ISIMA 1ère année : TP C N°7 --- Graphisme 3D sous X-Windows
Introduction à la programmation par évènements

Le but de ce TP est de manipuler quelques primitives graphiques de la bibliothèque Xlib sur une station de travail UNIX ou sur un terminal X. Dans votre boîte aux lettres vous avez reçu 2 fichiers `Xmtc.c` et `Xlib.c`.

- Sauvegarder les fichiers `Xmtc.c` et `Xlib.c`, les compiler et les exécuter.

- Affichage des courbes des fonctions lues dans le cadre du TP6 pour le G3 ou TP4 pour les G1 et G2 .

  Méthodologie : Etudier le code de Xlib.c et de Xmtc, puis créer un nouveau fichier recopiant Xlib.c et intégrant dans la partie *afficheGraphisme* le calcul et l'affichage des fonctions. Vous ferrez 2 types d'affichage, l'un uniquement avec des points et l'autre avec des segments de droites reliant 2 points. Calculez l'échelle et le pas d'affichage en fonction de la taille de la fenêtre graphique, de votre intervalle [a, b] et des valeurs minimales et maximales attendues.

  Pour commencer vous tracerez les axes et testerez la fonction y = x en fixant des constantes par exemple : affichage sur une fenêtre de 400 par 400 pixels (avec x et y dans [-2, 2]), vous essayerez ensuite avec un sinus, puis enfin vous incorporerez les évaluations du TP6/TP4.

Les étudiants avancés en programmation peuvent gérer des pixmaps pour l'affichage :

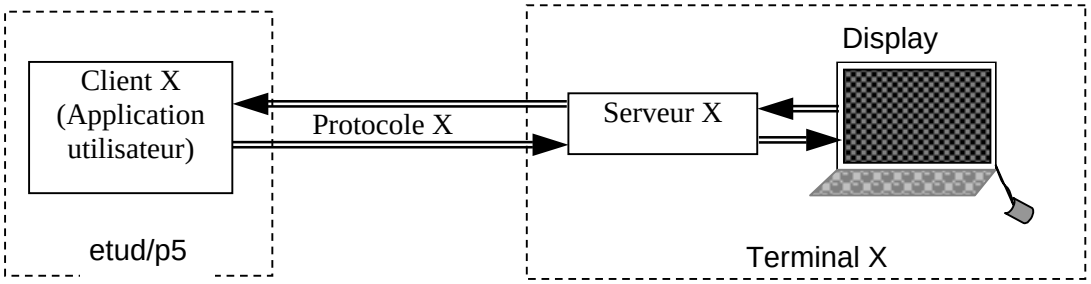
- faire un 1er affichage à l'écran,
- capturer cet écran dans un pixmap,
- rafficher ce pixmap en cas d'événements "Expose".

Vous vérifiez aussi la libération des ressources X (phase 5) qui ne sont pas toutes prises en compte dans les codes donnés en exemple. Piste supplémentaires : choix du type d'affichage (point ou ligne) avec des événements clavier, zoom défini grâce à la souris (définition d'un zone et cadrage de l'affichage de `f(x)` sur cette zone).

ISIMA 1ère année : Graphisme sous X-Window

Système X Window

Le système X Window est basé sur un modèle client/serveur. Le serveur et le client sont deux processus distincts qui communiquent entre eux via le protocole X.



Une application Xlib comprend 5 phases :

- Déclarations générales du programme : inclusion des fichiers d'entête, déclarations de variables globales …
- Connexion avec le serveur X :

```
Display * XOpenDisplay(char *dpy);
```

- Initialisations spécifiques à X Window :

```
DefaultScreen(Display *dpy);
WhitePixel(Display * dpy, int ecran);
...
Window XcreateSimpleWindow(Display *, Window win,
      int x, int y, int width, int height, int bord,
      unsigned long avant_plan, unsigned long arr_plan);
…
```

- Boucle principale pour la réception et le traitement des événements :

```
while ( !sortie) {
     XNextEvent( dpy, evenement) ;
     …
}
```

Un programme Xlib doit réceptionner les événements provenant du serveur, les analyser et réagir en fonction de type d'événement. Les événements qu'un programme traite sont sélectionné dans la phase 3) via

```
XSelectInput(Display *dpy, Window fenetre,
             long masque_evenement);
```

Rappel de quelques événements :

Nom de l'événement	Nom du masque associé	Description
`ButtonPress`	`ButtonPressMask`	appui sur un bouton de la souris
`ButtonRelease`	`ButtonReleaseMask`	bouton de la souris relâché
`MotionNotify`	`ButtonMotionMask`	déplacement du pointeur de la souris avec un bouton appuyé
`KeyPress`	`KeyPressMask`	une touche a été appuyée

- Libération des ressources :

```
XFreeGC(Display *dpy, GC context_graph);
XDestroyWindow(Display *dpy, Window fenetre);
XCloseDisplay(Display *dpy);
```

Quelques fonctions graphiques pour l'affichage de fonction :

