptation de la plage que nous avions anticipé au départ. Nous avons en plus pu y ajouter des améliorations que ce soit le drapeau de la plage ou notre filet de volley-ball. Finalement, l'ajout de nouveaux objets a été assez facile grâce à notre travail antérieur sur les fonctions de dessin de forme.