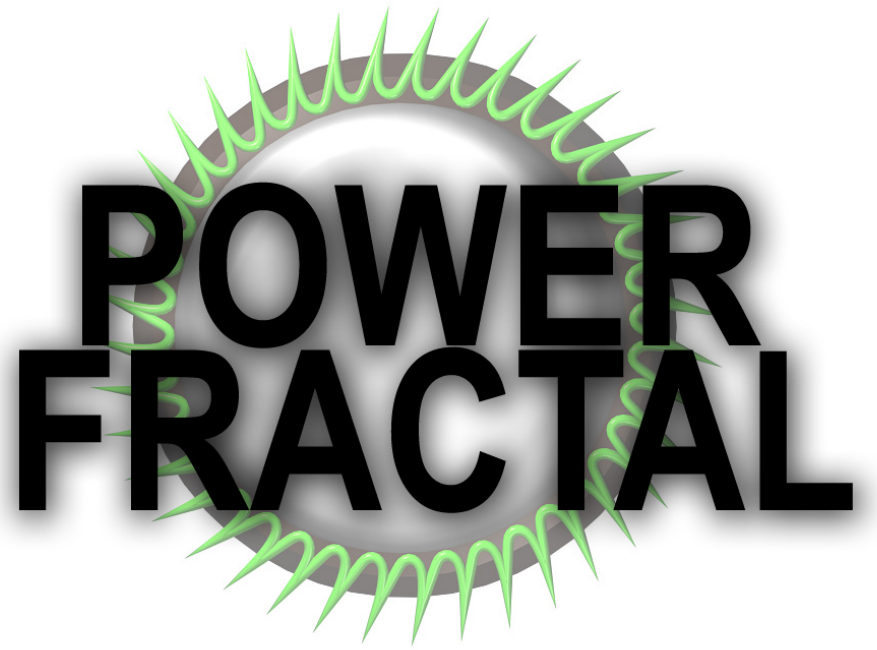
Projet de semestre 2001



G.Burri & A.Crivelli
[EIA]

Table des matières

1. Résumé

L'algorithme de base des fractales étant relativement simple, nous avons choisis de concentrer nos efforts sur les outils de 'navigation' dans la fractale ainsi que l'interface du programme avec l'utilisateur.

Connaissant peu les possibilités d'interfaçage avec Ada, nous avons essayé plusieurs possibilités. Profitant d'une utilisation parallèle de l'outil GTK-Ada, nous avons pensé que c'était un bon moyen de mettre à disposition de l'utilisateur une fenêtre proche de celles de Windows tout en gardant un maximum de portabilité (Linux). Malheureusement cette solution étant très difficile à mettre en œuvre, pour cause de documentation pratiquement inexistante, nous nous somme vu obligés de nous rabattre sur GUI Builder. Cet outil, distribué par Aonix en même temps que ObjectAda, utilise les librairie de Windows ce qui rend impossible toute portabilité, mais cela permet d'avoir recours, entre autre, au boite de dialogue de Windows (choix de couleur). Une fois encore cette interface a du être abandonnée car la documentation est très limitée.
Pour avoir déjà utilisé les librairies Spider, nous connaissions déjà sa simplicité mais aussi ses limites. Nous avons donc choisi d'abandonner l'interface graphique et de recentrer nos efforts sur les outils de navigation. Le nombre d'outils devenant vite important, il n'était plus possible de gérer le programme par un simple menu. Il nous a donc fallu opter pour un système avec ligne de commande.

Au final nous avons un programme divisé en deux fenêtres. La première est réservée pour le dessin de la fractale et la seconde est utilisée pour la saisie des multiples commandes.

2. Introduction

Une fractale est un dessin basé sur un calcul de nombres complexes. Ces dessins, sont non seulement particulièrement esthétiques, mais en plus ils offrent des caractéristiques intéressantes comme le fait de pouvoir zoomer à l'infini et de toujours retrouver la forme de base.

Le but de notre programme est donc de calculer et d'afficher un tel dessin puis de pouvoir naviguer dedans à volonté, mais il permet aussi de choisir une palette de couleurs ou encore d'exporter la fractale dans un fichier BMP, pour ,par exemple, l'utiliser comme fond d'écran.

3. Étapes lors du développement

3.1 Introduction

Le choix du langage de programmation a été très facilement résolu, puisque Ada95 est le seul langage que nous connaissons vraiment bien. Mais le développement a subi plusieurs fois des changements importants à cause de notre peu d'expérience en matière d'interfaçage avec Ada.

3.2 Conception de l'algorithme

Dans un premier temps nous avons testé l'algorithme de base en utilisant les librairies Spider qui sont facile à mettre en œuvre. Les résultats étant très concluant nous nous somme mis à la recherche d'une interface graphique pour l'utilisateur.

(Pour plus de détails sur l'algorithme, voir le document technique ci-joint à la section : 3.2)

3.3 Recherche d'une interface

3.3.1 GTK pour Ada

Profitant d'un développement parallèle qui nécessitait lui aussi une interface graphique, nous avons découvert GTK-Ada de Ada Core Technologies qui est un outil complet d'interfaçage. En effet, il contient non seulement toutes les librairies graphiques nécessaire, mais en plus nous avons à disposition un éditeur qui permet de créer un projet et de dessiner les fenêtres ainsi que tout son contenu (à la manière de VisualBasic). Et de plus GTK-Ada permet une portabilité vers Linux.

