Guide pratique du jeu sous Linux

Version française du *Linux Gamers' HOWTO*

Peter Jay Salzman `<p CHEZ dirac POINT org>`

Frédéric Delanoy

Traduction française: Frédéric Delanoy

Version : 1.0.1.fr.0.9

9 juin 2004

Historique des versions

Version 1.0.1.fr.0.9	2004-06-09
Traduction de la version 1.0.1 du « Linux Gamers' HOWTO »	
Version 1.0.1	2004-02-04

Les mêmes questions reviennent continuellement sur les listes de diffusion et groupes de discussion Linux. Beaucoup d'entre elles sont dues à la méconnaissance du fonctionnement de Linux, du moins en ce qui concerne le jeu. Jouer peut être malaisé : cela requiert la connaissance d'une très grande étendue de sujets parmi lesquels on trouve les compilateurs, les bibliothèques, l'administration système, la gestion de réseaux, l'administration de XFree86, et cætera, vous voyez le genre. Chaque aspect de votre ordinateur joue un rôle dans le jeu. C'est un sujet exigeant, mais ce fait est occulté par le but principal du jeu : s'amuser et décompresser.

Ce document est un point de départ pour résoudre la plupart des problèmes courants et pour donner aux joueurs les connaissances nécessaires afin de réagir intelligemment en cas de problème avec leurs jeux. Comme de coutume sous Linux, vous devez connaître un peu ce qui se passe en coulisses pour que vos jeux continuent à fonctionner correctement ou pour poser un diagnostic et agir en conséquence dans le cas contraire.

Table des matières

1. Informations administratives

Si vous avez des idées, corrections ou questions relatives à ce guide, envoyez-moi un courriel. Le fait de recevoir du retour (même si je n'ai pas le temps de répondre) me donne l'impression que je fais quelque chose d'utile. Dès lors, cela me motive à écrire plus encore et à compléter ce document. Vous pouvez me contacter (NdT : en anglais) à l'adresse `<p CHEZ dirac POINT org>`. Ma page web est http://www.dirac.org/p et mes pages Linux sont situées sur http://www.dirac.org/linux. N'hésitez pas à envoyer vos commentaires et suggestions relatifs à ce guide. Même si je ne les retiens pas tous, votre apport est le bienvenu.

N'hésitez pas à faire parvenir tout commentaire relatif à la version française de ce document à `<commentaires CHEZ traduc POINT org>`.

Je présume une connaissance pratique de Linux, et j'utilise donc certains termes comme les niveaux d'exécution et les modules sans les définir. S'il y a assez de questions (ou même de protestations), j'ajouterai des informations plus basiques à ce document.

1.1. Droits d'utilisation

Important

Le texte ci-dessous est la version française de la licence de ce document. Seule la version originale de cette licence, présentée dans la section suivante, fait foi.

La version originale de ce document a été réalisée par Peter Jay Salzman `<p CHEZ dirac POINT org>` pour les années 2001-2002, et à Peter Jay Salzman et Frédéric Delanoy pour les années 2003-2004.

Vous avez le droit de copier, distribuer et modifier la version originale de ce document selon les termes de la *Open Software License* [http://opensource.org/licenses/osl-1.1.txt] (OSL), version 1.1, complétés par les dispositions présentées dans le paragraphe suivant. Je déteste les guides qui incluent la licence : des arbres meurent…

Si vous voulez créer un travail dérivé ou publier ce guide à des fins commerciales, je souhaiterais être contacté au préalable. Cela me donnera l'occasion de vous fournir la version la plus récente. J'apprécierais également soit une copie de votre travail, soit une pizza aux épinards, à l'ail, aux champignons, à la feta et aux cœurs d'artichauts.

La version française de ce document a été réalisée par Frédéric Delanoy. Elle est publiée en accord avec les termes de la *Open Software License*.

1.2. Authorship and Copyright

Important

Le texte ci-dessous est la licence de ce document. Ce texte fait foi. Il est composé de la licence en anglais du document orignal, suivi de la licence en français de sa traduction.

This document is copyright (c) 2001-2002 Peter Jay Salzman, `<p(at)dirac(dot)org>`; 2003-2004 Peter Jay Salzman and Frédéric Delanoy. Permission is granted to copy, distribute and/or modify this document under the terms of the Open Software License, Version 1.1, except for the provisions I list in the next paragraph. I hate HOWTO's that include the license; it's a tree killer. You can read the OSL at http://opensource.org/licenses/osl-1.1.txt.

If you want to create a derivative work or publish this HOWTO for commercial purposes, I would appreciate it if you contact me first. This will give me a chance to give you the most recent version. I'd also appreciate either a copy of whatever it is you're doing or a spinach, garlic, mushroom, feta cheese and artichoke heart pizza.

La version française de ce document a été réalisée par Frédéric Delanoy. Elle est publiée en accord avec les termes de la *Open Software License*.

1.3. Remerciements

Merci à Mike Phillips qui a énormément commenté ce guide. Merci à Dmitry Samoyloff, `<dsamoyloff CHEZ yandex POINT ru>`, qui a traduit ce document en russe. Cela m'a fait chaud au cœur quand il m'a raconté qu'il traduisait mes mots en russe. D'autres remerciements reviennent à :

1. Moritz Muehlenhoff `<jmm CHEZ Informatik POINT uni-bremen POINT de>` qui m'a envoyé des mises à jour (même si je suis éternellement à la traîne…)

2. Frédéric Delanoy pour d'importantes différences, corrections de fautes de frappe ou d'erreurs docbook.

Je voudrais également remercier Michael Mc Donnell pour m'avoir envoyé des commentaires et des corrections.

1.4. Dernières versions et traductions

La version la plus récente peut être trouvée sur http://cvs.sourceforge.net/cgi-bin/viewcvs.cgi/lgh/LG-HOWTO ou http://www.dirac.org/linux/writing, mais c'est ma copie de travail personnelle. La version présente sur mon site web personnel pourrait être corrompue si je travaille sur le guide. La version sur sourceforge est la plus récente mais est garantie non corrompue, même si elle peut comporter quelques petits problèmes, comme des paragraphes non terminés. :)

La version stable la plus récente peut être trouvée sur http://www.tldp.org.

Dmitry Samoyloff, `<dsamoyloff CHEZ yandex POINT ru>`, est responsable de la traduction russe de ce guide. La version la plus récente peut être trouvée sur http://www.dirac.org/linux/writing.

2. Définitions : types de jeux

Tout le monde ne connaît pas les différents types de jeux existants. Pour pouvoir parler un langage commun, je vais m'intéresser à chaque type de jeu et fournir un très bref historique.

2.1. Arcade

Bien que les jeux d'arcade ont connu leur heure de gloire dans les années 80, ils restent néanmoins très populaires. Rien ne remplacera jamais une promenade dans une galerie d'arcade sombre, bondée et bruyante, y glissant une pièce dans votre machine préférée et jouant à un vieux jeu du genre *Space Invaders*™. Les jeux de style arcade essaient de simuler les jeux d'arcade eux-mêmes. Il y en a tellement qu'il est quasi impossible de tous les énumérer, mais on peut citer les clones de *Asteroids*™, *Space Invaders*™, *Pac-Man*™, *Missile Command*™ et *Galaxian*™.

2.2. Jeux de cartes, de logique et de plateau

Les jeux de cartes simulent des jeux comme le poker ou le solitaire. Le programme peut simuler votre ou vos adversaire(s).

Les jeux de logique simulent habituellement quelque casse-tête logique bien connu comme *Master Mind*™ ou le taquin.

Les jeux de plateau simulent les jeux que vous pourriez jouer sur une table avec des amis, comme le

Monopoly™, les échecs, et cætera. Le programme peut simuler votre adversaire.

2.3. Aventure en mode texte (ou fiction interactive)

Il était une fois, quand Apple][, Commodore et Atari dominaient le monde, les aventures en mode texte étaient le jeu dans le vent des « gens éduqués ». On vous fournit un scénario et vous pouvez interagir avec le monde dans lequel vous vous trouvez :

```
You are in a room.  It is pitch dark and you're likely
to be eaten by a grue.
> Light lantern with match.
You light the lantern.  This room appears to be a kitchen.
There's a table with a book in the center.  You
also see an oven, refrigerator and a door leading east.
> Open the oven.
In the oven you see a brown paper bag.
> Take the bag.  Open the bag.  Close the oven.
Inside the bag is a clove of garlic and a cheese sandwich.
The oven door is now closed.
```

À cette époque, les aventures en mode texte étaient des exécutables autonomes tenant entièrement sur une disquette ou une cassette. Il y avait souvent un fichier de données et un interpréteur. L'interpréteur lit les fichiers de données et fournit l'interface de jeu. Les fichiers de données constituent le jeu en lui-même, à la manière de la relation entre les jeux de combat à la première personne et les fichiers `wad`.

Le premier jeu d'aventure était *Adventure*™ (en fait « *ADVENT*™ », écrit sur un PDP-1 en 1972). Vous pouvez toujours jouer à Adventure (en fait, à un descendant) : il est fourni avec les « jeux bsd » sur la plupart des distributions Linux. Les aventures en mode texte ont été popularisées par Scott Adams et ont atteint leur pic de popularité à la fin des années 80 avec les jeux de Infocom auxquels on peut également jouer sous Linux.

Comme les modes graphiques se sont développés et sont devenus plus puissants, les aventures en mode texte ont cédé la place aux aventures en mode graphique. La mort de la fiction interactive a plus ou moins coïncidé avec la faillite de Infocom.

2.4. Aventures graphiques

Les aventures graphiques sont, en fin de compte, des aventures en mode texte traitées aux anabolisants. Leur degré d'utilisation des graphiques varie fortement. Dans les années 80, ils n'étaient qu'un peu plus que des aventures en mode texte et affichaient un écran rempli de graphiques statiques. Quand vous ramassiez un objet, l'arrière-plan était repeint et l'objet disparaissait. L'exemple type pourrait être les prétendues « aventures en haute résolution » comme *The Wizard And The Princess*™. Plus tard, les aventures graphiques sophistiquées faisaient déambuler votre personnage à l'écran, et vous pouviez même utiliser une souris, mais l'interface restait purement textuelle.

Ensuite, il y eut les « aventures pilotées à la souris » qui n'avaient plus d'interface textuelle, et disposaient souvent de graphiques dynamiques, comme un chat se baladant dans la pièce pendant que vous décidiez de la prochaine action à effectuer. Dans ce type de jeux, vous pointez un objet (p.ex. un livre) et vous pouvez choisir une action à partir d'une liste déroulante de fonctions. Un genre d'aventure orientée objet. :) Il n'y a pas beaucoup d'aventures graphiques écrites nativement pour Linux. La seule dont je me rappelle est *Hopkins FBI*™ (qui est mon jeu préféré sous Linux).

2.5. Jeux de simulation

Les simulations s'efforcent d'immerger le joueur dans un milieu auquel il n'aurait normalement pas accès. Cela peut être quelque chose comme un avion de chasse ou quelque chose d'imaginaire comme une unité de combat mécanisée. Dans tous les cas, les simulations recherchent le réalisme.

Certaines simulations ne comportent que peu ou pas de stratégie. Elles vous placent simplement dans un cockpit pour vous donner la sensation de piloter un avion. Certaines sont extrêmement complexes, et la frontière est souvent ténue entre les simulations et les jeux de stratégie. Un bon exemple serait *Heavy Gear III*™ ou *Flight Gear*™. De nos jours, les jeux de simulation et de stratégie sont pratiquement indissociables mais, il y a longtemps, les simulations se déroulaient en temps réel alors que les jeux de stratégie se jouaient au tour à tour. Cette dénomination est maladroite de nos jours, car un jeu comme *Warcraft*™, que tout le monde considère être un jeu de stratégie, serait une simulation par définition.

2.6. Jeux de stratégie

Les jeux de stratégie descendent des vieux jeux de plateau de type *Avalon*™ comme *Panzer Leader*™ et des vieux jeux de stratégie militaire publiés par SSI. Généralement, ils simulent quelque type de scénario. Il peut être pacifique, comme la gestion réussie d'une ville (*SimCity*™), ou pas, comme la vente illégale de drogue (*DrugWars*™) ou un jeu de stratégie militaire acharné comme *Myth II*™. Ce type de jeux a habituellement une longue durée de vie et requiert beaucoup de réflexion.

Les jeux de stratégie peuvent être à leur tour subdivisés en deux classes : le temps réel et le tour à tour. Les jeux de stratégie en temps réel sont basés sur le principe « un petit moment d'inattention, et c'est perdu ». Par exemple, vous gérez une ville et un incendie se déclenche quelque part. Au plus il vous faut du temps pour mobiliser les pompiers, au plus l'incendie fera du dégât. Les jeux de stratégie au tour à tour sont similaires aux échecs : l'ordinateur joue, puis c'est au tour du joueur, et cætera.

2.7. Jeux de combat à la première personne (First Person Shooter, FPS)

Quelle est cette lumière au loin ? Cela doit être l'éclair du fusil de chasse à double canon ! L'histoire des jeux FPS est longue et tortueuse et a commencé quand id Software a ouvert le code source de *Doom*™. Ce code a été repris (par d'autres équipes) et fusionné à de nombreuses reprises. D'autres moteurs précédemment fermés se sont ouverts, beaucoup de moteurs sont jouables via des émulateurs, beaucoup de jeux FPS commerciaux ont été publiés pour Linux et nombre de moteurs FPS étaient des projets *open source* à l'origine. Bien qu'il se peut que vous ne puissiez pas jouer à votre jeu FPS *préféré* sous Linux (*Half-Life*™ marche très bien sous winex), Linux ne présente certainement aucune lacune dans ce domaine !

Les FPS sont caractérisés par deux choses. Primo, vous tirez à peu près sur tout ce qui bouge. Secundo, l'action a lieu à la première personne, c.-à-d. au travers des yeux du personnage que vous interprétez. Vous pouvez même voir vos mains ou votre arme en bas de l'écran. Ils peuvent être du type fantastique (*Hexen*™), science-fiction (*Quake II*™), réaliste (*Soldier Of Fortune*™) parmi bien d'autres possibilités.

Tout comme les aventures en mode texte, les FPS obéissent au format moteur/fichier de données. Le moteur représente le jeu en lui-même (*Doom*™, *Quake*™, *Heretic2*™) et utilise les niveaux et les méchants décrits par les fichiers de données (`doom2.wad`, `pak0.pak`, etc). Beaucoup de jeux FPS permettent de créer ses propres fichiers de données non commerciaux. Il y a des centaines, si pas des milliers, de fichiers de données pour *Doom*™ que vous pouvez télécharger gratuitement depuis Internet. Souvent, les sociétés libèrent leurs moteurs afin que la communauté *open source* puisse les améliorer. Néanmoins, les fichiers de données originaux restent propriétaires. À ce jour, vous devez toujours acheter `doom.wad`.

2.8. Jeux à défilement horizontal

Les jeux à défilement horizontal sont similaires aux FPS mais vous voyez votre personnage comme une figure 2D qui court sur plusieurs écrans en tirant ou en effectuant d'autres tâches. Par exemple, il y a *Abuse*™ pour Linux et le *Duke Nukem*™ original. Ils ne sont pas nécessairement violents, comme *xscavenger*™, un clone du vieux jeu 8 bits *Lode Runner*™.

2.9. Jeux de combat à la troisième personne

Similaires aux FPS, mais vous voyez votre personnage à la troisième personne et en 3D. Dans les jeux de combat à la troisième personne modernes, vous pouvez en général réaliser des manœuvres dévastatrices comme les retournés acrobatiques et les roulades à la Jackie Chan. L'exemple type serait *Tomb Raider*™. Sous Linux, on dispose de *Heretic 2*™ et de *Heavy Metal FAKK2*™.

2.10. Jeux de rôle (Role Playing Game, RPG)

Quiconque a déjà joué à des jeux du style *Dungeons & Dragons*™ ou *Call of Cthulhu*™ sait exactement ce qu'est un RPG. Vous interprétez un personnage (parfois plusieurs) doté de certaines caractéristiques (p.ex. force, dextérité), d'aptitudes (p.ex. explosifs, confection de paniers, mécanique) et de propriétés (niveaux, argent liquide). Au fil du temps, le personnage devient plus puissant et le jeu s'adapte en conséquence : p.ex., au lieu de combattre des orcs, vous commencez à combattre des dragons noirs dans les plus hauts niveaux. Les récompenses augmentent en conséquence. Dans les bas niveaux, vous pourriez obtenir quelques pièces d'or comme récompense suite à la victoire dans une bataille ; plus loin, vous pourriez obtenir une épée magique ou un fusil d'assaut ravageur.

Les RPG proposent généralement une quête ayant un épilogue bien déterminé. Dans *nethack*™, vous devez retrouver l'amulette de Yendor pour votre dieu. Dans *Ultima II*™, vous détruisez la sorcière maléfique Minax. À un certain moment, votre personnage devient suffisamment puissant pour pouvoir vous lancer et essayer de terminer la quête.

