VisuCar *par André C.*

Pour explorer des jeux de caractères et reconstituer des sprites

Etat des lieux

Depuis très longtemps, j'ai cherché un utilitaire permettant de visualiser les jeux de caractères de l'Oric, d'afficher en grand le dessin détaillé de chaque caractère et d'en révéler les fameux 8 octets de définition. Plus encore (et même surtout), j'ai désiré visualiser les sprites composés de plusieurs caractères.

Les seuls programmes qui répondent à peu près à ce cahier des charges sont les programmes de redéfinition de caractères (voir mes articles dans les CEO-mags n° 310, 311, 312, 314 et 315) et encore, pas pour la visualisation des sprites. Donc, j'ai fini par m'y coller et cela a donné VisuCar (figure 1).

La principale difficulté avec les caractères redéfinis est ... qu'ils sont redéfinis ! Je m'explique : pour être exploitable, un tel programme doit pouvoir afficher en clair les instructions disponibles ainsi que les résultats demandés par l'utilisateur. Or si beaucoup de caractères sont redéfinis, cela devient vite incompréhensible. Il fallait donc trouver une astuce pour contourner ce problème.

First, récupérer les jeux de caractères redéfinis

Dans la très grande majorité des cas, les caractères redéfinis remplacent les caractères natifs en Ram et sont donc localisés de #B500 à #B7FF pour le jeu 0 (jeu normal, 96 caractères : de Ascii 32 à Ascii 127) et de #B900 à BAFF pour le jeu 1 (jeu semi-graphique, 64 caractères de Ascii 32 à Ascii 95).

Mais lorsque les besoins en caractères redéfinis sont importants et qu'il faut malgré tout garder un minimum de caractères normaux, certains programmes ne chargent les caractères redéfinis en Ram qu'au fur et à mesure des besoins.

Enfin, il semble que certains programmes en langage machine se passent des vecteurs normaux d'affichage et il n'est pas simple de comprendre où sont passés les caractères redéfinis. C'est heureusement un cas extrêmement rare.

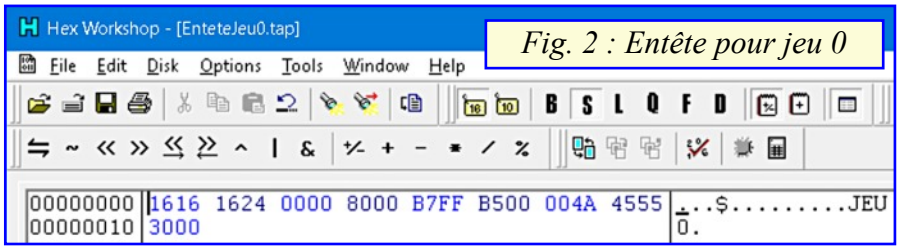

Fig. 2 : Entête pour jeu 0

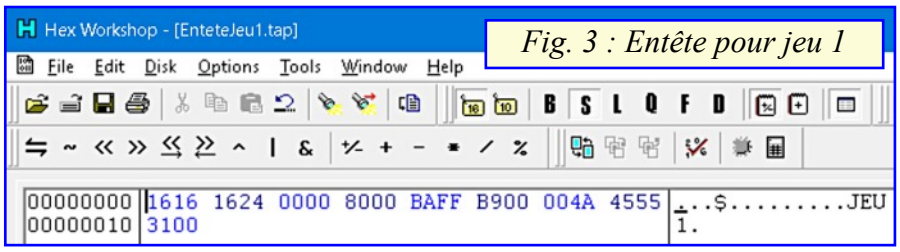

Fig. 3 : Entête pour jeu 1

En pratique, la récupération des caractères redéfinis est assez simple.

Le cas le plus facile est évidemment lorsque le programme comporte un fichier de type 'CHS' (c'est à dire contenant spécifiquement le ou les jeux redéfinis) : Ce fichier 'CHS' peut directement être exploré avec VisuCar !

