ART-INFO MUS-INFO

NOTE DE L'EDITEUR

DANS CE NOUVEAU NUMÉRO, ON TROUVERA PLEIN DE CHOSES DIVERSES :

- MICHEL BRET, UN DES COLORIXIENS DU GAIV PRÉSENTE UN PROGRAMME GRAPHIQUE.

- JÉRÔME CHAILLOUX, AVEC MAD 8 PROUVE AUX MINICOMPUTÉREUX ADEPTES DU 8008 QU'ON PEUT SE DÉSASSEMBLER SANS SOMBRER DANS L'ÉGAREMENT.

- JEAN-MICHEL FAVRE, ÉTUDIANT AU DÉPARTEMENT D'INFORMATIQUE, A CONÇU UN PETIT INTERPRÈTE MUSICAL PÉDAGOGIQUE.

- QUANT À JEAN-FRANÇOIS DEGREMONT, NON CONTENT DE SUPERVISER CE NUMÉRO, IL NOUS FAIT PART DE LA NAISSANCE DE SON DERNIER REJETON, AUQUEL LES BONNES FÉES ONT DÉJÀ PRÉDIT UNE CARRIÈRE ARTISTIQUE SANS PRÉCÉDENT.

- ENFIN POUR LES AMATEURS D'UNHEIMLICH : IMPOSSIBLE CRISTAL, DE JEAN-ERIC SCHOETTL.

- UNE PAGE D'ERRATA ÉGALEMENT, CORRESPONDANT À L'ARTICLE DE MARC BATTIER DANS LE N°27.

- LA COUVERTURE EST DE DANIEL GOOSSENS, UN DES PLUS SAUVAGES LISPIENS DU DÉPARTEMENT, ZÉLATEUR DE L'INTELLIGENCE ET DE L'ARTIFICE, ET PAR AILLEURS COLLABORATEUR D'UNE REVUE SCIENTIFIQUE BIEN CONNUE : FLUIDE GLACIAL.

- QUANT AU GROUPE ART ET INFORMATIQUE DE VINCENNES, IL EST PLUS PROSPÈRE QUE JAMAIS : EXPOSITIONS À MARSEILLE (JUIN-JUILLET 77), À GLASGOW (SEPT-OCT 77), CONFÉRENCES À SAN DIEGO (OCT 77), AU CNRS (NOV 77), SHOWS À PARIS (CENTRE CULTUREL SUISSE, JANVIER 78), À SOCHAUX (FÉVRIER 78), À GRENOBLE (MARS 78).
LES PROJETS, EUX, SONT INNOMBRABLES : TOUT VA BIEN.

J.A.

ESSAI D'ANIMATION

MICHEL BRET

AVANT : Peintures, Mathématiques, Informatique

PENDANT : Art & Info (COLORIX, table traçante)

BIENTOT : "Le pinceau pensa et ça moulina"