(1) `XDrawLine(Display dpy, Drawable fenetre, GC gcontexte, int x1, int y1, int x2, int y2);`
(2) `XDrawPoint(Display dpy, Drawable fenetre, GC gcontexte, int x, int y);`
(3) `XFillRectangle(Display dpy, Drawable fenetre, GC gcont, int x, int y, unsigned int largeur, unsigned int hauteur);`
(4) `XSetForeground(Display dpy, GC gcontexte, unsigned long couleur);`
(5) `XSetBackground(Display dpy, GC gcontexte, unsigned long couleur);`
(6) Les couleurs avant ou arrière plan, l'épaisseur de trait sont aussi définies dans contexte graphique. L'utilisation de couleurs personnelles peut être réalisée via l'allocation de couleur dans un colormap : voir `DefaultColormap`, `XAllocNamedColor`, …. Une liste de couleurs est défnie dans le fichier `/usr/X11R6/lib/X11/rgb.txt`.

et quelques fonctions qui vous seront utiles pour la manipulation des "pixmap"s :

(1) `pixmap XCreatePixmap (display, fenetre, taille_x, taille_y, codage_coul);`
(2) `codage_coul DefaultDepth(display, ecran);`
(3) `XCopyArea(display, drawable_origine, drawable_destination, contexte_graphique, origine_x, origine_y, taille_, taille_y, destination_x, destination_y);`
(4) `XFreePixmap(display, pixmap);`
(5) `XClearWindow(display, fenetre);`

Compilation

La compilation de programmes Xlib nécessite l'utilisation de la bibliothèque X11. Il faut :

- indiquer (souvent) explicitement l'emplacement des fichiers d'entête X11 avec l'option **–I** du compilateur
- et celui de la bibliothèque avec l'option **–L**
- effectuer une édition de lien avec la bibliothèque X11 avec l'option **–l**

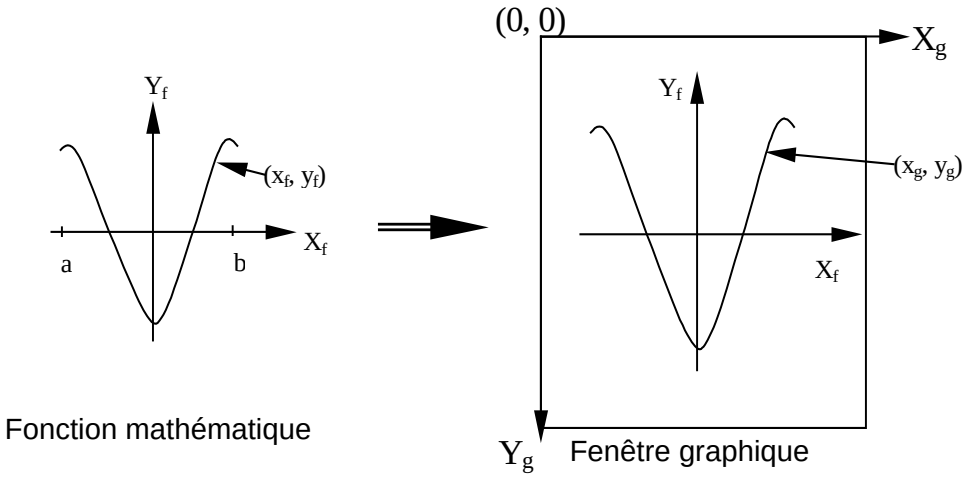
Exemple de compilation :

gcc –I/usr/X11R6/include –L/usr/X11R6/lib Xmtc.c –o Xmtc –lX11

Base graphique sous X Window

Une fenêtre graphique a son propre système de coordonnées (voir la partie droite de la figure ci-dessous). Les coordonnées d'un point (pixel) de la fenêtre graphique sont des entiers qui varient de `0` jusqu'à `L–1` ou `H-1`, `L` et `H` étant la largeur et la hauteur de la fenêtre graphique. Une fonction peut être visualisée avec une suite de points ou une suite de segments de droite.

Soient `[a, b]` l'intervalle en abscisse d'une fonction `y=f(x)` qu'on voudrait visualiser dans une fenêtre graphique. La figure ci-dessous représente une fonction dans son propre système de coordonnées (la partie gauche) et sa visualisation dans une fenêtre graphique (la partie droite).



Pour afficher un point (x_f, y_f) d'une fonction dans une fenêtre graphique, on doit d'abord calculer les coordonnées du pixel (x_g, y_g) correspondant. Deux opérations de transformations sont nécessaires pour cela :

- Changement d'échelle en X_f et Y_f de la fonction, pour ajuster la dimension de la fonction. Les facteurs maximaux de changement d'échelle sont :

  $$h_x = L/(b-a) \; ; \; h_y = H/(y_{max} - y_{min})$$

  y_{max} et y_{min} étant les valeurs maximale et minimale de la fonction dans l'intervalle `[a, b]`.
- Passage du système de coordonnées de la fonction dans le système de coordonnées de la fenêtre graphique :

  $$x_g = L/2 + x_f * h_x \; ; \; y_g = H/2 - y_f * h_y$$

  si on veut placer l'origine du système de coordonnées de la fonction au centre de la fenêtre graphique.

Rq. : *Le système de coordonnées de la fonction est en réel et le système de coordonnées de la fenêtre graphique est en entier. Attention à la conversion de type.*