Bien que la série immensément populaire des *Ultima*™, écrite par Richard Garriot (alias Lord British) pour Origin, n'était pas le premier RPG, elle a popularisé et propulsé le genre RPG sur le devant de la scène. *Ultima I*™ a été publié en 1987 et est le point de départ de 9 (en fonction du mode de comptage) suites très populaires, se terminant par *Ultima IX: Ascension*™. Vous pouvez jouer à *Ultima VII*™ sous Linux avec Exult.

Le jeu RPG type sous Linux est *Rogue*™ (la bibliothèque ncurses était à l'origine une routine de traitement d'écran pour *Rogue*™ !) et on en décline des tas de variantes comme *Zangband*™ et *Nethack*™ (qui a lui-même de nombreuses variantes). Certains RPG sont assez compliqués et constituent de véritables exploits de programmation. Il semble y avoir une carence en RPG commerciaux sous Linux. Si l'on ne compte pas les variantes de *Rogue*™, c'est également le cas des RPG *open source*.

3. Bibliothèques

Voici différentes bibliothèques consacrées au jeu que l'on peut trouver sous Linux.

3.1. Glide2

Glide2 est une API et un pilote graphiques de bas niveau qui accèdent aux fonctions 3D accélérées par matériel des cartes Voodoo I, II et III de 3dfx sous XFree86 3.x.

Un programme ne peut utiliser les fonctionnalités spéciales accélérées matériellement de ces cartes qu'en utilisant la bibliothèque Glide2 d'une des deux façons suivantes :

- directement en utilisant Glide2 (*Myth II*™, *Descent III*™)
- indirectement en utilisant Mesa construit avec un dorsal Glide2 pour simuler OpenGL (*Rune*™, *Unreal Tournament*™)

3dfx a ouvert les spécifications et le code source à la communauté *open source*. Cela a permis à Daryll Strauss de porter Glide2 sous Linux, autorisant les utilisateurs de XFree86 3.x à utiliser les cartes Voodoo I, II et III sous Linux.

Puisque Glide2 accède directement à la carte vidéo, les applications Glide2 doivent soit être exécutées par root, soit être setuid-root. Une façon d'éviter cela était de créer le module noyau 3dfx. Ce module (et son fichier de périphérique `/dev/3dfx`) permet l'accélération graphique matérielle Glide2 pour les utilisateurs non-root d'applications non-setuid.

Malheureusement, Glide2 n'est pas une solution d'avenir. Elle n'est utilisée que pour les cartes Voodoo I, II et III (qui deviennent obsolètes), sous XFree86 3.x (la majorité utilise XFree86 4.x). Et étant donné que 3dfx est maintenant une société défunte, il est certain qu'aucun développement n'aura désormais lieu sur Glide2 et qu'aucun nouveau jeu ne sera écrit en utilisant Glide2.

3.2. Glide3

À la différence de Glide2, Glide3 n'est pas une API utilisée pour la programmation de jeux. Elle n'existe que pour gérer le DRI pour les cartes Voodoo III, IV et V sous XFree86 4.x. Aucun des jeux utilisant Glide2 ne fonctionnera avec Glide3. Cela ne devrait pas être surprenant dans la mesure où Glide2 et Glide3 prennent en charge des cartes vidéo différentes et des versions de XFree86 différentes. La seule carte vidéo pouvant utiliser à la fois Glide2 (sous XFree86 3.x) et Glide3 (sous XFree86 4.x) est la Voodoo III. On rapporte qu'une Voodoo III utilisant Glide2 surpasse une Voodoo III utilisant Glide3.

Quand vous utilisez une Voodoo III, IV ou V sous XFree86 4.x, vous devez utiliser une version de Mesa qui a été compilée pour utiliser Glide3 comme dorsal afin de pouvoir utiliser l'accélération matérielle OpenGL sur votre système.

3.3. OpenGL

OpenGL est une interface de programmation graphique de haut niveau développée à l'origine par SGI, et qui est devenue un standard industriel pour la programmation 2D et 3D. Elle est définie et soutenue par le Architectural Revision Board (ARB), une organisation qui inclut des représentants de SGI, IBM, DEC et Microsoft. OpenGL fournit un jeu de fonctionnalités puissant, complet et générique pour les opérations graphiques 2D et 3D.

OpenGL est constitué de 3 parties :

- GL : les appels OpenGL de base
- GLU : les appels utilitaires
- GLUT : le traitement des événements de fenêtre indépendants du système (événements de souris, du clavier, et cætera).

OpenGL n'est pas seulement une API, c'est aussi une implémentation, écrite par SGI. Elle essaie d'utiliser l'accélération matérielle pour diverses opérations graphiques quand elle est disponible, en fonction de de la carte vidéo dont vous disposez. Si l'accélération matérielle n'est pas possible pour une tâche particulière, OpenGL retombe sur le rendu logiciel. Cela signifie que si vous vous procurez OpenGL chez SGI, et que vous voulez disposer d'une accélération matérielle quelconque, elle doit être écrite en OpenGL et compilée spécifiquement pour une certaine carte graphique. Sinon, vous retomberez sur le rendu logiciel. Cela s'applique également aux clones d'OpenGL, comme Mesa.

OpenGL est l'équivalent *open source* de Direct3D, un composant de DirectX. Une différence importante est que puisque OpenGL est ouvert (et DirectX fermé), les jeux écrits en OpenGL sont beaucoup plus faciles à porter et à co-développer sous Linux que ne le sont les jeux utilisant DirectX.

3.4. Mesa

Mesa <http://www.mesa3d.org> est une implémentation libre de l'API OpenGL, conçue et écrite par Brian Paul. Bien qu'elle ne soit pas officiellement certifiée (cela nécessiterait plus d'argent que n'en dispose un projet *open source*), elle constitue une implémentation d'OpenGL presque totalement conforme aux spécifications de l'ARB. On rapporte que Mesa est même plus rapide que la propre implémentation OpenGL de SGI.

Tout comme OpenGL, Mesa utilise l'accélération matérielle quand c'est possible. Quand une tâche graphique particulière ne peut être accélérée matériellement par la carte vidéo, elle est rendue en logiciel ; la tâche est alors accomplie par votre processeur. Cela signifie qu'il existe différentes moutures de Mesa en fonction du type de carte vidéo dont vous disposez. Chaque mouture utilise une bibliothèque différente comme moteur de rendu. Par exemple, si vous avez une Voodoo I, II ou III sous XFree86 3.x, vous devriez utiliser mesa+glide2 (écrit par David Bucciarelli) qui est l'implémentation Mesa de OpenGL qui utilise Glide2 comme dorsal pour rendre les opérations graphiques.

3.5. DRI

Le rendu des graphiques comporte 3 protagonistes : l'application cliente (comme *Quake 3*™), le serveur X et le matériel (la carte graphique). Auparavant, les applications clientes ne pouvaient pas écrire directement sur le matériel, et il y avait une bonne raison à cela : un programme à qui l'on permet un accès en écriture direct sur le matériel peut faire planter le système de plusieurs façons. Plutôt que de faire confiance aux programmeurs pour écrire des programmes (accédant au matériel) totalement exempts de bogues et coopératifs, Linux l'a tout simplement interdit. Néanmoins, cela a changé sous XFree 4.x avec l'infrastructure de rendu direct (Direct Rendering Infrastructure [http://www.dri.sourceforge.net], DRI). La DRI autorise les clients X à écrire des informations de rendu 3D directement sur la carte vidéo d'une manière sûre et coopérative.

DRI fait abstraction du serveur X afin que le pilote 3D (Mesa ou OpenGL) puisse parler directement au matériel. Cela améliore les performances. Les informations de rendu 3D ne doivent même pas subir d'accélération matérielle. D'un point de vue technique, cela a plusieurs avantages :

- Les données associées aux sommets de polygones ne doivent pas être codées/décodées via GLX.
- Les données graphiques ne sont pas envoyées via une socket au serveur X.
- Sur les machines mono-processeur, le CPU ne doit pas changer de contexte entre XFree86 et son client pour rendre les graphiques.

3.6. GLX

GLX est l'extension X utilisée par les programmes OpenGL ; c'est le liant entre l'OpenGL indépendant de la plate-forme, et X dépendant de la plate-forme.

3.7. Utah GLX

Utah-GLX est le précurseur de DRI. Certaines décisions de conception sont différentes en ce qui concerne la séparation des données et des méthodes d'accès à la carte vidéo, comme le repos sur l'accès root plutôt que la création de l'infrastructure noyau permettant un accès sécurisé. Il prend en charge quelques cartes qui ne sont pas bien gérées par le DRI comme la famille ATI Rage Pro, la S3 Virge (bien que quiconque l'utilise pour jouer est pour ainsi dire cinglé), et un pilote TNT/TNT2 *open source* (très incomplet). Le pilote TNT/TNT2 est basé sur la rétro-ingénierie de la publication du code source obscurci des pilotes X 3.3 par nVidia. Néanmoins, ils sont très incomplets et, pour tout dire, inutilisables.

3.8. xlib

De temps à autre, vous verrez quelques malades (dit avec respect) qui écrivent un jeu en xlib. C'est un groupe de bibliothèques C qui comportent l'interface de programmation du plus bas niveau pour XFree86. Toute programmation graphique sous X fait *in fine* usage de la bibliothèque xlib.

Il n'est pas exagéré de dire que xlib est volumineux, mystérieux et compliqué. De ce fait, il existe des tas de bibliothèques comme SDL pour les graphiques 2D, et OpenGL pour les graphiques 3D et les jeux d'éléments graphiques (*widgets*) pour les éléments graphiques à l'intérieur des fenêtres qui cachent les détails de différents aspects de la programmation xlib.

Bien que quelques jeux soient écrits avec xlib, comme l'éditeur Doom Yadex, xlib en lui-même ne peut pas raisonnablement servir de bibliothèque d'écriture de jeux. La plupart des jeux n'ont pas besoin de l'interface de bas niveau fournie par xlib. De plus, en utilisant les bibliothèques de plus haut niveau, un programmeur de jeux peut développer son jeu sur plusieurs plates-formes, même celles qui n'utilisent pas XFree86.

3.9. *Widgets*

Les éléments graphiques (widgets) sont des objets qui constituent l'interface d'une application graphique. Ils incluent des choses comme les boîtes d'entrée de texte, les menus déroulants, les barres de défilement, les boutons radio et bien d'autres choses. Un jeu d'éléments graphiques (widget set) est une collection d'éléments graphiques apparentés qui sont conçus pour avoir une interface commune et un aspect cohérent. Gtk est le jeu d'éléments graphiques canonique sous Linux, mais il y en a beaucoup d'autres comme fltk (de petite taille, écrit en C++), Xaw, Qt (le jeu d'éléments graphiques de KDE) et Motif (celui utilisé par Netscape). Motif régnait dans le monde Unix, mais sa licence d'utilisation était très coûteuse. L'Open Group a finalement ouvert la licence de Motif pour les systèmes d'exploitation *open source*, mais c'était trop tard. Il y a beaucoup de jeux d'éléments graphiques complètement *open source* qui sont plus complets et plus beaux que Motif, y compris Lesstif, un clone totalement gratuit de Motif.

3.10. SDL (Simple DirectMedia Layer)

SDL [http://www.libsdl.org] est une bibliothèque de Sam Lantiga (diplômé de l'UCD !). C'est en fait une méta-bibliothèque, c.-à-d. que ce n'est pas seulement une bibliothèque graphique qui cache les détails de la programmation xlib, mais c'est aussi une interface simple d'utilisation pour le traitement du son, de la musique et des événements. Sa licence est la LGPL et elle prend également en charge les joysticks et OpenGL. À la différence de xlib, SDL convient fort bien à la programmation de jeux.

Le plus impressionnant dans SDL est son caractère multi-plates-formes. Mis à part quelques détails, un programme écrit en SDL compilera sous Linux, MS Windows, BeOS, MacOS, MacOS X, Solaris, IRIX, FreeBSD, QNX et OSF. Il existe diverses extensions permettant de manipuler à peu près tous les formats graphiques, lire des vidéos MPEG, afficher des polices *truetype*, gérer les acteurs (*sprites*) et à peu près tout ce qui est imaginable. SDL est un exemple de ce à quoi toutes les bibliothèques graphiques devraient aspirer.

Sam avait une motivation cachée pour l'écriture d'une si chouette bibliothèque : il était le programmeur en chef de Loki Software (il code maintenant pour Blizzard Software), qui utilisait SDL dans tous ses jeux sauf *Quake3*™.

3.11. GGI

GGI [http://www.ggi-project.org] est un projet qui vise à implémenter une couche d'abstraction graphique dans du code de bas niveau, de placer la prise en charge du matériel graphique dans une base de code commune, et d'apporter une plus grande stabilité et portabilité aux applications graphiques. Les applications LibGGI tournent entre autres sous SVGAlib, fb et X. Si l'on en juge à leurs captures d'écran, c'est une bibliothèque assez puissante.

Les applications qui utilisent LibGGI directement comportent *Heroes*™, *Ultrapoint*™, *Quake*™ et *Berlin*™. La plupart des applications qui utilisent SVGALib peuvent être exécutées sous X ou sous n'importe quel autre dorsal LibGGI en utilisant une bibliothèque enveloppe qui réimplémente SVGALib en utilisant LibGGI. Les applications SDL et clanlib peuvent s'afficher avec LibGGI mais les pilotes natifs de ces bibliothèques sont généralement plus rapides ; néanmoins, c'est un bon moyen pour que des applications SDL, clanlib et SVGALib s'exécutent là où elles n'auraient pas pu le faire auparavant.

GGI a un projet sœur, KGI, qui développe une alternative de niveau noyau aux systèmes du type *framebuffer* linux et DRI. Ce projet est beaucoup moins avancé que LibGGI lui-même, mais promet de combiner les vitesses de niveau DRI à la stabilité et à la sécurité auxquelles aspirent les utilisateurs UNIX.

3.12. SVGAlib, framebuffer et console

La console est l'écran noir non graphique que vous voyez lorsque votre ordinateur démarre pour la première fois (et qu'aucune application du genre xdm ou gdm ne tourne). C'est différent de l'environnement X qui comporte toutes sortes d'éléments graphiques comme les xterm. Une idée fausse fort répandue est de croire que X signifie « graphique » et que console signifie « non graphique ». Il peut assurément y avoir des graphiques en mode console ; nous discuterons des deux manières les plus habituelles de procéder.

SVGAlib est une bibliothèque graphique qui vous permet de dessiner des graphiques sur la console. Il existe beaucoup d'applications graphiques et de jeux utilisant SVGAlib comme zgv (un visualisateur d'images en mode console), prboom et hhexen. J'apprécie cette bibliothèque et les jeux graphiques en mode console en général : ils sont extrêmement rapides, plein écran et captivants. SVGAlib souffre de trois défauts. Primo, les exécutables SVGAlib doivent être lancés par root ou être setuid-root (néanmoins, la bibliothèque abandonne les privilèges root immédiatement après le début de l'exécution). Secundo, SVGAlib est dépendant de la carte vidéo : si votre carte vidéo n'est pas prise en charge par SVGAlib, c'est pas de chance. Tertio, SVGAlib est spécifique à Linux : les jeux écrits en SVGAlib ne fonctionneront *que* sous Linux.

Les *framebuffers* sont des consoles implémentées par un mode graphique plutôt qu'un mode texte du BIOS. Pourquoi simuler un mode texte dans un environnement graphique ? Cela permet d'exécuter des applications graphiques en console, comme p.ex. de choisir la police affichée en console (qui est normalement fixée par le BIOS). On peut trouver un bon « Guide pratique du frame-buffer » (*Framebuffer-HOWTO* [http://www.traduc.org/docs/howto/lecture/Framebuffer-HOWTO.html]) sur le LDP. Les jeux en console graphique écrits en utilisant le framebuffer souffrent des mêmes problèmes que ceux utilisant SVGAlib : le support matériel est limité, et le code ne fonctionnera que sous Linux.

3.13. OpenAL

OpenAL [http://www.openal.org] a pour objectif d'être au son ce que OpenGL est aux graphiques. Développé conjointement par Loki Software et Creative Labs, elle a pour but d'être une API neutre et multi-plates-formes pour le son. Sa licence est la LGPL et les spécifications peuvent être obtenues gratuitement depuis le site web de OpenAL. OpenAL est entièrement fonctionnel, mais depuis que Loki Software n'existe plus, son développement futur est incertain.

3.14. DirectX

DirectX est une collection d'API multimédia propriétaires, développée à l'origine par Microsoft en 1995, pour ses différents systèmes d'exploitation Windows. C'est une erreur de prétendre que « DirectX est similaire à OpenGL » ou « DirectX est similaire à SDL », comme il est souvent dit dans les didacticiels DirectX. Les API multimédia sont plus centralisées sous Windows qu'elles ne le sont sous Linux. Une formulation plus précise serait : « DirectX est similaire à DRI, OpenGL et SDL combinés ». En juin 2003, la version la plus récente de DirectX était la 9.0. Les composants de DirectX sont :

DirectDraw
: DirectDraw fournit un accès direct à la mémoire vidéo, comme DRI, de sorte que les graphiques 2D peuvent être placés directement sur la carte vidéo. DirectDraw est similaire au composant graphique de SDL, mais l'accès direct à la carte vidéo est effectué par DRI plutôt que par SDL. C'est pourquoi un jeu peut facilement faire tomber un système Windows mais ne devrait pas le faire avec un système Linux.