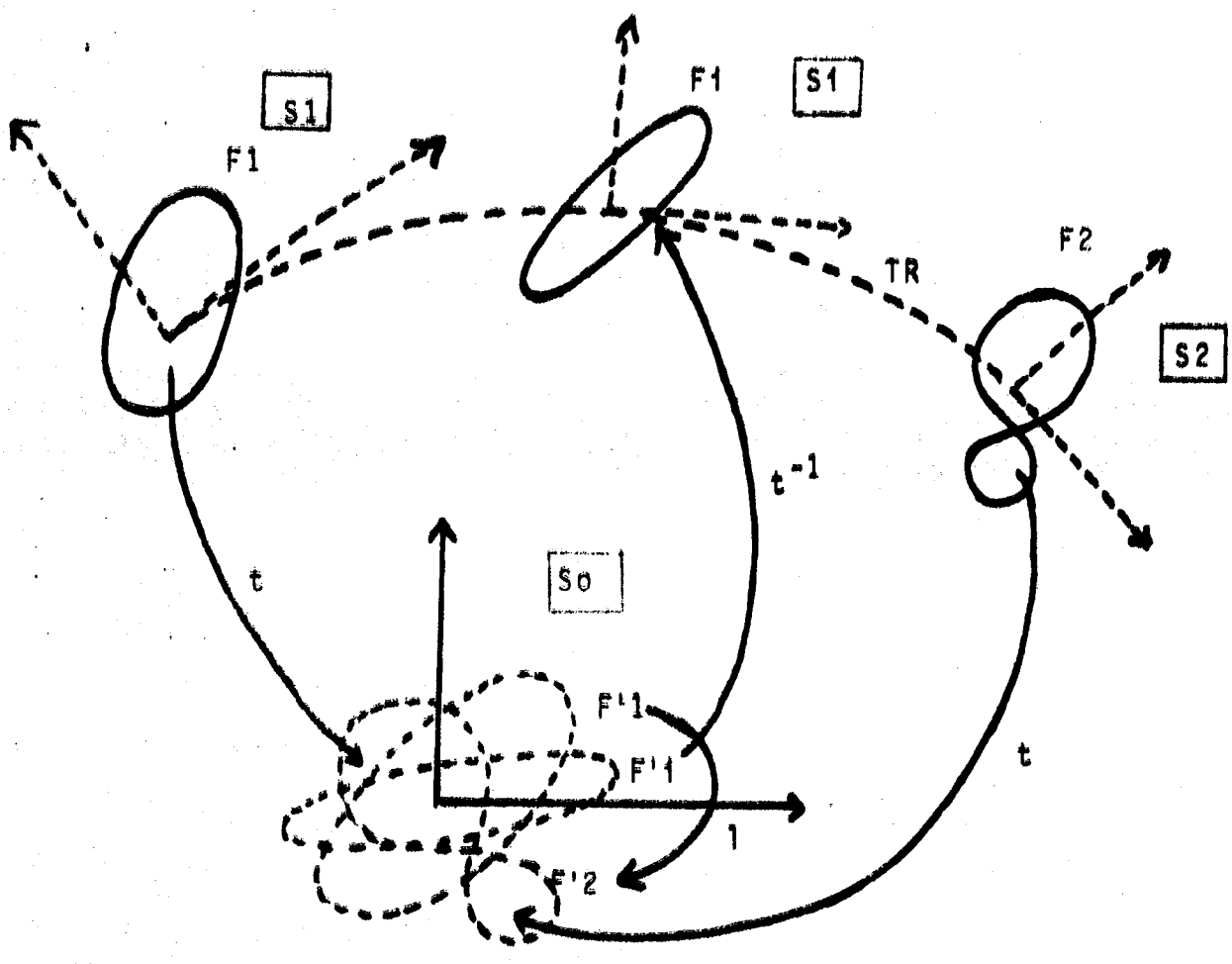
ARTINFO/MUSINFO #28

I - LE PROGRAMME ANY

A) PRINCIPE

On se donne une "forme" F1, une "forme" F2 et une "trajectoire" TR ; le programme prend une série de formes {Fi} telles que :

- ▲ la première soit F1
- ▲ la dernière soit F2
- ▲ leur ensemble est disposé "comme" TR



So : repère absolu
{Si} : repères locaux de la trajectoire TR
t : changement de repère
l : transformation linéaire F1 → Fi → F2

B) UNE PROCÉDURE QUI DESSINE :

Plutôt que de stocker beaucoup de choses, j'ai cherché une procédure reconstruisant une "forme" (dans cet essai : une ligne plane) à partir d'un petit nombre d'éléments (ici : quelques points remarquables et des paramètres précisant des caractéristiques locales de la ligne : tendue-creusée, simple-bouclée, etc.).

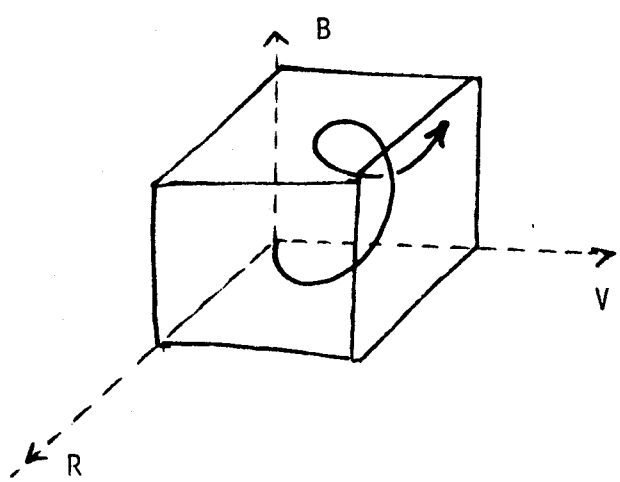
Pour cela une première procédure BIT détermine la forme de la ligne au voisinage de chaque point donné en fonction des positions relatives des points les plus proches et des paramètres donnés. Puis une seconde procédure (récursive) joint ces points en tenant compte des calculs précédents. (Voir listing commenté ci-après).