Mais la plupart du temps, le pro-

Fig. 1 : Ecran-titre

Fig. 4 : Menu général

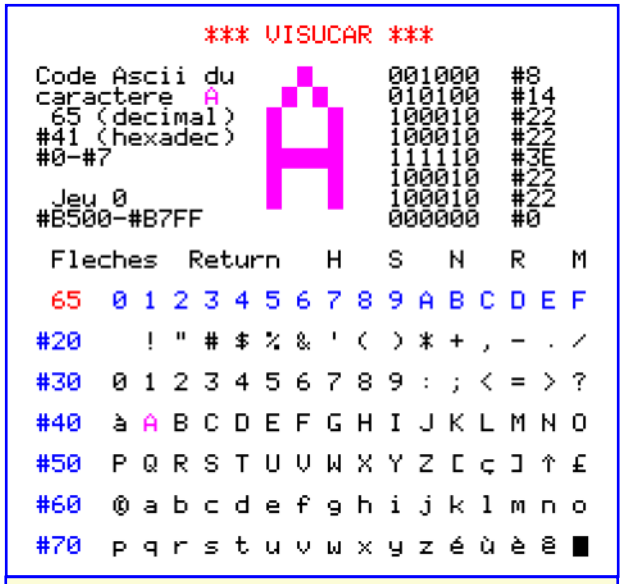

Fig. 5 : Exploration du jeu 0

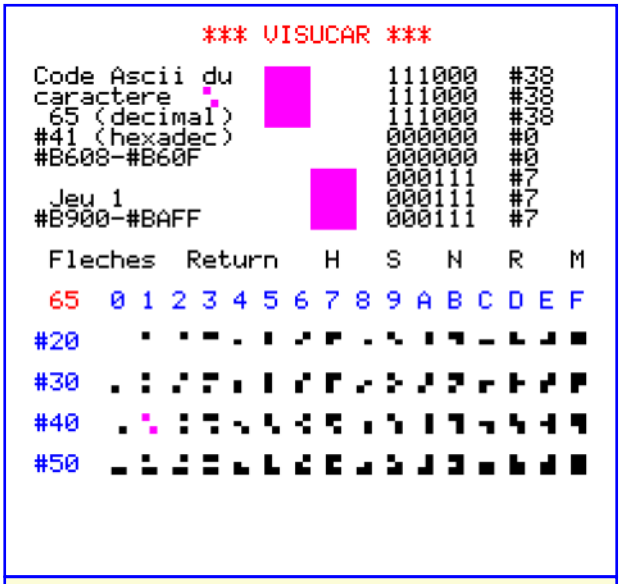

Fig. 6 : Exploration du jeu 1

gramme doit d'abord être exécuté pour mettre les caractères redéfinis en place dans la Ram. Après avoir lancé le programme et après apparition des caractères redéfinis à l'écran, deux cas de figures se présentent :
1) Un CTRL+C ou un RESET à chaud sont opérationnels et n'effacent pas la Ram : Lancez tout simplement la commande SAVE"NOM.CHS",A#B500,E#BAFF pour récupérer le pactole.
2) Le programme est protégé et efface la Ram avant de rendre la main. La solution, c'est Euphoric et sa fameuse touche F9 qui sauve un fichier DUMP contenant une copie des 64 Ko de la mémoire de l'Oric : Avec un éditeur hexadécimal, récupérez les deux zones d'offset #B500 à #B7FF pour le jeu 0 et #B900 à BAFF pour le jeu 1. Placez par devant une entête cassette de type : "16161624000008000B7FFB500004A45553000" pour le jeu 0 (figure 2, page précédente et EnteteJeu0.tap dans le zip) ou "1616162400008000BAFFB900004A45553100" pour le jeu 1, (figure 3, page précédente et EnteteJeu1.tap dans le zip). Vous disposez alors de deux fichiers .tap que vous pouvez charger avec un CLOAD avant de lancer VisuCAR ou après lancement de Visycar, avec la commande 'E' du menu principal (le transfert du .tap sur la disquette ne pose plus de problème depuis la sortie des versions 3.0 et 4.0 de Sedoric).

Au Menu de VisuCar

La figure 4 (page précédente) vous montre les commandes disponibles :
0 – Pour explorer le jeu 0, c'est à dire afficher l'ensemble des 96 caractères (avec adresses du jeu) et pour chaque caractère, le dessin agrandi au format 6x8 cases, les 8 octets de définition et leurs adresses en Ram (figure 5, ci-dessus qui montre, en exemple, le caractère A du jeu normal Natif).
1 – Idem pour explorer le jeu 1 : affichage des 64 caractères (avec adresses du jeu) et pour chaque caractère, dessin agrandi, octets de définition avec leurs adresses en Ram (figure 6, ci-dessus avec un exemple de caractère choisi parmi le jeu semi-graphique Natif.
2 – Pour assembler des caractères du jeu 0 et reconsti-