Direct3D (D3D)
: Direct3D, comme OpenGL, fournit une API graphique 3D. Alors qu'OpenGL est *open source*, de plus bas niveau et compile sous une multitude de systèmes d'exploitation, D3D est propriétaire, de plus haut niveau et ne compile que sous Windows. D3D est d'abord apparu dans DirectX 2, en 1996.

DirectXAudio
: Direct Audio est une combinaison de deux API audio, DirectSound et DirectMusic, qui offrent un accès direct à la carte son pour jouer du son et de la

	musique.
DirectInput	DirectInput permet l'utilisation de périphériques d'entrée de jeu comme les joysticks.
DirectPlay	DirectPlay offre une gestion réseau simplifiée pour les jeux multi-joueurs.
DirectShow	DirectShow prend en charge les fichiers vidéo comme `AVI` et `MPG`. C'était une API distincte de DirectX, mais elle a été intégrée dans DirectX 8.
DirectSetup	Cette API facilite l'installation de DirectX à partir d'une application pour simplifier l'installation des jeux.

DirectX est un peu pris en charge par winex, l'est mal par wine, l'est à peine par vmware et ne l'est pas du tout par Win4Lin.

Remarque sur la portabilité : pour chaque composant de DirectX, on peut trouver plusieurs bibliothèques correspondantes sous Linux. Mieux encore, un programmeur de jeux qui utilise des bibliothèques comme OpenGL, GGI ou SDL écrira un jeu qui compilera trivialement sous Windows, Linux et une multitude d'autres systèmes d'exploitation. Pourtant, les sociétés productrices de jeux persistent à utiliser DirectX et limitent de ce fait leur public aux seuls utilisateurs Windows. Si vous écrivez des jeux, veuillez envisager l'utilisation de bibliothèques multi-plates-formes et rester éloigné de DirectX.

Une société nommée realtechVR a démarré un projet *open source*, DirectX Port [http://www.v3x.net/directx] qui, comme wine, fournit une couche d'émulation de Direct3D qui implémente les appels Direct3D. Le projet se concentrait sur la plate-forme BeOS, mais l'est maintenant sur MacOS et Linux. Vous pouvez récupérer la toute dernière mouture depuis leur référentiel CVS sur <http://sourceforge.net/projects/dxglwrap>.

3.15. Clanlib

ClanLib est un kit d'outils de développement de niveau intermédiaire. Au plus bas niveau, il fournit des outils indépendants de la plate-forme (dans la limite du possible en C++) de gestion de l'affichage, du son, des entrées, du réseau, des fichiers, des threads, et cætera. ClanLib construit un cadre générique de développement de jeu, vous offrant une gestion aisée des ressources, une réplication des objets sur le réseau, des interfaces utilisateur graphiques (GUI) autorisant les thèmes, les langages de scripts dans les jeux et plus encore.

4. XFree86 et vous

Si vous avez l'intention de jouer sous X, il est primordial que vous le connaissiez quelque peu. Le « Guide pratique de l'utilisateur de X Window » (*XWindow-User-HOWTO* [http://www.traduc.org/docs/howto/lecture/XWindow-User-HOWTO.html]), et en particulier **`man XF86Config`** constituent des lectures *requises*. N'essayez pas d'y échapper : lisez-les. Elles ont un très bon rapport signal/bruit. Beaucoup de problèmes peuvent être résolus facilement si vous savez vous y retrouver dans `XF86Config` (ou `XF86Config-4`).

4.1. Recueillir des informations sur votre système X

Que vous essayiez de diagnostiquer un problème X ou que vous requerriez de l'aide sur une liste de diffusion ou un groupe de discussion Usenet, vous devrez disposer d'un maximum d'informations. Voici quelques outils qui peuvent vous y aider :

4.1.1. probeonly

La sortie de **probeonly** constitue l'un des meilleurs outils de diagnostic et l'une des meilleures sources d'informations sur votre système. Pour l'utiliser, arrêtez X le cas échéant et tapez depuis une console :

```
X -probeonly 2> X.out
```

La sortie de X va sur stderr, et il faut donc rediriger stderr avec « 2> » dans un fichier nommé `X.out`. Ce fichier contiendra tout ce qu'il y a à savoir sur votre système X. Il est crucial que vous connaissiez la différence entre les différentes marques que vous pourrez rencontrer dans la sortie de **probeonly** :

```
(--) probed              (**) from config file    (==) default setting
(++) from command line   (!!) notice              (II) informational
(WW) warning             (EE) error               (??) unknown.
```

Voici un exemple de quelques informations que j'ai pu glaner :

J'utilise des couleurs 16 bits :

```
(**) TDFX(0): Depth 16, (--) framebuffer bpp 16
```

X a détecté que la puce et la mémoire RAM de ma carte vidéo sont :

```
(--) Chipset 3dfx Voodoo5 found
(--) TDFX(0): VideoRAM: 32768 kByte Mapping 65536 kByte
```

4.1.2. Obtenir des informations sur votre configuration : xvidtune

xvidtune est votre ami si votre écran X est un peu trop décalé sur la droite, ou si la hauteur est trop réduite pour remplir votre écran. Néanmoins, c'est également un très bon outil de diagnostic. Il affiche :

- l'intervalle hsync/vsync (valeurs de synchronisation horizontale et verticale respectivement) spécifié dans votre fichier `XF86Config`.
- les 4 nombres horizontaux et les 4 nombres verticaux qui définissent votre mode vidéo (le premier couple horizontal/vertical indique la résolution de l'écran). Ces 8 nombres vous indiqueront quelle ligne de mode (*modeline*) votre X utilise. Voyez le « Guide pratique de configuration vidéo de XFree86 » (*XFree86-Video-Timings-HOWTO* [http://www.traduc.org/docs/howto/lecture/XWindow-User-HOWTO.html]) pour plus d'informations. Notez que des spécifications explicites ne sont plus nécessaires, car XFree 4.0.1 (et les versions ultérieures) les calcule automatiquement à partir des possibilités de votre moniteur et de votre carte vidéo. Néanmoins, c'est parfois utile en cas de matériel exotique ou si vous voulez un peu bidouiller votre affichage.
- La « fréquence d'horloge » à laquelle tourne votre carte vidéo.

4.1.3. Obtenir des informations sur votre configuration : xwininfo

xwininfo vous indique toutes sortes d'informations sur les fenêtres X. L'arrière-plan est également assimilé à une fenêtre. Ainsi, quand il vous demande de cliquer sur la fenêtre pour laquelle vous désirez des informations, cliquez sur votre arrière-plan. Il vous indique entre autres la résolution de l'écran et de la fenêtre, le nombre de couleurs (NdT : plus précisément, le nombre de bits utilisés pour les représenter), l'état de gravité de la fenêtre (qui donne une indication au gestionnaire de fenêtres sur l'endroit où placer les nouvelles fenêtres) et l'utilisation du cache d'affichage (*backing store*).

4.1.4. Autres sources d'information

xdpyinfo vous donne des informations intéressantes, comme la version de X et les extensions chargées (inestimable quand vous essayez de voir ce qui manque, comme GLX, DRI, XFree86-VidMode, et cætera).

4.1.5. Obtenir des informations sur votre système 3D

glxinfo donne des tas d'informations utiles sur OpenGL comme l'utilisation ou non du rendu direct, les versions de glx et mesa actuellement installées, les chaînes de vendeur/moteur de rendu, les fichiers de bibliothèque GL utilisés, et cætera.

4.2. Jouer à des jeux sous X sans gestionnaire de fenêtres

Quand vous jouez à un jeu sous X, vous devriez envisager de ne pas utiliser de gestionnaire de fenêtres. Des « poids lourds » comme enlightenment produiront un ralentissement perceptible ; même des plus légers comme twm vous volent des cycles processeur (et dans le cas de twm, même les jeux plein écran auront un cadre autour de leur fenêtre). Pour exécuter un jeu sans gestionnaire de fenêtres, modifiez `.xinitrc` dans votre répertoire personnel. Voici à quoi ressemble mon `.xinitrc` :

```
        #quake3 +set r_gldriver libGR.so.1
        #exec ut
        #lsdldoom -server 2
        #exec tribes2
        exec /usr/bin/enlightenment
```

Vous y remarquerez généralement un gestionnaire de fenêtres et/ou de bureau (GNOME ou KDE). Commentez les lignes contenant un gestionnaire de fenêtres et/ou de bureau avec un signe dièse (#) et placez votre jeu sur une nouvelle ligne accompagné des arguments de ligne de commandes éventuels. Si votre jeu n'est pas situé dans votre `$PATH`, donnez le nom de chemin complet. Notez que ceci ne s'applique qu'aux personnes utilisant **startx** pour démarrer X.

Je n'utilise jamais des choses comme **gdm** ou le niveau d'exécution 5 (je ne suis donc pas sûr de mon fait), mais je suspecte que si vous le faites, vous devrez agir un peu différemment. Ma recommandation est d'aller en mode mono-utilisateur (niveau d'exécution 1) avec :

```
    # telinit 1
```

ensuite d'éditer `.xinitrc`, et de revenir au niveau 5 avec

```
      # telinit 5
```

Ensuite, quand vous arrêtez de jouer, allez au niveau 1, modifiez `.xinitrc` et revenez au niveau 5. Je n'ai jamais expérimenté cela, et je ne peut donc pas me prononcer avec certitude, mais vous pourriez devoir tuer gdm. J'apprécierai du retour à ce sujet.

5. Divers

5.1. Registres d'intervalles mémoire

À partir des processeurs de classe Pentium (y compris Athlon, K6-2 et d'autres CPU), ces registres (Memory Type Range Registers, MTRR) contrôlent la façon dont le processeur accède aux intervalles d'adresses mémoire. Pour résumer, ce mécanisme remplace plusieurs petites écritures séparées sur la carte vidéo par une seule écriture (une rafale). Cela améliore l'efficacité des écritures sur la carte vidéo et peut accélérer le rendu graphique de 250 % voire plus !

Voyez `/usr/src/linux/Documentation/mtrr.txt` pour les détails. Notez que, depuis que ce fichier a été écrit, XFree86 a été amélioré pour détecter automatiquement l'adresse de base et la taille de votre RAM vidéo et configurer les MTRR.

5.2. Exploiter au maximum les ressources de votre système

- Si pour une raison quelconque, vous utilisez X 3.3, suivez les instructions données dans `mtrr.txt` (voyez la Section 5.1, « Registres d'intervalles mémoire ») pour configurer les MTRR. X 4.0 fait cela automatiquement pour vous.
- Si vous jouez sous X, n'exécutez pas de gestionnaire de fenêtres, et *certainement pas* de gestionnaire de bureau comme GNOME ou KDE. Voyez la Section 4.2, « Jouer à des jeux sous X sans gestionnaire de fenêtres » pour plus de détails.

  Arrêtez tous les processus non essentiels (en tant que root) en utilisant les scripts de démarrage de votre système. Sous Debian, les scripts de démarrage pour le niveau d'exécution 2 sont situés dans `/etc/rc2.d/`. Vous pouvez arrêter un service d'une manière ordonnée en envoyant à son script de démarrage la commande « stop » :

  ```
  # cd /etc/rc2.d
  # ./ntpd stop
  ```

  Une autre possibilité (radicale) est de simplement vous placer en mode mono-utilisateur avec

  ```
  # telinit 1
  ```

  Cela vous débarrassera même de **getty** ; votre système s'exécutera avec uniquement ce qui est absolument crucial pour son fonctionnement. Il y aura quelque chose comme 10 processus en cours d'exécution. L'inconvénient est que vous devez jouer en tant que root. Mais votre table de processus sera une ville fantôme, et tous les cycles CPU supplémentaires bénéficieront à votre jeu.

5.3. À propos des bibliothèques sous Linux

Un problème souvent rencontré est un fichier de bibliothèque non trouvé. Ils sont quelque peu mystérieux et ont des noms bizarres ; nous en parlerons donc un peu. Il y a deux types de bibliothèques, les statiques et les dynamiques. Quand vous compilez un programme, **gcc** utilise par défaut les bibliothèques dynamiques, mais vous pouvez lui faire utiliser des bibliothèques statiques en utilisant l'option `-static`. À moins que vous n'ayez l'intention de compiler des jeux à partir du code source, vous serez principalement intéressés par les bibliothèques dynamiques.

5.3.1. Bibliothèques dynamiques

Les bibliothèques dynamiques, aussi appelées « bibliothèques partagées », fournissent du code objet à une application alors qu'elle s'exécute, c.-à-d. que le code est lié à l'exécutable au moment de l'exécution, et non à celui de la compilation. Elles sont analogues aux `.dll` utilisées sous Windows. Le programme responsable de la liaison du code « au vol » est appelé **/etc/ld.so**, et les bibliothèques dynamiques elles-mêmes se terminent habituellement par `.so` avec un numéro de version, comme :

```
/usr/lib/libSDL.so
/lib/libm.so.3
```

Quand vous utilisez **gcc**, vous référencez ces bibliothèques en enlevant les chaînes de caractères `lib`, `.so` et tous les numéros de version. Donc, pour utiliser ces deux bibliothèques, vous devriez passer les options `-lSDL -lm` à **gcc**. Celui-ci placera alors une marque dans l'exécutable indiquant d'examiner les fichiers `/usr/lib/libSDL.so` et `/lib/libm.so.3` à chaque fois qu'une fonction SDL ou une fonction mathématique est utilisée.

5.3.2. Bibliothèques statiques

Contrairement aux bibliothèques statiques qui fournissent du code alors que l'application s'exécute, les bibliothèques statiques contiennent du code qui est lié (inséré) dans le programme lors de sa compilation. Aucun code n'est inséré au moment de l'exécution : le code est complètement autonome. Les bibliothèques statiques se terminent habituellement par `.a` suivi d'un numéro de version, comme :

```
/usr/lib/libSDL.a
/usr/lib/libm.a
```

Les fichiers `.a` forment réellement des archives de fichiers `.o` (objet), à la manière d'un fichier tar. Vous pouvez utiliser **nm** pour lister les fonctions contenues dans une bibliothèque statique :

```
% nm /usr/lib/libm.a
...
e_atan2.o:
00000000 T __ieee754_atan2

e_atanh.o:
00000000 T __ieee754_atanh
00000000 r half
00000010 r limit
00000018 r ln2_2
...
```

Quand vous utilisez **gcc**, vous référencez ces bibliothèques en enlevant les chaînes de caractères « lib », « .a » et tous les numéros de version. Donc, pour utiliser ces deux bibliothèques, vous devriez passer les options `-lSDL -lm` à **gcc**. Celui-ci importera alors du code de `/usr/lib/SDL.a` et `/usr/lib/libm.a` à chaque fois qu'il rencontre une fonction mathématique lors du processus de compilation.

5.3.3. Localisation des fichiers de bibliothèques

Si vous compilez vos propres jeux, le problème principal avec les bibliothèques sera soit que **gcc** ne trouve pas une bibliothèque statique, soit que la bibliothèque n'est pas présente sur votre système. Quand vous jouez à des jeux à partir du binaire, le problème sera soit que **ld.so** ne trouve pas la bibliothèque, soit que la bibliothèque n'est pas présente sur votre système. Il est donc opportun de d'abord parler de la façon dont **gcc** et **ld.so** s'y prennent pour trouver les bibliothèques.

gcc recherche les bibliothèques dans les « répertoires système standard » ainsi que dans ceux spécifiés avec l'option `-L`. Vous pouvez déterminer la liste des répertoires système standard avec **`gcc -`**

`print-search-dirs`.

ld.so examine un condensé binaire contenu dans un fichier nommé `/etc/ld.so.cache` pour y trouver une liste de répertoires contenant les bibliothèques dynamiques disponibles. Puisqu'il contient des données binaires, vous ne pouvez pas modifier directement ce fichier. Néanmoins, le fichier est généré à partir d'un fichier texte `/etc/ld.so.conf` que vous pouvez éditer. Ce fichier contient la liste des répertoires où **ld.so** doit rechercher les bibliothèques dynamiques. Si vous voulez ajouter des bibliothèques dynamiques dans `/home/joecool/privatelibs`, il faut ajouter ce répertoire dans `/etc/ld.so.conf`. Votre modification n'est prise en compte dans `/etc/ld.so.cache` qu'après avoir exécuté **ldconfig** ; une fois fait, **ld.so** commencera à rechercher des bibliothèques dans votre répertoire privé.

De plus, même si vous ne faites qu'ajouter de nouvelles bibliothèques à votre système, vous devez mettre à jour `ld.so.cache` pour qu'il reflète la présence des nouvelles bibliothèques.