C) APPLICATIONS

- ★ Avec une table de saisie et un jeu d'entraînement pour le choix des paramètres on peut faire du figuratif, de l'écriture cursive, etc.
- ★ En 3 dimensions (LSE 3D) on peut définir des lignes de l'espace et, par animation d'une ligne à l'autre, engendrer des surfaces
- ★ Quand quelquechose dépend de paramètres (c'est pas si rare que ça ...) on peut faire "bouger" de quelque chose en considérant une telle ligne dans l'espace des paramètres.

  Ainsi on peut définir des transformations de couleurs dans l'espace (R,V,B).

  L'intérêt est de permettre de définir rapidement et d'une façon intuitive une sorte de trajectoire de ce que l'on veut faire évoluer.



☆☆ PROGRAMME ☆☆

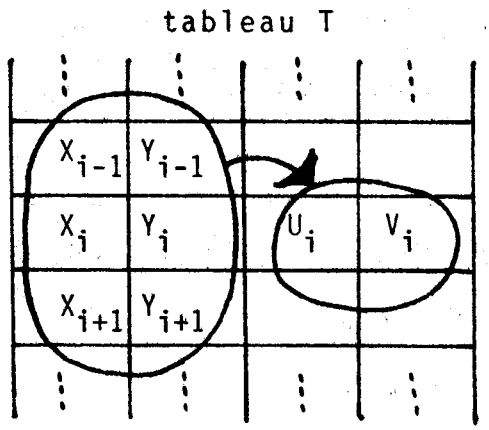
```
150 PRØCEDURE &BIT(T,NL) LØCAL DL
151 UA←T[NL,1]-T[NL-1,1];VA←T[NL,2]-T[NL-1,2];DL←RAC(UA*UA+VA*VA)
152 SI DL=0 ALØRS &ER(6,1);T[NL,3]←UA/DL;T[NL,4]←VA/DL;UA←T[2,1]-T[1,1]
153 VA←T[2,2]-T[1,2];DL←RAC(UA*UA+VA*VA);SI DL=0 ALØRS &ER(4,-1)
154 T[1,3]←UA/DL;T[1,4]←VA/DL;FAIRE 157 FØUR I←2 JUSQUA NL-1
155 UB←T[I+1,1]-T[I,1];VB←T[I+1,2]-T[I,2];UA←UA+UB;VA←VA+VB
156 DL←RAC(UA*UA+VA*VA);SI DL=0 ALØRS &ER(4,-1)
157 T[I,3]←UA/DL;T[I,4]←VA/DL;UA←UB;VA←VB
158 RETØUR
```