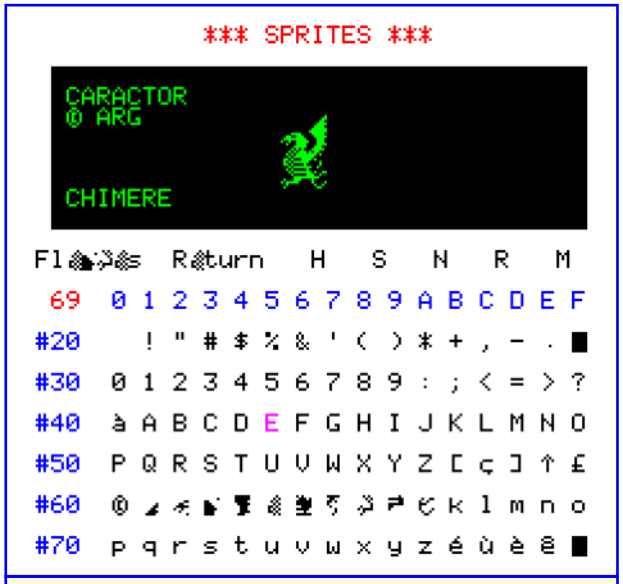

Fig. 7 : Assemblage jeu 0 redéfini

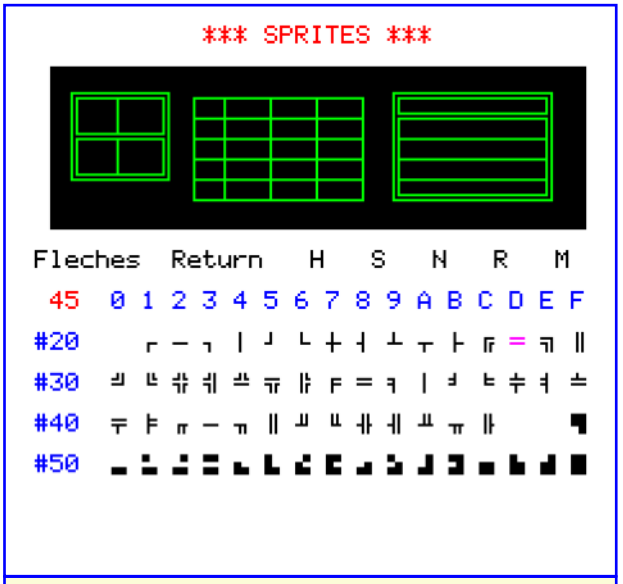

Fig. 8 : Assemblage jeu 1 redéfini

tuer un ou des sprites (figure 7, page précédente). On y voit un exemple de sprite : 'Chimère' emprunté au logiciel 'Caractor' (© ARG informatique.
3 – Idem avec le jeu 1 (figure 8, page précédente). Exemple de combinaisons de caractères pour former des cadres, tiré de 'Editecran' (© André Chéramy)
E – Pour Explorer un autre jeu de caractère, cette option permet de charger un fichier à partir de la disquette. Il est également possible de charger un fichier tap. Pour cela, au menu général de VisuCar, faire 'Q' (Quitter), CLOAD et RUN.
H – Pour obtenir de l'aide sur l'utilisation de VisuCar. Quatre écrans seront affichés avant de revenir au menu (figures 9 à 12, ci-dessous).
Q – Cette dernière option permet de quitter VisuCar.

Exploration du jeu 0 ou du jeu 1

Sur les figures 5 et 6 (page précédente), on peut voir au milieu de l'écran le mini-menu suivant : "Flèches Return H S N R M" Ce mini-menu résume les commandes possibles :
Les 4 flèches permettent de sélectionner un caractère afin d'en découvrir les détails. Par défaut le curseur pointe sur le caractère de code Ascii 65.
Un appui sur RETURN valide le choix et déclenche l'affichage de toutes les informations disponibles sur ce caractère, notamment les 8 octets de définition et leur adresse en Ram (figures 5 et 6 ,page précédente).
L'option 'H' affiche les mêmes écrans d'aide qu'au menu général et revient ensuite à l'écran qui était en cours (figures 9 à 12, ci-dessous).
La commande 'S' sauve l'écran en cours et donc les précieuses informations affichées. Le nom des fichiers successivement sauvés est de la forme 'VCDATAxxx.SCR' et il est généré automatiquement à partir du n° xxx=001. Attention ! Pensez à renommer ces fichiers avant de relancer VisuCar, sinon ils seront écrasés, car le compteur est réinitialisé à chaque RUN.
L'option 'N' remplace le jeu de caractères en cours d'exploration par le jeu Natif (non-redéfini) corres-