6. Quand de mauvaises choses arrivent à de bonnes gens

Bien sûr, il n'est pas possible de recouvrir tous les types de problèmes, mais je vais souligner certains points de bon sens.

Il y a deux types de problèmes : les aléatoires et les reproductibles. Il est difficile de diagnostiquer ou de corriger des problèmes aléatoires sur lesquels vous n'avez par définition aucun contrôle. Néanmoins, si le problème est reproductible (ex : « cela se produit quand j'appuie deux fois sur la flèche gauche »), alors vous pouvez agir.

6.1. RTFM !

Relis ton fameux manuel. Le « manuel » peut revêtir plusieurs formes. Pour les jeux *open source*, on peut trouver les fichiers `readme` (« lisez-moi »). Les jeux commerciaux sont accompagnés d'un manuel imprimé et éventuellement de quelques fichiers `readme` sur le CD. N'oubliez pas d'explorer le CD de votre jeu pour obtenir des astuces et des conseils utiles.

N'oubliez pas le site web du jeu. Son auteur a probablement déjà eu affaire à maintes reprises à des personnes ayant exactement le même problème que vous, et il pourrait avoir placé des informations spécifiques à ce jeu sur son site web. Un bon exemple : les FAQ en ligne de Loki Software situées sur http://faqs.lokigames.com.

6.2. Recherchez des mises à jour et des correctifs

Si vous jouez à un jeu *open source* que vous avez compilé, assurez-vous de disposer de la version la plus récente en visitant le site web du jeu. Si votre jeu fait partie d'une distribution, assurez-vous qu'il n'y ait pas de paquet rpm/deb plus à jour pour le jeu.

Les sociétés produisant des jeux commerciaux comme Loki publient des correctifs pour leurs jeux (souvent nombreux, p.ex. pour *Myth2*™), et certains sont même injouables en leur absence (*Heretic2*™). Recherchez des correctifs sur le site web du jeu que vous ayez un problème avec le jeu ou pas ; il peut y avoir une mise à jour pour un problème de sécurité dont vous pourriez ne pas avoir eu connaissance.

À propos, Loki propose maintenant un utilitaire qui recherche les logiciels Loki sur votre disque dur et les met à jour automatiquement. Consultez http://updates.lokigames.com.

6.3. Groupes de discussion

Si vous ne savez pas ce que sont les News (Usenet), alors ça vaut la peine de prendre 30 minutes pour vous y frotter. Installez un lecteur de News. Je préfère les outils en console, et j'utilise donc tin, mais slrn est également populaire. Netscape propose également un chouette lecteur de News graphique piloté à la souris.

Par exemple, je peux me promener sur le serveur de News de Loki Software avec **`tin -g news.lokigames.com`**. Vous pouvez également spécifier quel serveur de News utiliser grâce à la variable d'environnement `$NNTP` ou le fichier `/etc/nntpserver`.

6.4. Recherche sur Google Groupes

Chaque soumission faite sur Usenet est archivée dans la base de données de Google sur http://groups.google.fr. Cette archive était située sur http://www.deja.com, mais a été rachetée par Google. Beaucoup de personnes parlent toujours de « deja ».

Il est presque sûr que quel que soit le problème que vous avez avec Linux, qu'il ait ou pas un rapport avec le jeu, il a déjà été reporté et solutionné sur Usenet, pas une, pas deux, mais de nombreuses fois. Si vous ne comprenez pas la première réponse que vous voyez (ou si elle ne fonctionne pas), essayez l'une des suivantes. Si la page n'est pas dans une langue que vous comprenez, il existe des tas de sites de traduction qui convertiront le texte dans la langue que vous préférez, comme http://www.freetranslation.com et http://translation.lycos.com. Mon navigateur web préféré, *Opera*™ (disponible sur http://www.opera.com) vous permet d'utiliser le bouton droit de la souris pour sélectionner un extrait de texte, et de cliquer avec le bouton gauche sur la sélection pour le traduire. Très utile quand une recherche sur Google Groupes renvoie une page en allemand qui semble utile et que ma petite amie (qui lit bien l'allemand) n'est pas disponible.

La recherche sur Google Groupes propose une page de recherche élémentaire et avancée. Ne perdez pas de temps avec la recherche simple. La recherche avancée est située sur http://groups.google.com/advanced_group_search.

C'est facile à utiliser. Par exemple, si mon problème est que *Quake III*™ plante à chaque fois que Lucy saute, j'entre « linux quake3 crash lucy saute » dans la boîte de texte « Retrouver les messages avec tous les mots suivants ».

Certains champs permettent de limiter la portée de votre recherche à un groupe de discussion. Prenez le temps de lire et de comprendre la signification de chaque champ. Je vous le promets : ce service ne vous décevra pas. Utilisez-le, et vous serez quelqu'un de beaucoup plus heureux. Notez bien que les groupes de discussion privés ne sont pas archivés, comme le serveur de News de Loki Software. Néanmoins, vu que beaucoup de personnes utilisent Usenet, cela n'a généralement que peu d'importance.

6.5. Débogage : traces d'appel et fichiers core

Ce n'est généralement pas quelque chose que vous ferez pour les jeux commerciaux. Pour les jeux *open source*, vous pouvez aider l'auteur en lui fournissant un fichier core ou une trace de la pile. En bref, un fichier core (dit « core dump ») est un fichier qui conserve l'état du programme au moment où il s'est crashé. Il contient des indices précieux pour le programmeur relatifs à la nature du crash : ce qui l'a causé et ce que le programme faisait quand il s'est produit. Si vous voulez en savoir plus sur les fichiers core, j'ai un super tutoriel gdb disponible sur http://www.dirac.org/linux.

En *dernier* recours, l'auteur sera intéressé par la pile d'appels au moment du plantage du jeu. Voici comment procéder :

Il arrive que les distributions configurent leur système d'exploitation en sorte que les fichiers core (qui sont principalement utiles aux programmeurs) ne sont pas générés. La première étape est d'autoriser votre système à générer des fichiers core de taille illimitée :

```
ulimit -c unlimited
```

Vous devrez maintenant recompiler le programme et passer l'option `-g` à gcc (l'explication dépasse la portée de ce document). À présent, exécutez le jeu et répétez ce qui a fait planter le programme pour générer à nouveau un fichier core. Exécutez le débogueur avec le fichier core comme second argument :

```
$ gdb ExécutableJeuChouette core
```

À l'invite, tapez **`backtrace`**. Vous verrez quelque chose comme :

```
#0 printf (format=0x80484a4 "z is %d.\n") at printf.c:30
#1 0x8048431 in display (z=5) at try1.c:11
#2 0x8048406 in main () at try1.c:6
```

Cela peut être assez long, mais utilisez votre souris pour copier et coller ces informations dans un fichier. Envoyez-le par courriel à l'auteur et indiquez-lui :

1. le nom du jeu
2. le message d'erreur qui est apparu à l'écran quand le jeu a planté.
3. ce qui a provoqué le plantage et s'il est reproductible ou non.
4. la pile d'appels

Si vous avez une bonne bande passante, demandez à l'auteur s'il souhaite le fichier core généré par son programme. S'il est d'accord, envoyez-le lui. N'oubliez pas de lui demander au préalable, car les fichiers core peuvent être très, très gros.

6.6. Parties sauvegardées

Si votre jeu permet de sauvegarder des parties, alors l'envoi à l'auteur d'une copie de la partie sauvegardée est utile car cela lui permet de reproduire le dysfonctionnement. Pour les jeux commerciaux, cette possibilité est plus fructueuse que d'envoyer un fichier core ou une pile d'appels car les jeux commerciaux ne peuvent être recompilés de sorte à inclure des informations de débogage. Vous devriez impérativement demander avant d'envoyer une partie sauvegardée car ils ont tendance à être gros, mais une société comme Loki Software dispose de beaucoup de bande passante. Mike Phillips (un ancien de Loki Software) indique que l'envoi de sauvegardes de jeux à Loki est définitivement une bonne chose.

Évidemment, cela ne s'applique que si votre jeu plante de façon reproductible dans certaines circonstances. Si le jeu vous donne une erreur de segmentation (*segmentation fault*) à chaque fois que vous l'exécutez, ou est incroyablement lent, une sauvegarde de jeu n'aura que peu d'utilité.

6.7. Que faire quand on ne trouve pas un fichier ou une bibliothèque (ou se faciliter la vie avec strace)

Parfois, vous verrez des messages d'erreur indiquant qu'un fichier n'a pu être trouvé. Le fichier pourrait être une bibliothèque :

```
% ./exult
./exult: error while loading shared libraries: libSDL-1.2.so.0: cannot load sh
file: No such file or directory
```

ou un fichier de données, comme un fichier `wad` ou `map` :

```
% qf-client-sdl
```

```
    IP address 192.168.0.2:27001 UDP Initialized Error: W_LoadWadFile: couldn't lo
```

Supposez que `gfx.wad` est déjà sur mon système, mais qu'il ne peut être trouvé étant donné qu'il n'est pas dans le bon répertoire. Mais alors, où est le bon répertoire ? Ne serait-il pas utile de savoir où ces programmes recherchent les fichiers manquants ?

C'est ici que **strace** brille. Il vous indique quels appels système sont effectués, avec quels arguments, et quelles sont les valeurs de retour. Dans mon « Guide de programmation de modules noyau » (à paraître bientôt sur le LDP), je souligne tout ce que vous devez savoir sur **strace**. Mais voici les grandes lignes : saisissez la commande

```
    strace -o ./LS_LOG /bin/ls
```

L'option `-o` envoie la sortie de **strace** dans un fichier, ici LS_LOG. Le dernier argument de **strace** est le programme à surveiller, ici **ls**. Regardez le contenu de `LS_LOG`. Assez impressionnant, n'est-ce pas ? Voici une ligne typique :

```
    open(".", O_RDONLY|O_NONBLOCK|0x18000)  = 4
```

Nous avons utilisé l'appel système `open()` pour ouvrir « `.` » avec divers arguments, et la valeur de retour de l'appel est 4. Quel est le rapport avec les fichiers non trouvés ?

Supposez que je veuille regarder la démo de *StateOfMind*™ car je ne m'en lasse pas. Un jour, j'essaie de l'exécuter et quelque chose se passe mal :

```
% ./mind.i86_linux.glibc2.1
Loading & massaging...
Error:Can't open data file 'mind.dat'.
```

Utilisons **strace** pour détecter l'endroit où le programme recherchait le fichier de données.

```
    strace ./mind.i86_linux.glibc2.1 2> ./StateOfMind_LOG
```

Lançant vim et recherchant toutes les occurrences de `mind.dat`, je trouve les lignes suivantes :

```
    open("/usr/share/mind.dat",O_RDONLY) = -1 ENOENT (No such file)
    write(2, "Error:", 6Error:)    = 6
    write(2, "Can\'t open data file \'mind.dat\'."..., ) = 33
```

Je ne recherchais `mind.dat` que dans un seul répertoire. Il apparaît clairement que `mind.dat` n'est pas dans `/usr/share`. Nous pouvons maintenant essayer de localiser `mind.dat` et de le déplacer dans `/usr/share` ou, mieux, créer un lien symbolique.

Cette méthode fonctionne également pour les bibliothèques. Supposez que la bibliothèque `libmp3.so.2` est située dans `/usr/local/include` mais que votre nouveau jeu *Kill-Metallica*™ ne le trouve pas. Vous pouvez utiliser **strace** pour déterminer où *Kill-Metallica*™ doit rechercher la bibliothèque et créer un lien symbolique de `/usr/local/include/libmp3.so.2` vers l'endroit où *Kill-Metallica*™ recherchait le fichier de bibliothèque.

strace est un utilitaire très puissant. Quand vous essayez de savoir pourquoi quelque chose n'est pas trouvé, il est votre meilleur allié, et est même plus rapide que la consultation du code source. De plus, vous ne pouvez pas rechercher d'informations dans le code source des jeux commerciaux de Lokisoft ou Tribsoft. Mais vous pouvez toujours utiliser **strace** !

6.8. Consoles corrompues

Parfois, un jeu se termine anormalement et votre console se retrouve alors dans un état bizarroïde : le texte à l'écran est du charabia, votre bel écran noir habituel ressemble à un écran semi-graphique, et cætera. Quand vous tapez sur **Entrée**, un retour à la ligne n'est pas reproduit à l'écran. Parfois, certaines touches du clavier ne répondent pas. La déconnexion suivie d'une reconnexion ne marche pas toujours, mais il y a d'autres possibilités :

- Si aucun des caractères que vous tapez n'apparaît à l'écran, les réglages de votre terminal peuvent être incorrects. Essayez « stty echo ». Cela devrait rétablir l'écho des caractères.
- À l'invite, tapez **`reset`**. Cela devrait éliminer beaucoup de problèmes, y compris les consoles corrompues par un jeu basé sur SVGAlib ou ncurses.
- Essayez de ré-exécuter le même jeu normalement. Une fois, j'ai dû tuer *Quake III*™ en toute hâte, et j'ai donc effectué un **ctrl+alt+backspace**. La console était corrompue et présentait un écran quasi-graphique. Exécuter *Quake III*™ et le quitter normalement a corrigé le problème.
- Les commandes **deallocvt** et **openvt** fonctionneront pour la plupart des autres problèmes. **deallocvt *N*** tue entièrement le terminal `N`, de sorte que `Alt-FN` ne fonctionne plus du tout. **openvt -c *N*** ou **`openvt -c`** ***N*** le redémarre.
- Si certaines touches de votre clavier ne fonctionnent pas, faites preuve de créativité. Si vous voulez redémarrer mais que la touche **o** ne fonctionne pas, essayez d'employer **halt**. Une méthode que j'ai expérimentée est de taper une commande à l'invite et d'utiliser des caractères à l'écran en utilisant le copier-coller avec la souris. Par exemple, vous pouvez taper **`ps ax`**, et vous êtes sûr(e) d'avoir un **h**, **a**, **l** et un **t** quelque part à l'écran. Vous pouvez utiliser la souris pour copier et coller le mot « halt ».
- L'option la plus regrettable est le redémarrage. Si c'est possible, un arrêt ordonné est préférable ; utilisez **halt** ou **shutdown**. Sinon, utilisez **ssh** depuis une autre machine. Cela fonctionne parfois quand votre console présente d'importants dysfonctionnements. Dans le pire des cas, appuyez sur le bouton Reset ou Power (réinitialisation ou arrêt de l'alimentation).

Notez que si vous utilisez un système de fichiers journalisé comme ext3, reiserfs ou xfs, l'appui sur le bouton Power n'est pas aussi néfaste que cela. Vous êtes toujours supposé arrêter la machine d'une façon ordonnée, mais l'intégrité du système de fichiers sera préservée. Vous ne verrez normalement pas de **fsck** pour les partitions qui utilisent le système de fichiers journalisé.

6.9. Système bloqué

Quand un ordinateur se bloque, le clavier et la souris ne répondent plus du tout. C'est une conséquence directe d'un bogue dans le noyau Linux. Bien que Linux soit réputé pour sa stabilité, de telles choses peuvent arriver, en particulier avec les jeux qui occasionnent des événements matériels extrêmement synchronisés se produisant très rapidement, même pour un ordinateur. Quand un ordinateur se bloque, cela peut être un « blocage total », signifiant que le noyau a complètement cessé de fonctionner. Cela indique souvent que le matériel est en cause. Le seul remède à ce type de blocage est d'appuyer sur le bouton Reset ou Power. Le blocage peut également être « léger », à savoir que le noyau fonctionne toujours dans une certaine mesure. Il est possible de se remettre gracieusement de cette situation.

- La première chose à essayer est de taper **control+alt+backspace** qui tue X. Si vous récupérez le contrôle sur votre système, le noyau n'était pas réellement bloqué. Si cela ne fonctionne pas après quelques secondes, il faut alors redémarrer le système en suivant les recommandations suivantes :

- Utilisez **control+alt+delete** pour redémarrer le système. Vous savez que cela a fonctionné si l'ordinateur émet un bip après quelques secondes (c'est le BIOS qui dit « Tout est OK » au cours du cycle de démarrage).
- Connectez-vous à partir d'un autre système via ssh. Si vous y parvenez, redémarrez ou arrêtez le système.
- Si cela n'est pas possible, vous devrez utiliser la « touche magique **SysRq** » qui est documentée dans `/usr/src/linux/Documentation/sysrq.txt`. Voici un résumé pour l'architecture x86 (consultez la documentation pour les autres architectures). Notez que si votre clavier n'a pas de touche **SysRq**, il vous faudra utiliser la touche **PrintScreen** :

    1. Tapez **Alt+SysRq+s** pour tenter de synchroniser vos systèmes de fichiers montés afin que les changements apportés aux fichiers soient effectués sur disque. Vous pouvez entendre de l'activité du disque dur. Si vous regardez sur une console, le système devrait afficher le nom des périphériques pour lesquels cette opération a eu lieu.
    2. Quelques secondes plus tard, tapez **Alt+SysRq+u** pour essayer de remonter les systèmes de fichiers montés en mode lecture seule. Vous devriez entendre de l'activité disque. Si vous examinez une console, le système affichera les périphériques qui ont été remontés.
    3. Après un bref moment, utilisez **Alt+SysRq+b** pour redémarrer le système.
    4. Vous pouvez aussi taper **Alt+SysRq+h** pour obtenir un écran d'aide ultra-concis.