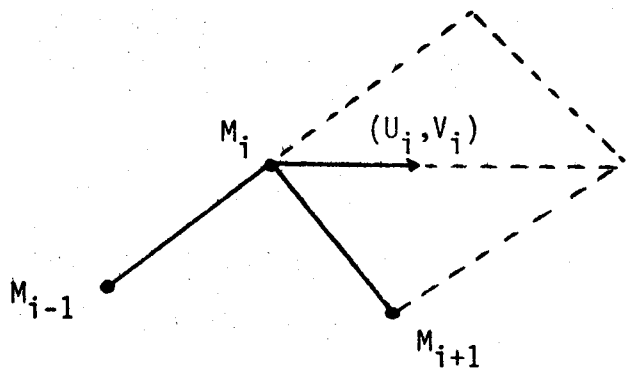
```
183 PRØCEDURE &BP(X1,Y1,U1,V1,X2,Y2,U2,V2,N,LK) LØCAL Y2,X2,Y1,X1
184 L←RAC((X2-X1)*(X2-X1)+(Y2-Y1)*(Y2-Y1))*LK
185 X3←X1+L*U1;Y3←Y1+L*V1;X4←X2-L*U2;Y4←Y2-L*V2
186 X5←(X3+X4)/2;Y5←(Y3+Y4)/2;TABLEAU TB[3,2];TB[1,1]←X1
187 TB[1,2]←Y1;TB[2,1]←X3;TB[2,2]←Y3;TB[3,1]←X5;TB[3,2]←Y5
188 &BP2(TB,N);TB[1,1]←X5;TB[1,2]←Y5;TB[2,1]←X4;TB[2,2]←Y4
189 TB[3,1]←X2;TB[3,2]←Y2;&BP2(TB,N);RETØUR
190 PRØCEDURE &BP2(TB,N) LØCAL N,TB,TP,E,F;TABLEAU TP[3,2]
191 SI N≠0 ALØRS ALLER EN 193;&S(TB[1,1],TB[1,2],TB[2,1],TB[2,2])
192 &S(TB[2,1],TB[2,2],TB[3,1],TB[3,2]);RETØUR
193 TP[1,1]←(TB[1,1]+TB[2,1])/2;TP[1,2]←(TB[1,2]+TB[2,2])/2
194 TP[2,1]←(TB[2,1]+TB[3,1])/2;TP[2,2]←(TB[2,2]+TB[3,2])/2
195 TP[3,1]←(TP[1,1]+TP[2,1])/2;TP[3,2]←(TP[1,2]+TP[2,2])/2
196 E←TB[3,1];F←TB[3,2];TB[2,1]←TP[1,1];TB[2,2]←TP[1,2]
197 TB[3,1]←TP[3,1];TB[3,2]←TP[3,2];&BP2(TB,N-1)
198 TB[1,1]←TP[3,1];TB[1,2]←TP[3,2];TB[2,1]←TP[2,1]
199 TB[2,2]←TP[2,2];TB[3,1]←E;TB[3,2]←F;&BP2(TB,N-1);RETØUR
```

```
200 PRØCEDURE &S(X1,Y1,X2,Y2) LØCAL Y2,X2,Y1,X1;SI CL=0 ALØRS RETØUR
201 AFFICHER[U]&SEG(X1,Y1,0);AFFICHER[U]&SEG(X2,Y2,1);RETØUR
```

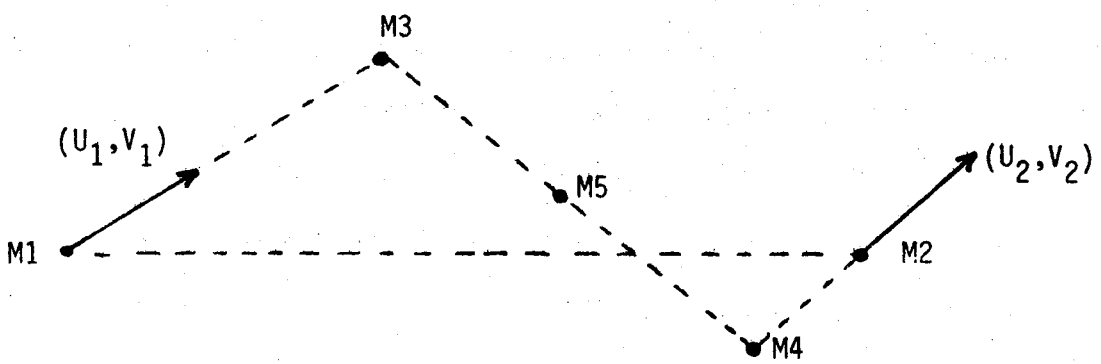
★☆ COMMENTAIRES ☆★

150 à 158 : *la procédure BIT* détermine les "tangentes"





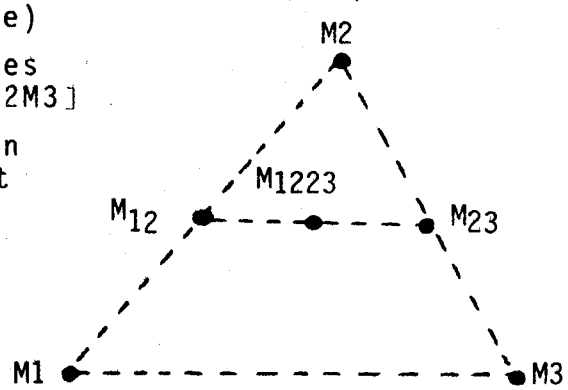
183 à 189 : *la procédure BP* détermine, pour chaque couple de points flanqués de leurs tangentes, 2 groupes de 3 points (permet de traiter les inflexions) et appelle la procédure BP2 pour chacun de ces triplets.