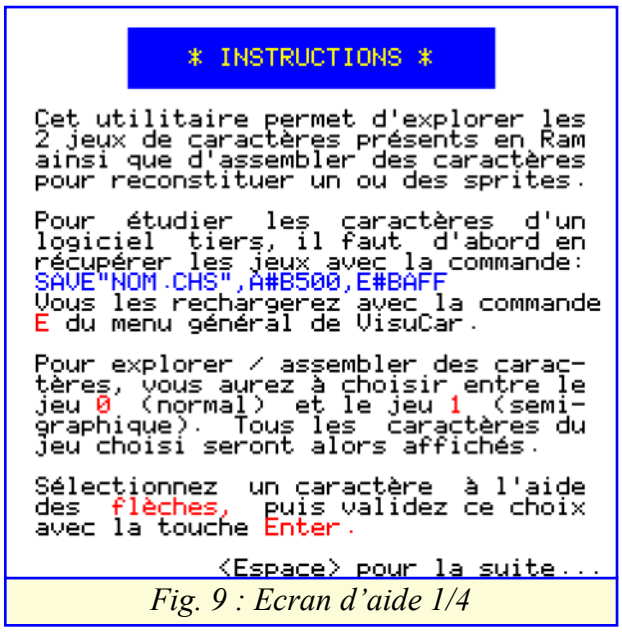

Fig. 9 : Ecran d'aide 1/4

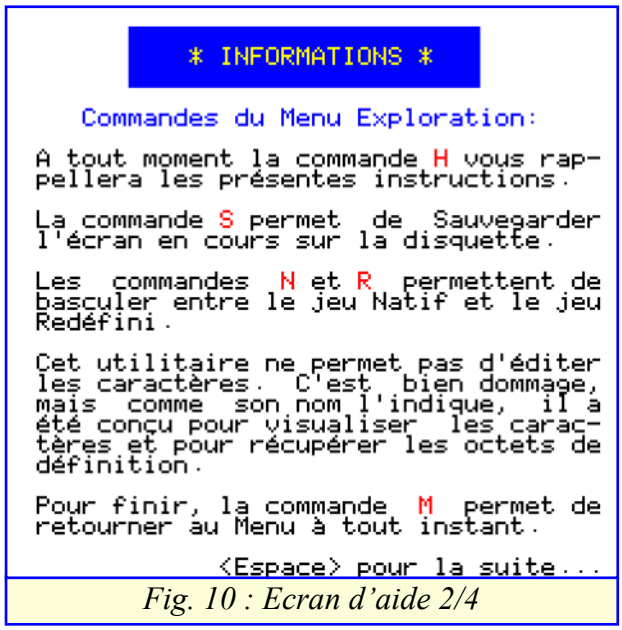

Fig. 10 : Ecran d'aide 2/4

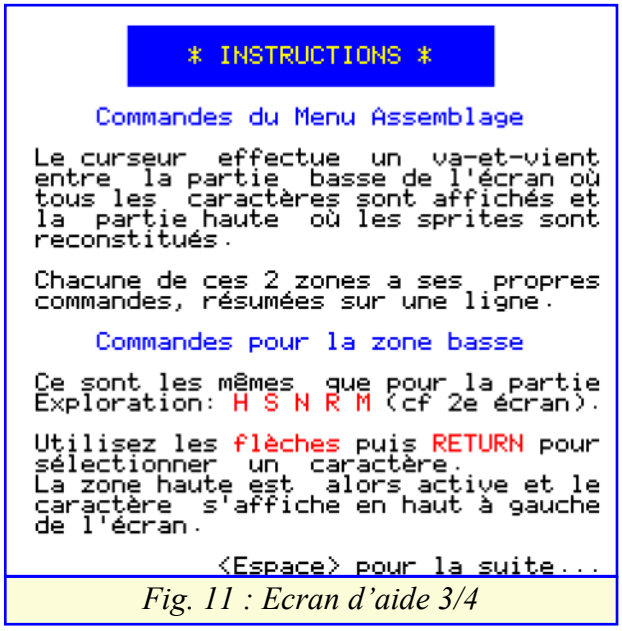

Fig. 11 : Ecran d'aide 3/4

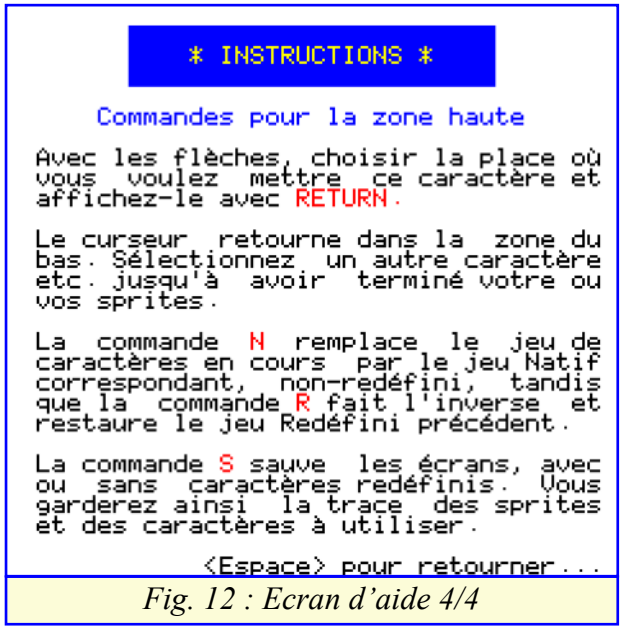

Fig. 12 : Ecran d'aide 4/4

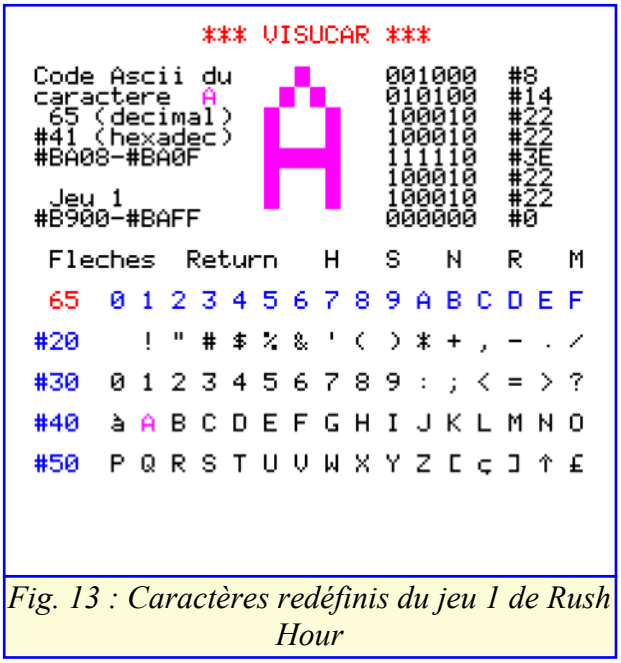

Fig. 13 : Caractères redéfinis du jeu 1 de Rush Hour

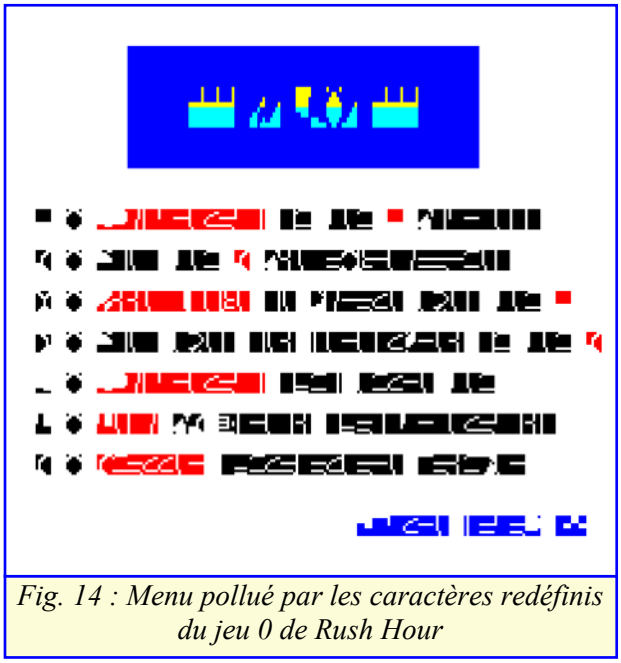

Fig. 14 : Menu pollué par les caractères redéfinis du jeu 0 de Rush Hour

pondant. Cette commande est importante, elle permet d'une part de repérer d'un seul coup d'œil les caractères qui ont été modifiés et d'autre part de décoder un écran devenu illisible quand de trop nombreux caractères ont été redéfinis (voir plus loin le cas emblématique du Rush Hour de Fabrice F.).
L'appui sur 'R' permet de Revenir à l'écran des caractères Redéfinis. Cette commande remplace le jeu natif par le jeu redéfini en cours précédemment.
Enfin la commande 'M' permet de revenir au menu et donc d'effectuer une autre tâche.