Pour prendre en charge la touche magique **SysRq**, votre noyau doit avoir été compilé à cet effet. Vous trouverez cette option sous « `Kernel Hacking | Kernel Debugging | Magic SysRq key` ». Si la séquence magique **SysRq** n'éteint pas correctement votre système, votre noyau s'est réellement bien planté et le seul remède est d'utiliser le bouton Reset ou Power.

7. Cartes vidéo

7.1. Historique

Il était une fois, une société de San Jose, en Californie, appelée 3dfx Interactive™ dominait le marché des cartes vidéo destinées au jeu. En octobre 1996, elle a mis sur le marché la Voodoo I, qui a connu un succès phénoménal. C'était la première carte proposant une accélération matérielle, mais elle n'effectuait que du rendu 3D ; il fallait une seconde carte vidéo 2D de haute qualité pour effectuer le rendu 2D (Matrox était immensément populaire à l'époque) alors que les informations 3D (voyez Glide2, à la Section 3.1, « Glide2 ») sont transmises à la Voodoo I et rendues, en utilisant le matériel rapide de la Voodoo pour effectuer les calculs graphiques nécessaires. La Voodoo Rush est sortie en avril 1996. Elle aurait dû être plus puissante, avec un GPU cadencé à 50 Mhz et 8 Mo de RAM. Mieux encore, elle constituait leur première carte combinée 2D/3D, libérant un port PCI précieux (la plupart des PC n'avaient que deux ports PCI à l'époque) mais la Rush n'a pas été aussi populaire. 3dfx a supprimé l'unité multi-textures de la Rush, qui a dès lors été surclassée par la Voodoo I. Pendant ce temps-là, ATI produisait sa série de Rage et nVidia celle de Riva 128, mais la Voodoo I dominait toujours largement.

C'était une belle époque pour Linux. id Software avait libéré le code source de Doom et porté Quake I sous Linux (décembre 1996). Nous goûtions pour la première fois au jeu commercial réel. Le choix était vite fait : vous achetiez une Voodoo. Et vous vous sentiez bien, car 3dfx avait ouvert ses pilotes. La reine des cartes vidéo fonctionnait grâce à des développeurs Linux. Non seulement nous disposions des meilleures cartes vidéo, mais de plus leurs pilotes étaient tous *open source*.

En mars 1998, 3dfx lançait la Voodoo II, avec sa bande passante mémoire de 3.6 Go/sec, 12 Mo de mémoire vidéo et un cœur fonctionnant à 90 MHz. Elle permettait des résolutions allant jusqu'à 1024x768. C'était 3dfx à son apogée. Comme la Voodoo I, la Voodoo II était une carte ne s'occupant que de la 3D, et se reposant sur une autre carte vidéo pour la 2D. La Voodoo Banshee est sortie en septembre 1998 comme une carte combinée 2D/3D, comme la Rush. Malgré un cœur plus rapide fonctionnant à 100 MHz, la Banshee était dépassée par la Voodoo II du fait de la suppression de l'unité multi-textures, comme pour la Rush. Et à nouveau, comme la Rush, elle n'était pas populaire. Mais 3dfx régnait en maître, et personne ne pouvait leur faire de l'ombre.

La Voodoo III est sortie en avril 1999. Il y en a eu plusieurs, au cœur variant de 143 à 183 MHz. Certaines versions disposaient d'une sortie TV. Il y avait des versions PCI et AGP (c'était la première carte vidéo AGP). C'était un autre succès, mais 3dfx commençait à perdre du terrain au profit de nVidia, qui produisait la TNT 2. Celle-ci surclassait la Voodoo II, et offrait une accélération 3D avec des couleurs 32 bits, alors que les Voodoo étaient limitées aux couleurs 16 bits. Mais la vie était toujours belle pour Linux. Nous disposions d'une carte qui était pratiquement au coude à coude avec nVidia, nos pilotes étaient *open source* et, en décembre 1999, id Software nous a fait un grand cadeau : ils ont ouvert le code source de Quake I.

Ensuite, la GeForce 256 de nVidia est apparue en octobre 1999. La Voodoo IV de 3dfx, son concurrent direct, avait à peu près une année de retard, ce qui est pour le moins ennuyeux quand on se bat sur un marché « de pointe ». Alors que les travaux en recherche et développement de nVidia étaient appliqués à ses cartes, 3dfx ne faisait qu'ajouter de la RAM plus rapide. Les Voodoo IV et V rendaient les couleurs 32 bits, prenaient très bien en charge l'anti-crénelage, proposaient un second GPU, plus de mémoire, et étaient pour ainsi dire supérieures aux autres cartes vidéo. Néanmoins, la sortie tardive des Voodoo IV et V couplée au fait qu'on pouvait obtenir la GeForce pour moitié moins explique le naufrage rapide de 3dfx. Pour Linux, les plus récentes Voodoo ne pouvaient accélérer que pour les couleurs 16 et 24 bits. Pire encore, le second GPU de la Voodoo V n'était pas exploité par le pilote Linux (et, à ce jour, la Voodoo V est fonctionnellement équivalente à la Voodoo IV sous Linux). La plupart des utilisateurs Windows sont passés à nVidia et, bien que les pilotes de cette dernière étaient propriétaires, même les utilisateurs Linux commençaient à la choisir. VA Linux, le plus grand vendeur des serveurs Linux, plaçait des nVidia dans ses machines.

Ensuite, en avril 2000, 3dfx a été attaqué sur un autre front : ATI commençait à produire sa première génération de Radeon. Auparavant, ATI avait toujours été un fabricant de puces graphiques innovant (leurs propres puces accélératrices 3D datent de 1996, à peu près au même moment que 3dfx), mais fort discret. Les Radeon étaient leur première carte accélératrice 3D à réellement intéresser les joueurs. Leurs Radeon écrasaient à la fois nVidia et 3dfx. Ils ont collaboré avec des développeurs Linux, ont ouvert le code source de tous leurs pilotes et ont été acclamés comme le grand espoir pour le jeu sous Linux. nVidia revint à la charge, et c'en était trop pour 3dfx. Entre la défaite dans les bancs d'essais contre la GeForce et la Radeon, leurs nouvelles cartes en retard et leurs prix élevés, 3dfx avait perdu sa part de marché et n'avait plus les fonds nécessaires pour continuer ses activités. Le 18 avril 2001, ils ont vendu la plupart des leurs avoirs et technologies à nVidia et, en octobre 2002, ont finalement fait aveu de faillite.

La disparition de 3dfx était très soudaine et une gifle pour la communauté *open source*. Je me souviens toujours de mon ami Gabe Rosa m'envoyant un courriel avec ces simples mots « `Look at /.` » (Va voir sur slashdot) et la vision de la nouvelle. C'était le 2e jour le plus sombre de l'histoire du jeu sous Linux (après la mort de Loki). Et c'était aussi vraiment dommage. 3dfx était sur le point de sortir une nouvelle Voodoo V avec 4 GPU qui aurait écrasé les offres de ATI et nVidia, ainsi qu'une nouvelle carte au nom de code « Rampage » qui les auraient ramené sur le devant de la scène. On raconte que la technologie de Rampage (qui a été vendue à nVidia) s'est retrouvée dans la GeForce 5900. Pas trop mal pour une technologie vieille de 3 ans !

Au début, tout était simple. Les joueurs Linux gardaient soit leurs Voodoo *open source*, acquéraient une Radeon *open source* ou une GeForce propriétaire. Néanmoins, les jeux grossissant et s'améliorant, ce n'était qu'une question de temps avant que les Voodoo ne soient plus viables pour les jeux modernes. Certains utilisaient toujours des Voodoo, mais ces personnes étaient pratiquement hors du coup en ce qui concerne le jeu.

ATI a produit un nombre incroyable de versions de chaque carte vidéo, et il devenait difficile de suivre l'évolution de leur terminologie. ATI et nVidia dominaient le marché. Leurs produits ont toujours été au coude à coude depuis lors, la GeForce prenant l'avantage un peu plus souvent que la Radeon. Mais les pilotes de la Radeon étaient *open source*, et de nombreux utilisateurs Linux lui restaient fidèles. Ensuite, cela s'est compliqué.

ATI a commencé à devenir de plus en plus réticent aux pilotes *open source* pour leurs nouvelles cartes et, soudainement, il n'était plus facile de savoir qui était le « bon ». L'excuse de nVidia était qu'une partie de leur code GL est sous licence d'une autre société, et ne peut par conséquent pas être « libérée ». Vraisemblablement, ATI ne veut pas collaborer afin de conserver ses secrets de fabrique. Et cela ne s'arrange pas. Les pilotes ATI Linux ont souffert de performances extrêmement faibles. Même quand une offre de ATI est meilleure que celle de la GeForce du moment pour Windows, la carte est toujours écrasée par la GeForce sous Linux. Du fait de pilotes ATI Linux calamiteux, les utilisateurs Linux ne peuvent se fier aux bancs d'essais ou aux tests de cartes prévus pour MS Windows. Ils ne sont tout simplement pas appropriés. Et c'est à peu près au point où nous en sommes pour le moment.

Finalement, le seul véritable banc d'essais des cartes vidéo sous Linux date malheureusement, à ma connaissance, de mars 2001, entre une Radeon 32 DDR et une GeForce 2. Vous pouvez le consulter vous-même sur http://www.linuxhardware.org/features/01/03/19/0357219.shtml, mais la conclusion est que la GeForce 2 domine de la tête et des épaules la Radeon 32 DDR.

7.2. Situation actuelle (13 juillet 2003)

La dernière offre de nVidia est la GeForce 5900, basée sur le jeu de composants NV35. Elle est bien prise en charge sous Linux par des pilotes de haute qualité mais propriétaires. Ils ne fournissent pas d'informations aux développeurs Linux, et vous ne pourrez donc utiliser que leurs pilotes binaires ; ils ne font pas partie de XFree86. nVidia utilise une architecture unifiée pratique : leurs pilotes prennent en charge de la TNT 2 à la GeForce 5900.

ATI a travaillé avec les développeurs Linux pour toutes les Radeon jusqu'à la Radeon 9200. Ces cartes font l'objet d'une prise en charge 2D et 3D dans XFree86. Je ne suis pas entièrement sûr de la qualité de ces pilotes *open source* ; néanmoins, *Soldier of Fortune I*™ et *Heavy Metal*™ ont toujours des problèmes de textures opaques avec la première génération de Radeon. Après la 9200, vous devez utiliser les pilotes binaires propriétaires, disponibles au format rpm, depuis le site web de ATI. Ces pilotes sont abominables : un de mes amis m'affirme que sa GeForce 4400 surclasse sa Radeon 9700 pro. C'est une honte !

Sur le papier, et selon les bancs d'essais pour Windows, la Radeon 9800 écrase la mal-conçue GeForce 5800 et dépasse légèrement la GeForce 5900. Sur le papier, elle est tout simplement la carte la plus impressionnante. Mais, à nouveau, le problème des pilotes ne nous permet pas d'en bénéficier. Si vous désirez acheter la meilleure carte pour Linux, vous devrez utiliser la GeForce 9800. Préparez-vous simplement à manger des nouilles pendant quelques mois : les deux cartes sont excessivement chères.

7.2.1. Support de SVGAlib

Au 30 juin 2002, la prise en charge par la SVGAlib des cartes Radeon est problématique. Les développeurs ont rapporté que SVGAlib fonctionne avec les Radeon 7500 et Radeon QD (modèle 64 Mo DDR) mais a quelques soucis avec la Radeon VE.

Je ne dispose pas d'informations concernant les cartes GeForce.

7.3. Quelle carte vidéo dois-je acheter ? (13 juillet 2003)

La réponse était très difficile l'an dernier, mais voici mon opinion :

1. Toutes les cartes GeForce requièrent un pilote propriétaire qui « salit » votre noyau. Néanmoins, c'est également la cas de toutes les cartes ATI suivant la Radeon 9200.

2. nVidia a prouvé qu'elle se souciait suffisamment de Linux pour écrire et actualiser des pilotes vidéo de haute qualité pour Linux. Même quand ATI a ouvert le code source de ses pilotes, ils se reposaient sur les développeurs Linux pour faire le sale boulot. Leurs pilotes propriétaires actuels sont ignobles.

3. La Radeon 9800 actuelle bat tout juste la GeForce 5900 dans les bancs d'essais et sur le plan

des spécifications, mais les utilisateurs Linux ne pourront en bénéficier du fait de la faiblesse des pilotes de la 9800.

4. ATI a depuis longtemps l'habitude d'abandonner le support de son matériel dès que c'est possible.

5. Sous MS Windows, quand la GeForce bat son principal adversaire Radeon, les critiques affirment généralement que les graphismes de la Radeon étaient plus soignés. Je ne sais pas si cela se ressent également sous Linux.

En fin de compte, la plupart devrait acheter une GeForce pour le moment.

7.4. Définitions : carte vidéo et terminologie 3D

Parlons à présent de la terminologie des cartes vidéo et des graphiques 3D. Ce n'est pas primordial pour faire fonctionner un jeu en pratique, mais cela peut vous aider à décider quelles options matérielles et logicielles vous conviennent le mieux.

7.4.1. Textures

Une scène rendue est constituée à la base de polygones et de lignes. Une texture est une image 2D (habituellement une bitmap) recouvrant les polygones d'un monde 3D. Pensez à une couche de peinture sur les polygones.

7.4.2. *T&L : Transform and Lighting*

Le T&L (transformation et éclairage) est le processus de traduction de toutes les informations du monde 3D (position, distance et sources de lumière) en une image 2D effectivement affichée à l'écran.

7.4.3. *AA : Anti Aliasing*

L'anti-aliasing (anti-crénelage) est le lissage de l'effet d'escalier d'une courbe ou d'un polygone, apparaissant lors du dessin d'une ligne brisée ou d'une courbe composée de pixels (de forme rectangulaire), aussi appelé « crénelage ». Il se produit quand les pixels forment une ligne crénelée plutôt qu'une courbe ou une ligne lisse. L'AA utilise un filtrage gourmand en temps CPU pour lisser de tels contours crénelés. Cela améliore l'aspect visuel d'un jeu, mais peut également grever dramatiquement les performances.

L'AA est utilisé dans différentes situations. Par exemple, quand vous grossissez une image, vous pouvez remarquer que des lignes qui étaient lissent deviennent crénelées (essayez avec The Gimp). Le rendu des polices de caractères est une autre grande application pour l'AA.

L'AA peut être fait soit par l'application elle-même (comme avec The Gimp ou le système de polices de XFree86), soit par le matériel, si votre carte le supporte. Puisque l'AA est gourmand en temps CPU, il vaut mieux l'effectuer en matériel, mais si nous parlons d'applications semi-statiques, comme The Gimp, cela ne pose pas vraiment de problème. Pour les situations dynamiques, comme les jeux, effectuer l'AA en matériel peut être crucial.

7.4.4. *FSAA : Full Screen Anti-Aliasing*

Le FSAA (anti-crénelage plein écran) implique habituellement le dessin d'une version grossie de l'écran entier dans un framebuffer séparé, en effectuant l'AA sur l'image entière puis en la ramenant à la résolution normale. Comme vous pouvez l'imaginer, c'est extrêmement gourmand en temps CPU. Vous ne verrez jamais de FSAA non accéléré matériellement.

7.4.5. *Mip Mapping*

Le « mip mapping » est une technique consistant à stocker diverses copies à l'échelle de la même texture dans la mémoire de la carte vidéo, afin de représenter la texture à différentes distances. Quand la texture est très éloignée, une plus petite version de la texture est utilisée. Quand la texture est proche, une plus grande est utilisée. Le *mip mapping* peut être utilisé quelle que soit la méthode de filtrage. Il réduit non seulement les besoins en bande passante mémoire (puisque les images sont stockées sur le matériel), mais offre également une meilleure qualité d'image.

7.4.6. Filtrage de textures

Le filtrage de textures est la fonctionnalité fondamentale requise pour fournir des graphiques 3D agréables. Il a plusieurs applications, comme mélanger sans encombre des textures adjacentes, et rendre réaliste des textures vues depuis un angle (p.ex. regarder un panneau d'affichage depuis un angle extrême). Il existe plusieurs techniques de filtrage de textures incluant l'échantillonnage de points et les filtrages bilinéaire, trilinéaire et anisotrope.

Quand je parle de « dégradation de performances », gardez à l'esprit qu'elle dépend de la résolution utilisée. Par exemple, à une basse résolution, l'impact sur les performances de l'utilisation du filtrage trilinéaire au lieu du filtrage bilinéaire est négligeable. Mais à de hautes résolutions, il peut être énorme. De plus, je ne connais aucune carte qui utilise le filtrage de textures anisotrope. Les pilotes TNT le prétendent, mais j'ai lu que ces pilotes utilisent toujours le filtrage trilinéaire au moment du rendu réel d'une image à l'écran.