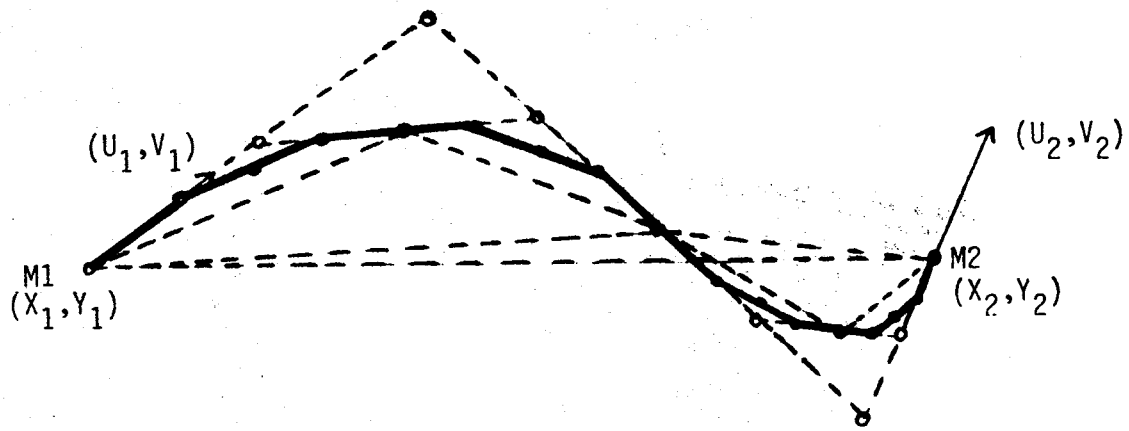
190 à 199 : *la procédure BP2* récursive (travaille sur des triplets de points M1,M2,M3 ; N est la profondeur de la récurrence)

Si N=0 : affichage des segments [M1M2]et [M2M3]

Sinon : détermination des points M_{12},M_{23} et M_{1223} puis appels de BP2 récursivement pour :
($M1$,M_{12},M_{1223},N-1)
(M_{1223},M_{23},M3,N-1)



Exemple : effet de BP(X1,Y1,U1,V1,X2,Y2,U2,V2,2,0.5)



II) Le projet du peintre évolue au fur et à mesure de sa réalisation, pour aller vite : le modèle c'est l'oeuvre. Je rêve d'un "pinceau intelligent", prévenant mes intentions et en donnant son interprétation personnelle, laquelle, modifiée par mes soins, induirait son comportement futur.

Le programme d'Intelligence Artificielle qui ferait ça serait donc capable de reconnaître les formes que j'ébauche en se basant sur les résultats du dialogue (apprentissage) que constitue la série des [interprétations (par le programme) — modifications (par l'utilisateur)].

M A D 8

Un Désassembleur pour 8008

Jérôme CHAILLOUX

ARTINFO/MUSINFO #28

MAD8 un desassembleur pour 8008

****
M A D 8
****

Jerome CHAILLOUX
Fevrier 1977

MAD8 est un desassembleur de rubans perfores hexadecimaux issus du micro-processeur 8008. Il permet d'obtenir des listages "en clair" de vos programmes a partir d'un ruban perfore. MAD8 fonctionne sur le T1600.

1.0 Les rubans hexadecimaux.

Les rubans hexadecimaux images-memoire sont produits au moyen des commandes W, E et N du moniteur 8008. Ces rubans ne contiennent que des caracteres imprimables et peuvent donc etre relus sur une TTY non connectee (TTY en mode local).

1.1 La commande N (null command)

syntaxe : .N

perfore une avance bande de 60 caracteres nulls (code 00).

1.2 la commande W (write command)

syntaxe : .W adresse de debut , adresse de fin

perfore (dans le format decrit ci-dessous) l'image le la zone memoire commencant et se terminant aux adresses specifiees dans la commande. On peut emettre plusieurs commandes W a la suite pour obtenir, sur le meme ruban physique, les images de zones memoires non-contiguës.