Après avoir créé les fenêtres et ses composants, GTK-Ada nous génère plusieurs fichiers de code Ada. Il ne reste plus qu'à écrire les procédures attribuées aux différents événements. Le problème que nous avons rencontré (autant nous que le groupe du projet développé en parallèle) est que le programme est fournit avec très peu de documentation et les exemples ne corresponde en rien au code généré.

Nous sommes tout de même arrivé à faire certaines choses, mais le gros problème venait de l'affichage. Malgré tous nos efforts de compréhensions, nous avons été incapable d'afficher quelque chose correctement et ensuite de gérer le buffer de façon à ne pas faire clignoter l'image sans cesse ou la redessiner lorsque celle-ci disparaissait (derrière une autre fenêtre) puis réapparaissait. Un programme de fractal ne peut pas travailler avec un affichage aussi désastreux. Nous avons donc décidé après maints efforts d'abandonner cette solution.

3.3.2 GUI Builder pour Object Ada

Après GTK-Ada nous nous sommes tournés vers GUI Builder. En effet, nos postes de travail étant récemment réinstallés à la suite de problèmes informatiques, nous disposons dorénavant de GUI Builder qui est distribué avec ObjectAda par Aonix. La différence avec ce nouvel outil est qu'il utilise les librairies de Windows, ce qui permet de créer une interface exactement comme les fenêtres habituelles de Windows et aussi d'utiliser les boite de dialogues (choix des couleurs). Le défaut étant évidemment que toute portabilité est désormais impossible.

GUI Builder inclue en plus un éditeur de code très sommaire, mais qui permet néanmoins d'écrire le code en même temps que la création de la fenêtre. Après un départ plutôt enthousiaste, nous sommes confronter une nouvelle fois au problème de l'affichage. GUI Builder manque lui aussi de documentation et notre apprentissage n'est pas aisé. En plus du fait que les procédures font appel aux librairies Win32, les différents paquetages se surchargent étrangement et nous n'avons pas réussi à les démêler. Bien que le buffer d'affichage soit bien géré automatiquement, nous n'arriverons jamais à dessiner une fractale.

3.3.3 Spider

Le temps jouant contre le temps et notre seul vrai problème étant jusqu'alors l'affichage, nous avons finalement opté pour Spider. Pour avoir déjà utilisé ces librairies nous connaissions sa simplicité, mais aussi ses limites (lenteur d'affichage). Nous avons dû abandonner l'idée de faire une interface graphique et nous résoudre à l'idée d'avoir une fenêtre de dessin et une deuxième pour les commandes.

Nous avons donc repris l'algorithme de test et nous avons enfin commencé le développement de tous les outils. Rapidement nous avons obtenu un programme permettant de choisir les couleurs, de zoomer, de centrer la fractal et de définir encore bien d'autres paramètres. Le système de menu à choix est alors devenu bien trop lourd à utiliser. Il nous a fallut trouver un autre moyen de gérer tous ces paramètres. La ligne de commande s'est avéré être la solution recherchée. En effet, elle permet de gérer un grand nombre d'actions et évite la lecture fastidieuse de long menu pour l'utilisateur. Le défaut étant que l'utilisateur doit connaître les commandes. C'est pourquoi une fonction d'aide a été implémentée.

3.4 Restructuration

Le projet grossissant à vue d'œil, il est devenu nécessaire de lui faire subir une restructuration. Le projet, jusque là en un seul fichier, a été sectionné en plusieurs paquetages, permettant ainsi un code plus clair et un peu plus grand confort de travail pour nous. Le système de base étant bien posé, nous avons pu ajouter de nouvelles fonctionnalités tel que la notion d'angle, la sauvegarde de fractal et l'exportation en BMP.
C'est en tout dernier que la notion de listes de fractal a été introduite avec toutes les fonctions supplémentaire qu'elle nécessite (sauver une liste, ajouter/effacer une fractal).

4. Conclusion

Le produit final s'annonce dans une première fenêtre puis la fenêtre de dessin s'ouvre et permet le choix d'une fractal dans la liste par défaut. Après cela l'utilisateur a accès à la console et donc à toutes les commandes disponibles (une trentaine). Il peut ainsi naviguer à volonté dans la fractale ou en choisir une autre dans la liste par défaut ou encore en créer une nouvelle. En jouant sur les nombreux paramètres, les possibilités de dessin sont absolument gigantesques, c'est pourquoi il pourra à tout moment sauver sa fractal (.JOF) ou la liste entière de fractales (.LOF) ou encore récupérer des anciennes compositions.

Le point faible de ce programme réside dans sa lenteur d'affichage dû au fait que la fractale est dessinée point par point et pas par le biais d'un buffer comme GTK-Ada aurait pu le faire. Un autre reproche à faire à notre choix de Spider est le fait de devoir jongler entre les deux fenêtres à tout moment.

5. Bibliographie

[1] « Quelques informations sur les fractales »
http://fractals.iuta.u-bordeaux.fr/jpl/jpl1.html