7.4.6.1. Filtrage de textures avec échantillonnage de points

L'échantillonnage de points est rare de nos jours, mais si vous exécutez un jeu avec le « rendu logiciel » (ce que vous devrez faire si vous exécutez un jeu avec accélération 3D sans carte accélératrice 3D), vous constaterez probablement son utilisation.

7.4.6.2. Filtrage de textures bilinéaire

Le filtrage bilinéaire est un filtrage de textures peu exigeant en temps de calcul mais de basse qualité. Il approxime les différences entre les textures en échantillonnant la couleur des quatre texels les plus proches (supérieur, inférieur, gauche et droit). Toutes les cartes vidéo accélératrices 3D modernes peuvent effectuer du filtrage bilinéaire en matériel sans chute des performances.

7.4.6.3. Filtrage de textures trilinéaire

Le filtrage trilinéaire est un filtre bilinéaire de haute qualité qui utilise les quatre pixels les plus proches du deuxième niveau de mip map (*mip map level*) le plus approprié pour produire des transitions plus douces entre les niveaux. Le filtrage trilinéaire échantillonne à partir de huit pixels et les intègre avant de calculer le rendu. Le filtrage trilinéaire utilise toujours le *mip mapping*. Il élimine l'effet de bandes qui apparaît entre des niveaux adjacents. La plupart des cartes vidéo accélératrices 3D peuvent effectuer du filtrage trilinéaire en matériel sans impact sur les performances.

7.4.6.4. Filtrage de textures anisotrope

Le filtrage anisotrope est la meilleure mais la plus demandeuse en temps de calcul des trois méthodes habituelles de filtrage de textures. Le filtrage trilinéaire est capable de produire de beaux résultats, mais il ne fait qu'échantillonner à partir d'une zone carrée, ce qui n'est pas toujours la méthode idéale. Anisotrope (signifiant « depuis n'importe quelle direction ») échantillonne à partir de plus de 8 pixels. Le nombre de pixels utilisés et lesquels sont utilisés dépendent de l'angle de vision de la surface par rapport à votre écran. Il fait des merveilles lors de la visualisation de caractères alphanumériques depuis un certain angle.

7.4.7. *Z-buffer*

Un Z-buffer est une partie de RAM qui représente la distance entre l'observateur (vous) et chaque pixel d'un objet. Beaucoup de cartes accélératrices 3D modernes disposent d'un z-buffer dans leur RAM vidéo, ce qui accélère considérablement les choses, mais il peut également être pris en charge par le moteur de rendu de l'application. Néanmoins, ce type de choses devrait clairement être fait en mémoire à chaque fois que c'est possible.

Chaque objet possède son ordre d'empilement, à la manière d'une pile de cartes. Quand des objets sont rendus dans un framebuffer 2D, le moteur de rendu supprime les surfaces cachées en utilisant le Z-buffer. Il y a deux façons de s'y prendre. Les moteurs stupides dessinent d'abord les objets éloignés puis seulement les objets proches, cachant les objets situés en dessous d'eux dans le Z-buffer. Les moteurs intelligents calculent quelles portions des objets seront cachées par les objets situés au-dessus et ne rendent tout simplement pas les portions que vous ne verriez de toute façon pas. Pour les textures compliquées, cela offre des grandes économies en temps processeur.

8. Son

8.1. Quelle carte son est la meilleure ?

Par « meilleure », j'entends « meilleure pour le jeu ». Les joueurs demandent une haute qualité sonore sans trop de paramétrage à effectuer. Les musiciens ont par contre d'autres exigences : le concept de « meilleure carte son » sera probablement différent pour eux. Si vous êtes un musicien, vous devriez consulter le « Guide pratique de la qualité du son sous Linux » (*Linux Audio Quality HOWTO [http://www.linuxdj.com/audio/quality/]*).

Maintenant que Linux commence à mûrir, cette question a perdu de son importance. Auparavant, les cartes son sans puce MIDI intégrée (la plupart des cartes son PCI) ne pouvaient pas interpréter le MIDI. C'était principalement un problème pour les jeux comme xdoom™ ou lxdoom™ qui utilisent musserv. De nos jours, il existe des émulateurs MIDI comme Timidity et des bibliothèques comme SDL qui ne requièrent pas de support MIDI matériel. Franchement, j'ai eu beaucoup de cartes et je n'ai pas constaté de différences en ce qui concerne le jeu. Si vous voulez des choses comme la conversion d'un enregistrement LP en format numérique, alors votre choix de carte son se portera sur un convertisseur analogique/numérique de niveau professionnel. Dans ce guide, nous supposerons que vous êtes plus un joueur qu'un ingénieur du son.

Votre décision devrait se baser sur la facilité de configuration. Si vous avez déjà une carte qui fonctionne bien, c'est suffisant. Si vous explorez le marché pour acheter une carte son, prenez quelque chose qui ne prend qu'une seconde à configurer. Les cartes PCI sont beaucoup plus faciles à gérer que les cartes ISA car vous ne devez pas indiquer à leur pilote quelles ressources système (IRQ, DMA, adresses d'entrée-sortie) utiliser. Certaines cartes ISA sont plug-n-play, comme la Creative AWE-64, et le noyau Linux a beaucoup évolué pour les auto-configurer.

Ma recommandation personnelle est une carte à base de es1370 ou es1371, qui utilise les pilotes son es1370 et es1371 sous Linux. Ces cartes vont de la plus ancienne Ensoniq es1370 à la plus récente Creative PCI-128. Ces cartes sont très bon marché et triviales à faire fonctionner sous Linux.

J'étais un grand fan des cartes son Creative Soundblaster AWE 32, AWE 64 et AWE 64 gold. Ces cartes ISA PnP sont bien prises en charge à la fois par OSS et par Alsa. Elles utilisent toutes la même puce de synthèse sonore E-mu 8000 qui leur permet de jouer 32 voix simultanément (elles ont 32 « canaux »). Elles sont ISA, mais plug-n-play. Quelques remarques : primo, le *Soundblaster AWE HOWTO* est très dépassé. Secundo, la AWE 64 et la AWE 64 gold peuvent jouer jusqu'à 64 voix simultanément, mais cela est fait en logiciel. Creative n'a jamais publié de pilote Linux pour ces cartes (ni n'ont publié d'informations de programmation à ce sujet), et les utilisateurs Linux ne peuvent donc pas utiliser les 32 canaux supplémentaires de la AWE 64 et de la AWE 64 gold. Pour eux, ces trois cartes sont complètement identiques (bien que la AWE 64 gold ait des connecteurs plaqués or, qui offrent une meilleure qualité de son que les connecteurs en acier habituels).

La Creative Soundblaster Live! est une carte son PCI extrêmement populaire de nos jours. Je n'en ai jamais possédé, et je ne peut donc pas vous donner mes impressions. Néanmoins, on a reporté à de nombreuses reprises des problèmes sérieux avec la Live! et les cartes mères AMD qui utilisent le southbridge 686b. Une recherche sur Google devrait produire un tas d'informations sur ce problème.

Un élément plus pertinent est les haut-parleurs, mais même ici la différence n'est pas énorme. J'ai eu des haut-parleurs Altec Lansing coûteux qui ne fonctionnent que légèrement mieux que les haut-parleurs bon marché. Tout dépend du prix que vous êtes disposé(e) à mettre, mais ne vous attendez pas à de grosses différences. Vous devriez vous procurer quelque chose avec un caisson de basses séparé : les différences sont perceptibles au prix de câbles d'alimentation et de connecteurs supplémentaires.

8.2. Pourquoi le son ne fonctionne-t-il pas ?

Tout d'abord, votre jeu n'est probablement pas en cause ; il s'agit probablement de votre configuration. À ma connaissance, il y a 3 possibilités pour faire fonctionner une carte son configurée sous Linux : les pilotes son libres OSS accompagnant le noyau Linux, les pilotes Alsa et les pilotes son OSS commerciaux. Personnellement, je préfère les pilotes OSS, mais beaucoup ne jurent que par Alsa. Les pilotes commerciaux sont bons quand vous avez du mal à faire fonctionner votre carte son avec des méthodes libres. Ne crachez pas dessus : ils sont très bon marché (du genre 10 ou 20 $), supportent des cartes son du dernier cri et évitent beaucoup d'essai-erreur lors du processus de configuration.

Il y a 5 choses qui peuvent mal se passer avec votre système son :

1. interruption partagée
2. pilote mal configuré
3. quelque chose accède déjà à la carte son
4. mauvais pilote utilisé
5. problème de permissions

8.2.1. Interruption partagée

La première chose à faire est de savoir s'il y a un conflit d'IRQ. Les cartes ISA ne peuvent pas partager de canaux d'interruption. Les cartes PCI le peuvent, mais certains types de cartes à grande bande passante n'aiment tout simplement pas partager, en ce compris les cartes réseau et son. Pour déterminer si vous avez un conflit, tapez **`cat /proc/interrupts`**. J'obtiens comme sortie sur mon système :

```
$ cat /proc/interrupts
           CPU0       CPU1
  0:   24185341          0          XT-PIC  timer
  1:     224714          0          XT-PIC  keyboard
  2:          0          0          XT-PIC  cascade
  5:    2478476          0          XT-PIC  soundblaster
  5:     325924          0          XT-PIC  eth0
 11:     131326          0          XT-PIC  aic7xxx
 12:    2457456          0          XT-PIC  PS/2 Mouse
 14:     556955          0          XT-PIC  ide0
NMI:          0          0
LOC:   24186046   24186026
ERR:       1353
```

La seconde colonne est présente car j'ai 2 processeurs sur cette machine ; si vous n'en avez qu'un, vous n'aurez qu'une seule colonne CPU. Les nombres sur la gauche sont les IRQ attribuées et les chaînes de caractères sur la droite indiquent quel périphérique a été assigné à cette IRQ. Vous pouvez voir que j'ai un conflit entre la carte son (soundblaster) et la carte réseau. Elles partagent l'IRQ 5. En fait, j'ai créé cet exemple de toutes pièces car je voulais vous montrer à quoi ressemble un conflit d'IRQ. Mais si j'avais ce conflit, ni mon réseau ni mon son ne fonctionneraient correctement (ou ne fonctionneraient tout court !).

Si la carte son est une PCI, le plus simple est de simplement déplacer l'une des cartes dans un port différent et d'espérer que le BIOS trie tout correctement. Une méthode plus avancée serait d'aller dans le BIOS et d'attribuer des IRQ à des ports spécifiques. Les BIOS modernes en sont capables.

8.2.2. Pilote mal configuré

Parfois, une carte utilisera toujours une certaine IRQ (uniquement pour les cartes ISA). Alternativement, certaines cartes ISA peuvent être configurées pour utiliser une IRQ spécifique en utilisant des cavaliers sur la carte elle-même. Avec ce type de cartes, vous devez passer les IRQ, adresses mémoire et port d'entrée-sortie corrects au pilote.

C'est un sujet spécifique à la carte son, qui dépasse le cadre de ce guide.

8.2.3. Quelque chose accède déjà à votre carte son

Peut-être une application accède-t-elle déjà à votre carte son ? Par exemple, peut-être y a-t-il un lecteur MP3 qui est en mode pause… Si quelque chose accède déjà à votre carte, d'autres applications ne pourront l'utiliser. Même s'il a été conçu pour partager la carte entre des applications, je trouve que esd (le serveur de son de Enlightenment) ne fonctionne pas toujours correctement. Le meilleur outil est **lsof**, qui montre quels processus accèdent à un fichier. Votre carte son est représentée par `/dev/dsp`. À l'instant, j'écoute un MP3 (pas de Metallica bien sûr…) avec mp3blaster.

```
# lsof /dev/dsp
COMMAND    PID USER   FD   TYPE DEVICE SIZE   NODE NAME
mp3blaste 1108    p    6w   CHR   14,3      662302 /dev/dsp
```

fuser est similaire, mais vous permet d'envoyer un signal à un processus quelconque accédant au fichier de périphérique.

```
# fuser -vk /dev/dsp

                     USER        PID ACCESS COMMAND
/dev/dsp             root       1225 f....  mp3blaster
                     root       1282 f....  mp3blaster
```

Après avoir saisi la commande, mp3blaster a été tué par un signal SIGKILL. Voyez les pages de manuel de **lsof** et **fuser** ; elles sont très utiles. Vous devrez les exécuter en tant que root car vous demandez des informations concernant des processus pouvant appartenir à root.

8.2.4. Mauvais pilote (ou aucun pilote) utilisé

Il n'y a que deux façons de configurer votre carte :

1. La prise en charge doit être compilée directement dans le noyau
2. Le bon pilote doit être chargé en mémoire

Vous pouvez trouver quel pilote utilise votre carte son en utilisant **lsmod** ou en examinant la sortie de **dmesg**. Étant donné que le son est de première importance pour moi, je le compile toujours dans mes noyaux. Si aucun pilote n'est chargé, vous devez déterminer ce qui a été compilé dans votre noyau. Ce n'est pas si facile. Il vaut mieux compiler votre noyau. À propos, sachez que la compilation de votre propre noyau est la première étape vers la compétence sous Linux. C'est pénible la première fois, mais une fois que c'est fait correctement, cela devient très facile, en particulier si vous conservez tous vos anciens fichiers `.config` et utilisez des choses comme **`make oldconfig`**. Voyez le « Guide pratique du noyau Linux » (*Kernel-HOWTO* [http://www.traduc.org/docs/howto/lecture/Kernel-HOWTO.html]) pour les détails.

Si vous n'avez pas compilé votre noyau vous-même, il est très probable que votre système soit configuré de sorte à charger les pilotes son sous forme de modules. C'est ainsi que fonctionnent les distributions. Compilez tout ce qui est possible sous forme de module et essayez de les charger tous. Ainsi, si vous ne voyez pas le pilote de votre carte son avec lsmod, votre carte n'est probablement pas encore configurée.

8.2.5. Problème de permissions

Si la carte son ne fonctionne que lorsque vous êtes root, vous avez probablement un problème de permissions. Si c'est le cas, sous root, examinez le groupe propriétaire de la carte son en utilisant **`ls -l /dev/dsp`** ; il s'agira probablement de `audio`. Ensuite, sous root, ajoutez votre utilisateur non-root au groupe `audio` dans `/etc/group`. Par exemple, j'ai ajouté les utilisateurs `toto` et `lulu` au groupe `audio` sur mon système :

```
audio:x:29:toto,lulu
```

N'oubliez pas d'utiliser **grpconv** si vous utilisez les mots de passe masqués (*shadow passwords*), ce qui devrait être le cas sur la plupart des distributions récentes, afin de préserver une configuration de groupes cohérente. Ensuite, déconnectez-vous et reconnectez-vous comme utilisateur normal. Votre carte son devrait fonctionner à présent. Merci à James Barton pour m'avoir rappelé d'ajouter ceci à ce guide.

9. Problèmes divers

9.1. Problèmes d'accélération matérielle

XFree86 4.x fournit une approche plus centralisée et plus autonome en ce qui concerne la vidéo. Beaucoup des joyeusetés comme les modules noyau pour un accès non root aux cartes vidéo ont, heureusement, disparu.

9.1.1. L'accélération matérielle ne fonctionne pas du tout

Si vous obtenez quelque chose comme 1 image par seconde, alors votre système n'utilise pas d'accélération matérielle 3D. Voici deux causes possibles :

1. votre système 3D est mal configuré (très probable) ;
2. le jeu X est mal configuré (moins probable).

La première étape est déterminer ce qui se passe.

1. Si vous utilisez X 4.0 (les utilisateurs de X 3.* passeront à l'étape 2), regardez la sortie de la commande **`X -probeonly`**. Vous verrez :

   ```
   (II) XXXXXX: direct rendering enabled
   ```

   ou

   ```
   (II) XXXXXX: direct rendering disabled
   ```

   où XXXXXXX dépend de la carte vidéo que vous possédez. Si le rendu direct est désactivé, alors votre configuration X est en cause, et pas le jeu. Vous devez déterminer pourquoi le DRI est désactivé. L'outil le plus important est le « Guide de l'utilisateur DRI » (*DRI Users Guide*). C'est un document très bien écrit qui vous donne des informations pas à pas sur la façon de configurer correctement le DRI pour votre machine. Une copie est disponible sur http://www.xfree86.org/4.0/DRI.html.

   Notez que si vous réussissez ce test, votre système est *capable* de faire du rendu direct. Vos bibliothèques peuvent toujours être en cause. Passez donc à l'étape 2.

2. Il existe un programme appelé **glxgears** qui accompagne le paquet « mesademos ». Vous pouvez obtenir mesademos sous Debian (**apt-get install mesademos**) ou vous pouvez chercher le rpm sur http://www.rpmfind.net. Vous pouvez également télécharger les sources depuis le site officiel de mesa et les compiler vous-même.