1.3 la commande E (end-of-file commande)

syntaxe : .E adresse de lancement

perfore un bloc "fin de ruban" (voir son format ci-dessous) qui contient l'adresse de lancement du programme. Cette commande perfore en outre une avance bande.

1.4 enchainement des commandes de perforation.

```
.N            }--perforation d'une 1ere avance bande.
.Whhhh,hhhh   |
  ...         |
  ...         |--perforation des differentes zones memoires
  ...         |
.Whhhh,hhhh   |
.Exxxx        }--perforation de l'adresse de lancement
                 et d'une fin de bande.
```

1.5 format des rubans hexadecimaux

Ils sont formes d'enregistrements qui possedent tous la meme structure. Chaque octet est represente par deux digits hexadecimaux (hh).

```
: hh hhhh 00 hh .. .. hh hh
```

octet de cheksum, calcule de telle maniere que la somme de tous les octets de l'enregistrement (y compris le nombre de donnees, l'adresse de l'octet de cheksum lui meme) soit egale a 0.

n octets de donnees a charger en memoire a partir de l'adresse specifiee. Le nombre n est le 1er octet de l'enregistrement.

type de l'enregistrement (toujours 00).

adresse memoire (sur 2 octets) ou il faut charger les donnees.

nombre d'octets de donnees a charger en memoire.

caractere ":" destine a marquer le debut de l'enregistrement.

La fin logique du ruban est detectee par un enregistrement dont le nombre de donnees a charger est egal a zero.

```
: 00 hhhh 00 hh
                 octet de cheksum
              toujours 00.
        adresse de lancement du programme.
    marque de fin de ruban logique.
  marque de debut d'enregistrement.
```

2.0 Principe du desassembleur.

Les rubans hexadecimaux sont tres utiles pour garder des programmes sous un format tres reduit, en revanche, ils sont peu commodes pour documenter, modifier ou meme comprendre ces memes programmes. MAD8 va editer chaque octet code du ruban perfore sous les differents formats :
- mnemonique instruction (INTELGREU),
- valeur hexadecimale
- caractere ASCII.

Les instructions du 8008 etant de longeur variable, les resulats ainsi obtenus sont tres mediocres pour le listage des instructions. Par exemple le desassemblage des 3 octets correspondant a l'instruction JMP 1000 produit l'edition de :

```
JMP  44  D
HLT  00  ?
INC  10  ?
```

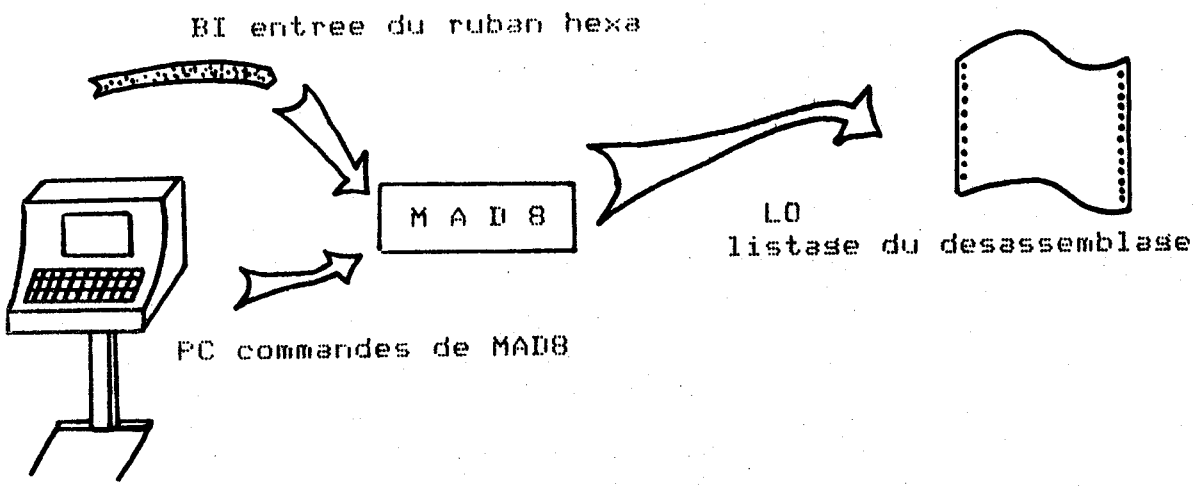
MAD8 va s'efforcer de distinguer dans l'image memoire les zones contenant des instructions, des zones contenant des donnees. Pour ce faire, MAD8 va simuler l'execution du programme a partir de son adresse de lancement, et marquer les octets qui sont consideres comme des instructions. Cette simulation consiste a interpreter les instructions qui provoque des ruptures de sequence i.e. les instructions de type HLT, JMP, CAL, RET et RST, conditionnelles ou inconditionnelles. Ce marquage ne peut s'effectuer qu'apres avoir charge en entier le programme a desassembler.