Assemblage de caractères pour former un sprite

Que ce soit avec le jeu 0 ou avec le jeu 1, la procédure d'assemblage est identique. L'écran comporte deux zones dotées de commandes similaires avec des nuances d'utilisation. Le curseur (en fait, la case active) effectue des va-etvient entre ces deux zones.
Dans la zone du bas (dite de 'Sélection'), tous les caractères du jeu sont affichés et les commandes disponibles sont celles décrites précédemment dans la partie 'Exploration' : "Flèches Return H S N R M". Dès que la touche RETURN a été pressée, le curseur passe dans la zone du haut.
Dans la zone du haut (dite d'Assemblage'), le caractère qui vient d'être sélectionné s'affiche en haut à gauche de l'écran. Le mini-menu du milieu d'écran s'est adapté et devient : "Flèches S N R et Return pour finir". Notez que les commandes H et M ne sont plus disponibles et que la dernière commande devant être utilisée doit obligatoirement être RETURN.
Les 4 flèches permettent de choisir l'endroit où le caractère choisi sera affiché. Pour reconstituer un sprite, il faut évidemment connaître les caractères qui le composent et leur place respective. Sinon, il faudra tâtonner. Notez que pour déplacer un caractère, il faut d'abord l'effacer (sélectionnez le caractère espace) puis le réafficher à sa nouvelle place (désolé pour cette

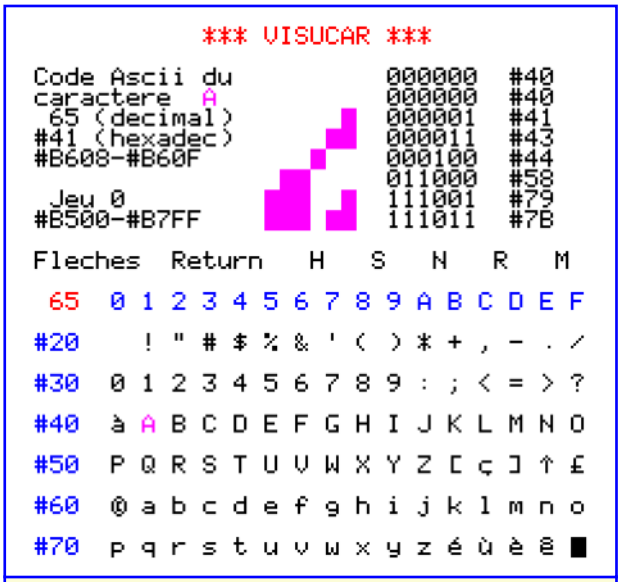

Fig. 15 : Exploration du jeu 0 redéfini de Rush Hour vu avec la Cde 'N'.

Fig. 16 : Exploration du jeu 0 redéfini de Rush Hour vu avec la Cde 'R'.

limitation).
Les commandes ‘S’ (Sauve l’écran), ‘N’ (met en service le jeu Natif) et ‘R’ (Restaure le jeu redéfini) fonctionnent comme décrit précédemment.
Un appui sur RETURN affiche le caractère à l’endroit choisi et le curseur retourne dans la zone de sélection. RETURN est donc la seule sortie de la zone d’assemblage. Le programme reboucle sans fin entre les deux zones et la seule sortie réelle est ‘M’ puis ‘Q’.

Exemple emblématique : Rush Hour de Fabrice F.

