Exo3- explications

* Pourquoi les entêtes de fonction ne sont pas dans un header à part ?pour qu’on puisse les utiliser quand on veut ; si on ne fait pas ça, les fonctions ne pourront être utilisées que si elles sont définies avant.
* S’il n’y a pas d’interaction avec le clavier faut-il quand même intégrée la fonction glutKeyboardFunc(&window\_key); dans le main ?
* La fonction : glutMainLoop(); fait une boucle sur tout ce qui précède ? fait une boucle sur le programme principal.
* Comment se passe la définition de la couleur ? les couleurs sont définies dans des define en début de code, pourquoi ? on règle ainsi la proportion de chaque couleur. On ne peut mettre que des 0 ou des 1, les autres valeurs ne change rien.
* Comment on fait si on veut faire des dégradés sur le fond ?
* Paramètre alpha ? affiche la croix blanche quand on est à 1 ou plus, n’affiche plus la croix blanche quand on met à 0.
* Normalisation du vecteur avant le calcul des nouvelles coordonnées car sinon on translate B pas au bon endroit, c’est récupérer uniquement la distance AB.

========================================================

int main(int argc, char \*\*argv)

{

// initialisation des paramètres de GLUT en fonction

// des arguments sur la ligne de commande

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_RGBA);

// définition et création de la fenêtre graphique, ainsi que son titre

glutInitWindowSize(WIDTH, HEIGHT);

glutInitWindowPosition(0, 0);

glutCreateWindow("Premier exemple : carré");

// initialisation de OpenGL et de la scène

initGL();

init\_scene();

// choix des procédures de callback pour

// le tracé graphique

glutDisplayFunc(&window\_display);

// le redimensionnement de la fenêtre

glutReshapeFunc(&window\_reshape);

// la gestion des événements clavier

glutKeyboardFunc(&window\_key);

// la boucle prinicipale de gestion des événements utilisateur

glutMainLoop();

return 1;

}

===============================================================

GLvoid est un type de retour de même que GLint, GLdouble… OpenGL a ses propres type de retour pour des questions de compatibilité et de portabilité entre les différents langages.

GLvoid initGL()

{

glClearColor(RED, GREEN, BLUE, ALPHA);

}

Cette fonction permet de réinitialiser le fond à chaque fois que l’on lance l’application.

=====================================================================

// Initialisation de la scene. Peut servir à stocker des variables de votre programme

// à initialiser

void init\_scene()

{

}

==============================================================

// fonction de call-back pour l´affichage dans la fenêtre

GLvoid window\_display()

{

glClear(GL\_COLOR\_BUFFER\_BIT);//le parameter est enfaite une constant qui représente toutes les couleurs présentent dans le fond

glLoadIdentity();

// C'est l'endroit où l'on peut dessiner. On peut aussi faire appel

// à une fonction (render\_scene() ici) qui contient les informations

// que l'on veut dessiner

render\_scene();

// trace la scène grapnique qui vient juste d'être définie

glFlush();

}

=================================================

GLvoid window\_reshape(GLsizei width, GLsizei height)

{

glViewport(0, 0, width, height);

glMatrixMode(GL\_PROJECTION);

//permet de sélectionner la matrice que l’on souhaite utilisée, la matrice courante. La matrice de projection sert uniquement pour l’affichage. Cette fonction doit être appelée chaque fois que l’on change de matrice.

glLoadIdentity();

// ici, vous verrez pendant le cours sur les projections qu'en modifiant les valeurs, il est

// possible de changer la taille de l'objet dans la fenêtre. Augmentez ces valeurs si l'objet est

// de trop grosse taille par rapport à la fenêtre.

glOrtho(-2.0, 2.0, -2.0, 2.0, -2.0, 2.0);

// toutes les transformations suivantes s´appliquent au modèle de vue

glMatrixMode(GL\_MODELVIEW);

}

# Affichage des points et vecteurs :

1. On commence par intégrer les entêtes des fonctions :

Void DessinePoint (const Point&) et void DessineVector (const Point&, const Vector&)

1. Dans la méthode render\_scene ;

On commence par créer les points ou vecteurs que l’on souhaite afficher. Puis on fait appel à la fonction dessinePoint en passant ce point en paramètre.

1. Après toutes les fonctions on fait une nouvelle section où l’on va définir la fonction dessinePoint.
2. Optimisation Ok