Une fois le marquage effectue, MAD8 edite l'image memoire sous les formats instructions ou donnees.

Ce principe a toutefois des limitations. Les branchements indirects et/ou indexes ne peuvent pas etre interpretes. Les donnees ne doivent pas etre melangees au programme. En particulier les appels de sous-programmes dont les arguments sont places juste en dessous de l'appel effectif risque de placer MAD8 dans un etat de confusion deplorable.

3.0 utilisation du desassembleur.

MAD8 est un utilitaire standard du T1600 qui utilise les differentes FUs :



3.1 activation de MAD8

*CALL MAD8 appel du programme MAD8. Si BOS/D imprime le messase d'erreur ERB 06, le programme n'est plus sur le disque; il ne vous reste plus qu'a desassembler votre programme a a la main ou a remettre MAD8 sur le disque.

*BI TR affectation du lecteur de ruban

*PC TK affectation du clavier TTY pour entrer les commandes de MAD8.

*LO LP affectation de l'imprimante pour le listase final.

3.2 commandes MAD8

*IMAD initialise MAD8 et lit le 1er ruban sur l'unite BI. Cette commande est oblisatoire et ne doit etre emise qu'une seule fois sous peine de perdre l'imase memoire qui avait ete cree.

ou

MAD8 ne peut simuler que les 8 iers k du 8008. Si vous voulez desassembler des programmes en REPROM (i.e des programmes dont l'adresse est plus grande que 2000 hexa), vous pouvez specifier dans la commande la 1ere adresse a simuler.

*IMAD,hhhh Cette deuxieme forme n'est dons pas a utiliser pour des posrammes en RAM.

*CMAD permet de lire d'autres rubans sans reinitialiser le systeme, si votre votre programme se trouve sur plusieurs rubans perfores. Cette commande peut etre emise plusieurs fois.

*MMAD marque l'image memoire. On suppose que la premiere adresse du programme (son adresse de lancement) a ete lue sur le bloc fin de ruban du dernier ruban lu.

*MMAD,hhhh marque l'image memoire a partir de l'adresse specifiee dans la commande. Cette commande peut etre emise plusieurs fois en particulier pour specifier les differentes adresses se trouvant dans une table de branchements indirects indexes.

*LMAD edite sur l'unite LO le resultat du desassemblage. Cette commande peut etre emise plusieurs fois pour obtenir plusieurs copies du desassemblage.

*EOJ fin d'execution de MAD8.

3.3 utilisation du disque

Le lecteur de ruban de la TTY est tres lent. Il est parfois avantageux de creer un fichier sur disque contenant l'image du ruban hexadecimal a desassembler ce qui evite de recharger le ruban apres chaque erreur.
Pour copier le ruban perfore sur disque, il faut utiliser l'utilitaire standard du T1600 : le FUP6.

*CALL FUP6 appel de l'utilitaire FUP6.

*INPUT,TR,PTAP definition du support d'entree

```
*OUTPUT,nom-:I definition du fichier de sortie.  L'extension
             :I  est  reservee  pour  les  fichiers  hexa  de
             l'Intel.

*TRANSF      effectue le transfert

*EOJ         fin du travail
```

4.0 Exemples d'utilisation de MAD8

desassemblage direct d'un ruban perfore :

```
*CALL MAD8        appel de MAD8.
*BI TR            selection du lecteur en entree.
*LO LP            selection de l'imprimante.
*IMAD             initialisation et lecture
 --- lecture du ruban ---
*MMAD             marquage des instructions.
*LMAD             listage de l'image memoire.
 --- impression du resultat ---
*EOJ              voila le travail
```