Le programme de Fabrice n’est pas protégé et on peut donc accéder facilement aux DATA qui sont d’ailleurs bien commentés. L’affichage statique des véhicules à l’écran laisse tout le temps nécessaire pour les observer et en apprécier le design. VisuCar n’est donc pas vraiment utile pour explorer Rush Hour et en tous cas pas indispensable.
Cependant, Rush Hour est un cas intéressant à plus

```
999 ' DISPLAY SPORTCAR
1000 PRINT@X,Y+1;COL$;"@ABCDE ";
1001 PRINT@X,Y+2;COL$;"FGHIJKL";
1002 PRINT@X,Y+3;COL$;"MNOPQRS";
1003 RETURN
1009 ' DISPLAY COMMON CAR
1010 PRINT@X,Y+1;COL$;"TUBVCDW";
1011 PRINT@X,Y+2;COL$;"XYZ[\]^";
1012 PRINT@X,Y+3;COL$;"MNOPQR_";
1013 RETURN
1019 ' DISPLAY HORIZ. TRUCK
1020 PRINT@X,Y+0;COL$;"#!!!!!!!$  ";
1021 PRINT@X,Y+1;COL$;"VVVVVVVV&'(";
1022 PRINT@X,Y+2;COL$;"!!!!!!!!)*+";
1023 PRINT@X,Y+3;COL$;",N./0123456";
1024 RETURN
1099 ' DISPLAY VERT. TRUCK (LONG)
1100 PRINT@X,Y;"    ";
1101 PRINT@X,Y+1;COL$;"a!b";
1102 PRINT@X,Y+2;COL$;"c!d";
1103 PRINT@X,Y+3;COL$;"e!f";
1104 PRINT@X,Y+4;COL$;"ghi";
1105 PRINT@X,Y+5;COL$;"j!k";
1106 PRINT@X,Y+6;COL$;"lmn";
1107 PRINT@X,Y+7;COL$;"o!p";
1108 PRINT@X,Y+8;COL$;"qrs";
1109 PRINT@X,Y+9;COL$;"tuv";
1110 PRINT@X,Y+10;COL$;"wxy";
1111 PRINT@X,Y+11;COL$;"z{|";
1112 RETURN
1119 ' DISPLAY VERT. TRUCK
1120 PRINT@X,Y;"    ";
1121 PRINT@X,Y+1;COL$;"j!k";
1122 PRINT@X,Y+2;COL$;"lmn";
1123 PRINT@X,Y+3;COL$;"o!p";
1124 PRINT@X,Y+4;COL$;"qrs";
1125 PRINT@X,Y+5;COL$;"tuv";
1126 PRINT@X,Y+6;COL$;"wxy";
1127 PRINT@X,Y+7;COL$;"z{|";
1128 RETURN
```

Fig. 18 : Détail du listing de Rush Hour montrant l’assemblage des caractères pour former les véhicules

d’un titre et notamment en ce qui concerne la redéfinition des caractères. Pour dessiner ses 5 véhicules (sport car, common car, horizontal truck, vertical truck et long vertical truck), qui sont en fait des méga-sprites, Fabrice a eu besoin de redéfinir 95 caractères ! De plus il devait aussi garder beaucoup de caractères normaux du jeu 0, tels que les lettres (par exemple pour le scrolling des publicités sur certains véhicules) et les chiffres (par exemple pour l’affichage des numéros de tableaux et des codes).
Il ne semblait pas facile de caser tout ça sans entrer dans des complications de mise en œuvre. Mais Fabrice ne pouvait pas resté coincé pour si peu ! Voici sa solution :
1) Recopier les 64 premiers caractères du jeu 0 (de ‘Espace’ Ascii 32 à ‘£’ Ascii 95) dans le jeu 1 (dont la place occupée en Ram est réduite par rapport à celle du jeu 0). La figure 13, page précédente, montre le résultat de cette copie.
2) Puis redéfinir tous les caractères du jeu 0 (sauf l’espace). Les 96 caractères du jeu 0 n’étant plus utilisés pour l’affichage normal sont devenus disponibles.
Pour l’affichage (très minoritaire) des caractères usuels, il suffit de mettre un attribut semi-graphique (#09) par devant ! Autre avantage, l’affichage (quantitativement et dynamiquement majoritaire) des méga-sprites sera simplifié, puisque en mode Text, c’est le jeu 0 qui est utilisé par défaut.
Mais pour VisuCar cette situation est compliquée. La figure 14, page précédente montre à quoi ressemble le menu général après chargement du jeu 0 de Rush Hour ! Pas vraiment lisible, non ? Heureusement, VisuCar est équipé pour faire face à ce genre de situation :
1) D’une part, l’aller-retour entre jeu Natif et jeu Redéfini permet de lire sans difficulté le Menu, l’écran de sauvegarde et les écrans d’aide.
2) D’autre part, les commandes ‘N’ (mise en service du jeu Natif) et ‘R’ (Restauration du jeu Redéfini) rendent possible l’utilisation de cet utilitaire, même quand l’écran est brouillé par des redéfinitions trop nombreuses
Ainsi les figures 15 et 16, page précédente, illustrent ce qui se passe avec Rush Hour lors de l’exploration du caractère Ascii 65 (ex caractère A) du jeu 0.
La situation est encore plus critique lorsqu’on essaie d’assembler divers caractères du jeu 0 pour reconstituer les véhicules de Rush Hour. La figure 17, page suivante, illustre l’impossibilité de s’y retrouver dans cette marée de caractères redéfinis. Heureusement le listing de Rush Hour montre quels caractères sont utilisés pour chacun des 5 véhicules (figure 18,ci-contre). La commande ‘N’, qui bascule sur le jeu Natif, (figure 19, page suivante) permet de s’y retrouver et de placer patiemment les bons caractères au bon endroit