   L'exécution de **glxgears** montrera des pignons en rotation. La **xterm** depuis laquelle vous exécutez **glxgears** affichera « X frames in Y seconds = X/Y FPS » (X images en Y secondes). Vous pouvez comparer votre système avec la liste de bancs d'essais ci-dessous.

   ```
   CPU TYPE      VIDEO CARD      X VERSION    AVERAGE FPS
   ```

   Compiler les modules Mesa et DRI vous-même peut vous faire gagner 15 images par seconde, une grosse augmentation de performances ! Donc, si vous obtenez, disons, 20 images par seconde de moins qu'une machine comparable, il est possible que **glxgears** utilise le rendu logiciel. En d'autres termes, votre carte graphique n'accélère pas les graphiques 3D.

   Plus important encore que le nombre d'images par seconde, est la non-variation de ce nombre pour les petites et les grandes fenêtres. Si l'accélération matérielle fonctionne, le nombre d'images par seconde pour **glxgears** devrait être pratiquement indépendant de la taille de fenêtre. Si ce n'est pas le cas, alors vous ne bénéficiez d'aucune accélération matérielle.

9.2. L'accélération matérielle ne fonctionne que pour root

9.2.1. XFree86 4.x

Si les lignes suivantes ne sont pas présentes dans votre fichier `XF86Config-4`, placez-les y :

```
Section "DRI"
        Mode 0666
EndSection
```

Cela permet aux utilisateurs non-root d'utiliser le DRI. Pour les paranoïaques, il est possible de restreindre l'utilisation du DRI à seuls quelques utilisateurs non-root. Voyez le *DRI User Guide*.

9.2.2. XFree86 3.x

9.2.2.1. Cartes Voodoo

L'accélération matérielle pour les cartes Voodoo a lieu *uniquement* en couleurs 16 bits et échoue silencieusement lors du démarrage de X avec un autre nombre de couleurs.

De plus, les cartes Voodoo ont besoin du module noyau `3dfx.o` et d'un fichier de périphérique `/dev/3dfx` (majeur 107, mineur 0) pour l'accélération matérielle pour les utilisateurs normaux (non root). Ni le module ni le fichier de périphérique ne sont utilisés sous XFree86 4.x.

10. Émulation et machines virtuelles

Linux est beaucoup critiqué du fait de l'absence de la profusion de jeux présents sous d'autres plates-formes. Franchement, il y a assez de jeux pour moi, même s'il serait très chouette d'avoir certains des jeux du dernier cri et des classiques comme *Half-life*™ et *Carmageddon*™. Heureusement, il y a plus d'émulateurs que vous ne pouvez en tester. Bien que jouer à un jeu émulé n'est pas aussi amusant que de le jouer sur la machine originale, et que faire fonctionner correctement certains émulateurs peut s'avérer difficile, ils existent, et il y en a beaucoup !

10.1. Qu'est-ce qu'une machine virtuelle ?

Un « véritable ordinateur » fournit beaucoup de choses à un système d'exploitation : CPU, canaux d'entrée-sortie, mémoire, un BIOS pour fournir un accès de bas niveau à la carte mère et aux ressources d'entrée-sortie, et cætera. Quand un système d'exploitation veut écrire sur un disque dur, il communique par l'intermédiaire d'un pilote de périphérique qui fait l'interface directe avec le matériel.

Néanmoins, il est possible de donner à un programme toutes les ressources matérielles dont il a besoin. Quand il veut accéder à un disque dur, donnons-lui de la mémoire où écrire. Quand il veut attribuer une IRQ, donnons-lui l'impression d'avoir attribué une IRQ. Si vous faites ceci correctement alors, en principe, la pauvre application est incapable de savoir si elle accède réellement au matériel ou si on la trompe en lui donnant des ressources qui simulent le matériel. Une machine virtuelle est l'environnement qui trompe les applications en leur faisant croire qu'elles tournent sur une machine réelle. Elle fournit tous les services offerts par un véritable ordinateur.

Les machines virtuelles ont été à l'origine utilisées dans les années 1960 pour émuler les systèmes d'exploitation à temps partagé. De nos jours, nous les utilisons pour exécuter des logiciels qui ont été conçus pour des systèmes d'exploitation différents, ou plus communément, pour émuler un système d'exploitation entier. Du fait de la nature des machines virtuelles, le système d'exploitation étranger ne peut faire la différence entre fonctionnement à l'intérieur d'une machine virtuelle et fonctionnement dans une « vraie » machine.

10.2. Apple 8 bits

Tous les émulateurs Apple][8 bits requièrent une copie de la ROM originale, quel que soit le système que vous voulez émuler, dans un fichier. Si vous cherchez suffisamment bien, vous pouvez trouver des copies des ROM pour les Apple][,][+,][e,][c et //gs. Ils sont toujours propriété de Apple, et vous ne pouvez les utiliser légalement que si vous possédez réellement un de ces ordinateurs.

10.2.1. KEGS

KEGS, de Kent Dickey `<kentd CHEZ cup POINT hp POINT com>`, est un émulateur Apple II qui a été écrit à l'origine pour HP-UX, mais amélioré et taillé pour Linux. Il tourne sous X pour n'importe quel nombre de couleurs, et supporte des tailles de mémoire variables, les joysticks et le son. KEGS amorce toutes les variantes de Apple II, et prend en charge tous les modes graphiques des Apple][. Je n'arrive pas à trouver de page d'accueil pour cette application.

10.2.2. apple2 et xapple2

apple2 basé sur SVGAlib et xapple2 utilisant X peuvent émuler n'importe quelle variante de Apple][sauf le //gs. L'interface est assez originale, mais utilisable. La configuration est aussi un peu étrange ; cet émulateur bénéficierait d'un outil de configuration basé sur SVGA ou X. Il supporte la partie non documentée du jeux d'instructions 6502 sur laquelle se basent certains jeux. apple2 est actuellement maintenu par Michael Deutschmann `<michael CHEZ talamasca POINT ocis POINT net>` et semble être développé à une allure lente mais constante. Je ne pense pas que cette application ait une page d'accueil.

10.3. DOS

10.3.1. dosemu

dosemu [http://www.dosemu.org] est l'émulateur DOS canonique sous Linux. Quand vous pensez à DOS, ne pensez pas à des choses comme PROCOM PLUS OU D'AUTRES PROGRA~1 QUI ONT DES NOMS COURTS ET QUI SONT TOUS EN MAJUSCULES. Quelques classiques ont été écrits pour DOS comme *Carmageddon*™, *Redneck Rampage*™ et *Tomb Raider*™. dosemu peut les faire tourner. Malheureusement, il peut être malaisé de le faire fonctionner et, depuis janvier 2002, le code audio est quelque peu défectueux. Pas un gros problème si vous essayez d'exécuter *Wordperfect*™ ou une vieille application de base de données, mais ça empêche de jouer en pratique. Parvenir à faire fonctionner correctement dosemu n'est pas facile, mais c'est malheureusement le mieux qu'on puisse faire pour les jeux DOS. Bonne chance. Si vous utilisez avec succès dosemu, prévenez-moi.

10.4. Win16

10.4.1. Wabi

Wabi est un émulateur Win16 commercial, c.-à-d. qu'il exécute des applications Windows 16 bits prévues pour un environnement Windows 3.1, Windows 3.11 ou Windows for Workgroups 3.11. Wabi a été initialement créé par SCO Unix il y a longtemps et a été acheté par Caldera un beau jour vers la mi-2001.

Wabi est rapide et fait bien son boulot, même si j'ai entendu dire qu'il est plus stable sous Solaris que sous Linux. Il pourrait être utile pour jouer à de plus anciens jeux Win16, mais il y a trois problèmes :

- vous devez posséder une copie légale de Windows 3.1/3.11 ou de Windows for Workgroups 3.11 ;
- Wabi est affreusement cher pour ce qu'il fait ;
- Wabi ne fonctionne pas en couleurs 24 ou 32 bits.

Wabi ne gère *pas* DOS par lui-même, mais il semble qu'il puisse utiliser un émulateur DOS comme dorsal pour exécuter des programmes DOS. On a parlé d'un Wabi 3.0 qui aurait effectué de l'émulation Win32, mais pour autant que je sache, ce projet est tombé aux oubliettes. Je pense que Wabi fonctionne sous Linux sur toutes les architectures (quelqu'un peut-il le vérifier ?).

10.5. Win32

10.5.1. wine

wine [http://www.winehq.com], qui porte l'acronyme GNUide de *Wine Is Not An Emulator* (Wine n'est pas un émulateur) est une implémentation non commerciale de l'API Win32. La raison pour laquelle ce n'est pas un émulateur est subtile et pas du plus grand intérêt pour la plupart des non-informaticiens, et nous parlerons donc d'émulateur ici (il s'agit en fait d'une traduction au moment de l'exécution des appels de l'API Win32 en appels POSIX/X11). wine a beaucoup évolué, et est capable d'émuler beaucoup de jeux importants, ce qui est une bonne nouvelle pour les utilisateurs Linux intéressés.

wine ne fournit `pas` d'API DOS, et vous ne pouvez donc pas l'utiliser pour exécuter des applications DOS. Pour cela, vous devriez jeter un œil à dosemu. wine n'a jamais très bien implémenté DirectX, bien que quelques jeux fonctionnent sous wine. Pour les jeux, vous devriez vous tourner vers winex.

En plus de la traduction au moment de l'exécution de l'API Win32 vers POSIX/X11 (il exécute des applications Windows sous Linux), wine effectue également une traduction au moment de la compilation de l'API Win32 vers POSIX/X11 (il compile le code source d'une application Windows sous Linux). Vu sous cet angle, wine est un utilitaire de portage Windows-vers-Linux. L'architecture x86 n'est pas requise, mais est recommandée car elle permet une exécution binaire x86 réelle ainsi qu'une utilisation directe des DLL.

Vous pouvez utiliser wine « avec Windows », ce qui signifie qu'il utilise des bibliothèques qui proviennent en réalité de Microsoft Windows lui-même. Cela n'est légal que si vous possédez une copie de Windows qui n'est pas actuellement utilisée sur un ordinateur. On dit que wine fonctionne le mieux quand il est exécuté avec Windows. Vous pouvez également utiliser wine sans Windows. Les gens de winehq [http://www.winehq.com] écrivent leur propre jeu de bibliothèques appelé `libwine` qui implémente l'API Win32 sans aucun code provenant de Microsoft.

wine était à l'origine placé sous licence MIT/X11, et pouvait donc être utilisé à la fois à des fins commerciales et non commerciales. À la mi-2002, des parties de wine sont passées à la LGPL afin de ne plus pouvoir être utilisées à des fins commerciales. Cela pose un problème à des sociétés comme Transgaming (Section 10.5.3, « winex ») et a ouvert la voie à un nouveau projet issu de wine appelé ReWind.

10.5.2. rewind

rewind [http://rewind.sourceforge.net/] a été démarré par Éric Pouech (un développeur de wine) et Ove Kåven (un développeur de winex) en réponse au changement de licence de wine. Il a vu le jour comme un instantané de la dernière version de wine complètement placée sous la licence MIT/X11. Le but est que rewind demeure sous licence MIT/X11 afin que des sociétés comme Transgaming puissent offrir des produits dérivés de wine.

10.5.3. winex

winex est publié par une société appelée Transgaming [http://www.transgaming.com]. Ses développeurs utilisent wine et y ajoutent le support DirectX/DirectDraw. Bien que winex soit commercial, leur modèle économique est intéressant.

L'utilisateur final (vous) peut télécharger le code source gratuitement. Néanmoins, pour 5 $ US par mois, vous pouvez devenir un abonné de Transgaming, ce qui procure trois avantages principaux :

- Les abonnés peuvent à tout moment télécharger des versions empaquetées de winex dans le format `deb`, `rpm` ou `tar.gz` (y compris les mises à jour). Elles sont également plus fonctionnelles que le source publiquement disponible : celui-ci est une version antérieure qui ne dispose pas de certaines des fonctionnalités les plus récentes, comme la prise en charge des programmes protégés contre la copie.
- Les utilisateurs abonnés peuvent indiquer lors de sondages mensuels quels sont les points qu'il faut améliorer en priorité dans winex. Par exemple, ils peuvent voter pour des choses comme « Améliorer la prise en charge des programmes protégés contre la copie », « Meilleur support d'Installshield » ou « Améliorer la prise en charge de DirectX 8.0 ». Il me semble que les développeurs écoutent réellement les sondages.
- Le site web de Transgaming comporte quelques forums d'assistance aux utilisateurs. D'un côté, ils utilisent le format le plus affreux, horrible, confus, dispendieux et idiot qu'il m'ait été donné de voir, et j'espère bien ne plus jamais revoir de forum ayant un format aussi mauvais que celui de Transgaming. D'un autre côté, vous pouvez demander de l'aide et les développeurs sont *très* bons pour trouver une réponse à votre question ; leur vigilance est assez impressionnante. Les non abonnés peuvent parcourir les forums, mais seuls les abonnés peuvent y écrire (et, par conséquent, y demander de l'aide).

Les développeurs de winex avaient l'intention de publier périodiquement leurs améliorations à Installshield, DirectX et DirectDraw dans wine. En contrepartie, au fur et à mesure de la maturation de wine, les développeurs de winex auraient pris les nouvelles versions de wine pour les utiliser dans winex. Néanmoins, depuis la naissance de Transgaming, des parties de wine sont passées à la licence plus restrictive GNU LGPL. Cela signifie en gros que les versions de wine publiées après la date du changement de licence ne peuvent plus être utilisées par winex. Par conséquent, winex sera à présent basé sur rewind.

10.5.4. Win4Lin

Win4Lin [http://www.netraverse.com] est un produit commercial de Netraverse. Comme vmware, il utilise l'approche de la machine virtuelle pour exécuter des applications Windows, et affiche donc une grande fenêtre depuis laquelle vous pouvez démarrer Windows et exécuter toutes sortes d'applications Windows. À la différence de vmware, Win4Lin ne prend en charge que Windows 95/98/ME, mais cela s'avère être mieux pour les joueurs. Puisque Win4Lin se concentre sur ces systèmes d'exploitation, on dit qu'il est plus rapide et exécute mieux les jeux sous ces systèmes d'exploitation que vmware. Il est également bien moins cher que ce dernier. Win4Lin, dont la version la plus récente au 30 juin 2003 est la 5.0, souffre néanmoins de certaines limitations :

- Il ne prend pas en charge DirectX ou DirectDraw, alors que vmware a un support « limité » pour DirectX.
- Il ne prend en charge que les périphériques série et parallèle. C'est important pour ceux qui utilisent des joysticks USB. Notez que vmware peut gérer jusqu'à 2 périphériques USB.
- Au 30 juin 2003, comptez 89.99 $ sans documentation imprimée et 99.99 $ avec. De plus, il n'y a pas de copie d'évaluation disponible, bien qu'il y ait une garantie de remboursement sous 30 jours. Néanmoins, puisque c'est commercial, vous avez le support technique. vmware est beaucoup plus cher.
- Comme pour vmware, vous devez posséder une copie autorisée de Win95 ou Win98. Win4Lin ne peut utiliser une installation existante de Windows à la manière de wine.
- Il ne tourne que sur les architectures x86.

10.5.5. VMWare

VMWare [http://www.vmware.com] est une machine virtuelle qui peut exécuter plusieurs systèmes d'exploitation simultanément sur un PC standard : les systèmes d'exploitation pris en charge comprennent ceux de Microsoft, Linux, Novell Netware et FreeBSD. Vous pouvez entre autres l'utiliser pour exécuter un système d'exploitation MS Windows et y lancer votre jeu favori. Vous pouvez même faire tourner Linux sous Linux ; utile par exemple si vous voulez tester une autre distribution. Stupéfiant ! Mais il y a des mauvais côtés. Vous devriez assurément disposer d'une bonne configuration pour l'utiliser ; le minimum annoncé est un CPU x86 500 Mhz avec 128 Mo de RAM, mais un processeur plus rapide avec au moins 256 Mo de RAM semble le minimum absolu si vous désirez des performances raisonnables. Toutes les distributions Linux ne sont pas prises en charge : les dernières RedHat, Mandrake et Suse le sont, mais c'est pas de chance si vous avez une autre version et/ou distribution (comme Debian). De plus, la prise en charge par vmware de DirectX est limitée, et vous pourriez ne pas pouvoir jouer à des jeux récents.

Voyez http://www.vmware.com pour plus d'informations. Ce n'est pas bon marché (environ 300 $ pour la version Workstation), mais vous pouvez obtenir une version d'évaluation limitée à 30 jours.

10.5.6. Que choisir ?

En premier lieu, vous devriez essayer un émulateur. Bien que certains jeux fonctionnent sous wine, vous aurez probablement le plus de succès avec winex : sa prise en charge de DirectX s'améliore constamment. Dans la version 3.1, la prise en charge de DirectX 8 est quasiment achevée, mais ce n'est pas forcément le cas des versions plus anciennes de DirectX (et donc des jeux plus anciens).