Fig. 17 : Tentative d'assemblage avec le jeu 0 redéfini de Rush Hour vu avec la Cde 'R'.

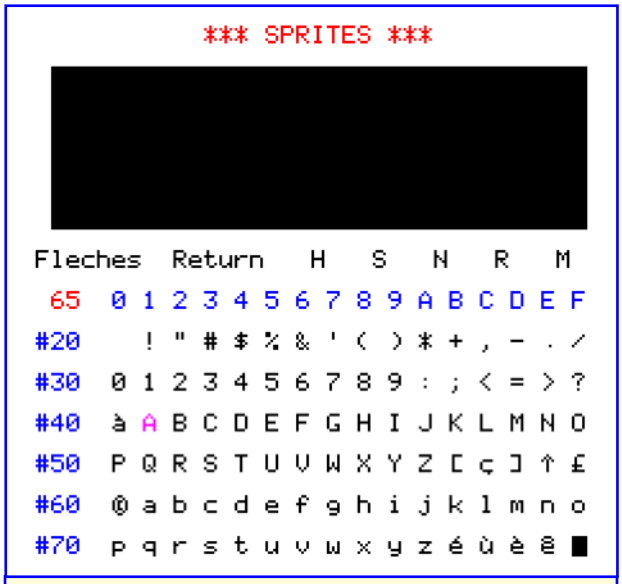

Fig. 19 : Comme la fig 17, mais après avoir pressé la Cde 'N'.

(figure 20, ci-dessous). Un appui sur 'R' (Restauration du jeu Redéfini) permet d'admirer le résultat (figure 21, ci-dessous).

Conclusion : Possibilités et limitations

VisuCar est un outil qui devrait vous permettre de visualiser tous les caractères redéfinis, qu'ils soient localisés dans le jeu 0 ou dans le jeu 1 et d'en connaître les octets de redéfinition, ainsi que leurs adresses. La présente version de cet utilitaire ne permet pas de générer des lignes de DATA pour réutiliser ces caractères dans un programme Basic. Mais cela ne pose pas vraiment de problème lorsqu'on sait où se trouvent ces infos dans la Ram. VisuCar ne permet pas non plus de modifier le dessin des caractères. Ce n'est pas un éditeur de caractères.

VisuCar permet par contre de reconstituer des sprites en combinant des caractères redéfinis. A la limite, il pourrait être utilisé comme un éditeur de sprites. En effet, il permet de juger de la validité d'une autre combinaison des caractères redéfinis. Gageons qu'il est bien possible de créer de nouveaux véhicules en réutilisant les 95 caractères redéfinis de Fabrice !

La limitation la plus sérieuse concerne les programmes atypiques (cas heureusement assez rares) : C'est le cas des programmes qui ne copient les octets de redéfinition en Ram qu'en fonction des besoins (contournement astucieux du nombre maximum possible de caractères redéfinis).

C'est aussi le cas des programmes qui n'utilisent pas les vecteurs normaux d'affichage. On ne sait pas où sont les caractères. Tout ce qu'on sait, c'est qu'ils existent, puisqu'on les voit à l'écran. Ceci est particulièrement gênant avec les programmes en langage machine. La seule issue est d'effectuer des recopies d'écran, de redessiner les caractères et d'en recalculer les octets de redéfinition. Un éditeur de caractère sera alors un outil précieux. La bonne vieille methode quoi !

En espérant que mon utilitaire vous sera… utile !

Fig. 20 : L'assemblage est fait avec les caractères Natifs non-redéfinis.

Fig. 21 : Voici le résultat obtenu après avoir pressé la touche 'R'.