Vous pourriez également essayer une machine virtuelle comme Win4Lin ou VMWare au lieu d'un émulateur. Si votre but est d'exécuter des applications Win95/98/ME sous Linux, sans USB et sur l'architecture x86, le coût et le centrage sur les systèmes d'exploitation de type Win95 de Win4Lin en font un meilleur choix que vmware. Néanmoins, si vous devez avoir la prise en charge de l'USB ou exécuter Linux sur une plate-forme autre que x86, vmware est votre seule possibilité.

Maintenant, si votre but est d'exécuter des jeux pour des systèmes d'exploitation de type Win95 sous Linux, Win4Lin semble presque toujours meilleur que vmware. Le plus gros problème est que vmware a un support limité de DirectX alors que Win4Lin n'en a aucun. Ce fait seul rend tant Win4Lin que vmware inutilisables pour la plupart des jeux un tant soit peu évolués. Mais si vous voulez essayer, vous aurez probablement plus de succès avec vmware.

11. Interpréteurs

11.1. Moteur SCUMM (LucasArts)

Lucasarts a écrit un moteur pour les aventures pilotées à la souris nommé SCUMM (*Script Creation Utility for Maniac Mansion*). Ils ont écrit beaucoup d'aventures graphiques en utilisant SCUMM, comme leur célèbre série *Monkey Island*™ (tous les trois). Ludvig Strigeus `<strigeus CHEZ users POINT sourceforge POINT net>` a pu utiliser la rétro-ingénierie pour comprendre le format SCUMM et écrire un interpréteur pour les jeux l'utilisant qui compile sous Linux et Win32. Il s'agit de scummvm [http://scummvm.sourceforge.net/]. Leur site web est très bon, et regorge d'informations sur SCUMM et l'utilisation de scummvm pour ce type de jeux.

Une page de compatibilité peut être trouvée sur le site web de scummvm. Ça vaut ce que ça vaut, mais j'ai pu finir beaucoup des jeux qui sont listés à 90 % sans le moindre problème. scummvm est très robuste, et vous permet d'acheter des jeux Lucas Arts utilisant le format SCUMM, copier les fichiers de données sur votre disque dur et les jouer sous Linux. En février 2002, j'ai suivi leur CVS, et ce projet subit un développement constant. Gloire à l'équipe de scummvm.

11.2. AGI : Adventure Gaming Interface (Sierra)

Les anciens jeux d'aventures graphiques DOS de Sierra utilisaient un langage de script appelé AGI (*Adventure Gaming Interface*). Quelques jeux écrits en AGI : *Leisure Suit Larry I*™ (EGA), *Space Quest I*™, *Space Quest II*™, *King's Quest II*™, *Mixed-Up Mother Goose*™, et cætera. On peut y jouer en utilisant sarien [http://sarien.sourceforge.net], un interpréteur *open source* pour les jeux AGI.

Sarien a été écrit en SDL, et devrait donc tourner sur toute plate-forme qui peut compiler des programmes SDL. De plus, il y a des versions pour DOS, les PDA utilisant Strong-Arm, QNX (mon dieu ! du jeu embarqué !), les systèmes à base de MIPS et les Pocket PC à base de SH3/4. Les développeurs ont clairement perdu la tête (d'une bonne façon !). Sarien propose de nombreuses améliorations non trouvées dans les jeux originaux, comme une console déroulante du genre de celle de Quake, un visualisateur d'images et de dictionnaire, un son amélioré et la prise en charge de AGDS, un clone russe de AGI. Sarien est en cours de développement et les développeurs ont très bien documenté la conception interne de Sarien si jamais quelqu'un veut s'impliquer dans son développement.

11.3. SCI : SCript Interpreter ou Sierra Creative Interpreter (Sierra)

Les jeux d'aventures graphiques plus récents de Sierra (c.-à-d. de la fin des années 80) utilisaient un interpréteur appelé SCI. Il y a beaucoup de versions de SCI puisque Sierra a constamment amélioré son moteur. Les premiers jeux SCI s'exécutaient sous DOS, mais Sierra s'est finalement tourné vers Win32. Quelques exemples de jeux écrits avec SCI : *Leisure Suit Larry 1*™ (VGA), *Leisure Suit Larry 2-7*™, *Space Quest 3-6*™, *King's Quest 4-6*™, *Quest For Glory 1-4*™ et beaucoup d'autres. Comparées aux jeux AGI, les aventures SCI ont une meilleure prise en charge de la musique, un moteur plus complexe et des tas de fioritures.

Beaucoup de jeux utilisant SCI (écrits en SCI0) peuvent être joués en utilisant freesci, disponible sur http://freesci.linuxgames.com. Comme *Sarien*™, FreeSCI peut utiliser beaucoup de cibles graphiques incluant SDL, xlib et GGI, de sorte que ce programme peut compiler et s'exécuter sous un nombre incroyable de plates-formes. Les développeurs ont très bien documenté leur application.

11.4. Infocom Adventures (Infocom, Activision)

La Z-machine [http://www.gnelson.demon.co.uk/zspec/index.html] est une machine virtuelle bien documentée conçue par Infocom pour exécuter leurs jeux de fiction interactive. Cela leur permet d'écrire des fichiers de données de jeu d'une façon multi-plates-formes, car seul le moteur lui-même, la Z-machine, est dépendant de la plate-forme. La Z-machine a subi différentes évolutions durant la vie de Infocom, et deux révisions supplémentaires (V7 et V8 créées par Graham Nelson) après la mort de Infocom. Les versions ultérieures disposaient même d'une prise en charge limitée du son et des graphiques !

Un des interpréteurs de Z-machine parmi les plus populaires est Frotz [http://www.cs.csubak.edu/~dgriffi/proj/frotz/]. Leur excellente page d'accueil regorge de liens sympas pour les amateurs de fiction interactive. Frotz est placé sous GPL, exécute toutes les versions de la Z-machine et compile sur la plupart des versions de Unix. Frotz est à l'origine de nombreuses variantes, comme une version pour PalmOS et les PDA à base de Linux.

jzip [http://jzip.sourceforge.net/] est un autre interpréteur de Z-machine très populaire qui exécute les fichiers de données des Z-machine V1-V5 et V8. jzip est très portable ; il compile sous tous les Unix, OS/2, Atari ST et DOS.

Il y a en fait beaucoup d'autres interpréteurs de Z-machine comme nitfol et rezrov (écrit en Perl !). Chaque interpréteur a ses points forts, et vous pouvez trouver des liens les référençant sur les pages d'accueil de Frotz et jzip.

11.5. Scott Adams Adventures (Adventure International)

On peut dire que Scott Adams est le père de la fiction interactive. Bien qu'il ait lui-même été inspiré par la première œuvre de fiction interactive, *Adventure*™, Scott a propulsé l'aventure au devant de la scène. Ses jeux étaient disponibles pour Atari, Apple 2, Commodore, Sorcerer, TI et CPM. Sa société, Adventure International, a publié quelques jeux très appréciés entre 1978 et 1984 avant d'arrêter. Elle a récemment publié un nouveau jeu (une version Linux n'est pas disponible) mais depuis le déclin du jeu d'aventure, elle s'est plutôt tenue à l'écart de l'industrie du jeu.

Alan Cox a écrit scottfree, un interpréteur de fichier de jeux d'aventure du type Scott Adams pour Unix. En utilisant scottfree et l'un des fichiers de données du type Scott Adams qui peuvent être téléchargés depuis le site web de Scott [http://www.msadams.com/], vous pouvez vous délecter de ces classiques.

11.6. Ultima Underworld : The Stygian Abyss (Origin, Blue Sky Productions)

Le projet Underworld Adventures [http://uwadv.sourceforge.net/] est un effort visant à porter le classique de 1992, *Ultima Underworld: The Stygian Abyss*™ sur des systèmes d'exploitation modernes comme Linux, MacOS X et Windows. Il utilise OpenGL pour les graphiques 3D, SDL pour les tâches spécifiques à la plate-forme et est publié sous la GNU GPL. Underworld Adventures fournit un système graphique impressionnant qui utilise les fichiers de jeu originaux, et vous avez donc besoin du disque de jeu original pour jouer.

Underworld Adventures offre également un tas d'outils pour afficher les cartes de niveaux, examiner les scripts de conversation uw1 et bien d'autres choses.

11.7. Ultima 7 (Origin, Electronic Arts)

Ultima 7™ en en fait constitué de 2 jeux : la partie I (*The Black Gate*™) et la partie II (*Serpent Island*™) qui utilise une version légèrement améliorée du moteur de *The Black Gate*™. De plus, une disquette additionnelle a été publiée à la fois pour la partie I (*The Forge Of Virtue*™) et la partie II (*The Silver Seed*™).

Une équipe a développé exult [http://exult.sourceforge.net/], un interpréteur *open source* permettant d'exécuter les deux parties de *Ultima 7*™ et leurs disquettes additionnelles. Exult est écrit en C++ et utilise SDL, et compilera donc sur toute plate-forme pouvant compiler des programmes SDL. Il offre également certaines améliorations par rapport aux versions originales du moteur de *Ultima VII*™. Vous devrez acheter une copie de *Ultima 7*™ pour pouvoir jouer. Les développeurs n'ont pas l'intention d'étendre Exult pour interpréter les autres Ultima étant donné que les moteurs ont changé radicalement entre les différentes versions.

L'équipe d'Exult a également travaillé dur pour créer un éditeur de niveaux, *Exult Studio*™, et un compilateur de scripts qui permettra aux utilisateurs de créer leurs propres RPG dans le style *Ultima*™.

11.8. System Shock (Electronic Arts, Origin)

System Shock™ est un jeu de combat à la première personne/aventure classique datant de 1994, ce qui le place en contemporain de *Doom*™. Néanmoins, son moteur est beaucoup plus évolué que celui du *Doom*™ original : par exemple, *System Shock*™ disposait d'acteurs (*sprites*) 3D, d'un angle de vue modifiable (à la souris), et de la possibilité d'empiler des objets, donnant l'illusion d'une carte 3D complète, comme *Quake*™. Les critiques que j'en ai lues sont très positives. Elles concordent pour dire que ce jeu a les fonctionnalités de *Quake*™ avec une intrigue et un scénario plus convaincants que *Half-life*™. Le moteur de *System Shock*™ était optimisé pour la sophistication, alors que celui de *Doom*™ était optimisé pour vous balancer des tas de monstres à la figure : une approche complètement différente. Très impressionnant pour un vieux jeu !

Le *System Shock Hack Project* [http://madeira.physiol.ucl.ac.uk/tsshp/sshock.html] est une tentative de mise à jour du jeu pour les systèmes d'exploitation modernes. Ce projet utilise SDL, et est publié sous une licence BSD modifiée. Bien que vous ayez besoin des fichiers de jeu originaux pour jouer à SSHP, il devrait fonctionner avec la version de démonstration de *System Shock*™, qui est disponible gratuitement.

12. Sites web et ressources

12.1. Méta-sites web de jeux

Voici quelques ressources générales pour les joueurs Linux :

Le *Linux Game Tome* : http://www.happypenguin.org	Les jeux en eux-mêmes.
Linuxgames [http://www.linuxgames.com]	Actualités sur les jeux Linux.
http://www.holarse.net	Méta-site web sur les jeux Linux pour les germanophones.
Mobygames [http://www.mobygames.com]	Une base de données de tous les jeux vidéo informatiques connus. C'est un site très complet et un des mes favoris.

12.2. Jeux Linux commerciaux

12.2.1. Où acheter des jeux commerciaux

ebgames [http://www.ebgames.com] ne vend plus de logiciels Linux. Ils ont cessé de vendre des jeux et des distributions Linux à peu près au même moment où Loki Software a fait aveu de faillite, ce qui est une honte car ils avaient les plus bas prix sur les jeux Linux qu'il m'ait été donné de voir. Néanmoins, de temps en temps, ils proposent des choses comme Code Warrior ou Red Hat Linux.

Tux Games [http://www.tuxgames.com]	Votre magasin pour l'achat de jeux Linux commerciaux (les vendeurs de logiciel comme Tribsoft et Loki ont aussi des magasins en ligne sur leurs sites web).

12.2.2. Qui publiait des jeux pour Linux

Ce sont des sociétés qui publiaient des jeux pour Linux mais qui pour des raisons diverses ne sont plus actives dans l'industrie du jeu sous Linux.

Loki Software : http://www.lokigames.com	En tant que société qui a apporté *CTP*™ et *Quake3*™ sous Linux, Loki était le père du jeu sous Linux. Ils étaient des pionniers et avaient, de loin, le plus de titres (je les ai tous). Loki a porté des jeux sous Linux, principalement en utilisant la bibliothèque SDL. La mort de Loki en janvier 2002 était le plus grand revers subi par Linux dans sa tentative de conquête du marché domestique. Linuxgames.com propose un bel historique de Loki sur http://www.linuxgames.com/articles/lokitimeline/

http://www.tribsoft.com	Tribsoft a publié *Jagged Alliance 2*™, un excellent jeu de rôle/stratégie qui a réclamé plus de 2 semaines de ma vie. Ils devaient publier *Europai Universalis*™, *Majesty*™ et *Unfinished Business*™. Néanmoins, en date du 3 janvier 2001, Mathieu Pinard de Tribsoft a dit qu'il faisait une pause et que Tribsoft ne publierait plus de jeux pour le moment. Il continuera toujours l'assistance pour *JA2*™ mais ne vous attendez pas à des correctifs ou à des mises à jour.
MP Entertainment : http://www.hopkinsfbi.com	MP Entertainment a publié *Hopkins FBI*™, mon jeu préféré jamais publié pour Linux. Plus violent que *Quake*™. Plus de nudité que Hustler. Plus de mauvais goût que Liberace [http://www.flatwaremedia.com/liberace/gallery.cfm]. C'est une bande dessinée sur votre moniteur. Il était prévu qu'ils publient *Hopkins FBI II*™ et quelques autres titres, mais les annonces datent déjà de quelques années sans aucun signe d'arrivée prochaine du jeu. Ils ont ignoré toutes mes demandes d'informations, et j'en ai donc conclu que MP Entertainment est dans la même situation que Tribsoft. Vous pouvez toujours acheter ou télécharger une démo de *Hopkins FBI*™ depuis leur site web. Si quelqu'un a plus d'informations sur cette société ou sur l'auteur de *Hopkins FBI*™, veuillez me contacter.
Phantom EFX : http://www.phantomefx.com	Ils proposent *Reel Deal Slots*™, qui est très bien fait ! Je ne suis pas trop amateur de jeux de cartes ou d'argent, mais ce jeu est impressionnant ! Étant donné que leur seul spécialiste Linux a quitté la société, *Reel Deal Slots*™ est leur seule, et à ce jour dernière, publication pour Linux.

12.3. Autres ressources

Cette section propose des URL qui devraient être mentionnées mais qui ne trouvaient pas leur place dans une section séparée à l'intérieur de ce guide, et que j'ai donc listées ici dans une sorte d'annexe.

Linux Game Publishing : http://www.linuxgamepublishing.com	Linux Publishing ne vend pas directement au public, mais fournit des services professionnels de publication de jeux aux éditeurs de jeux. Je pense que cela signifie la réplication de disques, l'emballage et la vente aux détaillants.
Site officiel de XFree86 : http://www.xfree86.org	Page d'accueil de XFree86.
Linux Game Development Center : http://lgdc.sunsite.dk/index.html	C'est le site web de référence pour qui veut programmer des jeux sous Linux. C'est une montagne d'informations contenant des articles bien écrits sur tous les aspects de la programmation de jeux (pas nécessairement spécifiques à Linux), des liens vers d'importantes ressources relatives à la programmation de jeux, des interviews, tests, sondages et des tas d'autres trucs. Il est difficile d'imaginer un meilleur site web sur le sujet.
La FAQ de Linux Gamers : http://www.icculus.org/lgfaq/	Malgré le fait sidérant qu'elle ne mentionne nulle part ce présent guide comme une ressource dans leur texte, je considère la FAQ comme un bon complément à ce guide. J'ai essayé de ne garder que le strict minimum d'informations spécifiques à des jeux particuliers dans ce guide. La FAQ prend l'approche opposée : elle se concentre principalement sur les jeux eux-mêmes, y compris des problèmes spécifiques à certains jeux et l'endroit où les obtenir. La FAQ et le guide sont complémentaires à cet égard, et j'ai essayé de ne pas reproduire leur contenu. Même si les auteurs sont un peu hargneux, leur effort est à souligner. Si vous voulez une source d'informations globale pour des questions spécifiques à certains jeux, la FAQ

	est un excellent point de départ. De plus, elle donne des informations sur un assez grand nombre de jeux Linux.
Le « Guide pratique de la qualité du son sous Linux » (*Linux Audio Quality HOWTO*) : http://www.linuxdj.com/audio/quality/	Ce guide intéressera principalement les musiciens qui utilisent des cartes professionnelles ou semi-professionnelles pour l'enregistrement et la création de musique sur un ordinateur. Les informations sont très détaillées, peut-être même trop pour les joueurs.