Exemple du desassemblage complet du moniteur 8008.

```
*EOJ

*CALL FUP6        creation d'un fichier disque
*INPUT,TR,PTAP    contenant le ruban hexa du moniteur.
*OUTPUT,MONIT-:I,D2
*TRANSF
 --- lecture du ruban ---
*EOJ

*CALL MAD8
*BI MONIT-:I,D2
*PC TK
*LO LP
*IMAD,2000
*MMAD
*MMAD,38A3        marquage de tous les modules
*MMAD,39DE        du moniteur.
*MMAD,3967
*MMAD,39A9
*MMAD,39D4
*MMAD,3A00
*MMAD,3A13
*MMAD,3A1A
*MMAD,3C43
```

```
*MMAD,3A45
*MMAD,3A5F
*MMAD,3A8B
*MMAD,3A91
*MMAD,3AF6
*MMAD,3B68
*MMAD,3BB7
*MMAD,3BD5
*LMAD
 --- impression du desassemblage ----
*EOJ
```

5.0 Exemple de listage produit par MAD8.

```
---
---
0040  JMP  0703
0043  LLI  32 2
0045  LHI  00
0047  LCI  01
0049  JMP  3D80
004C  LLI  30 0
004E  LHI  00
0050  LAM
0051  INL
0052  LBM
0053  LHA
0054  LLB
0055  RET
0056  CAL  004C
0059  LME
005A  CAL  3DEB
005D  LAH
005E  LBL
005F  LLI  30 0
0061  LHI  00
0063  LMA
0064  INL
0065  LMB
0066  RET
0067        IN   41 A
0068        CFC  42 B
0069        IN   43 C
006A        JMP  44 D
006B        IN   45 E
006C        JFZ  48 H
006D        ---  4C L
006E        IN   4D M
006F        IN   49 I
0070  CAL  3F44
0073  LBI  08
0075  LLI  6F
0077  LHI  00
0079  CPM
007A  JFZ  007F
007D  LAB
007E  RET
007F  DCL
0080  DCB
0081  JFS  0079
0084  JMP  3C43
0087  LLI  33 3
0089  LHI  00
008B  LME
008C  RET
008D  CPI  46 F
008F  RTZ
0090  CPI  54 T
0092  JFZ  3C43
0095  LAI  20
0097  LLI  33 3
0099  LHI  00
009B  ADM
009C  LMA
009D  RET
009E  CAL  00A7
00A1  LLI  33 3
00A3  LHI  00
00A5  LEM
00A6  RET
00A7  CPI  43 C
00A9  RTZ
00AA  CPI  5A Z
00AC  LBI  08
00AE  JTZ  00BF
00B1  LBI  10
00B3  CPI  53 S
00B5  JTZ  00BF
00B8  LBI  18
00BA  CPI  50 P
00BC  JFZ  3C43
00BF  LAB
00C0  JMP  0097
00C3  CPI  08
00C5  JTZ  055E
```

```
00C8 | ADE
00C9 | ADI 7C
00CB | JMP 0553
---
0500 | CAL 3CC7
0503 | LBI 3A :
0505 | CAL 3809
0508 | CAL 0043
050B | LLI 30 0
050D | LMD
050E | INL
050F | LME
0510 | CAL 3CC7
0513 | CAL 004C
0516 | CAL 3DFB
0519 | CAL 3F44
051C | CPI 24 $
051E | JTZ 3844
0521 | CPI 42 B
0523 | JFZ 0538
0526 | CAL 3F44
0529 | CAL 3F44
052C | CAL 3C4B
052F | CAL 0043
0532 | CAL 0056
0535 | JMP 0510
  :  |   :
```

c'est probablement des données

c'est probablement du programme

c'est sûrement l'adresse d'implantation

DEFINITION ET IMPLEMENTATION D'UN LANGAGE PEDAGOGIQUE ADAPTE A LA PROGRAMMATION MUSICALE

JEAN-MICHEL FAVRE

ARTINFO/MUSINFO #28

Le matériel utilisé est l'Intel 8008 du Département d'Informatique (microprocesseur à mots de 8 bits) avec en périphérie un DAC à 8 sorties et un synthétiseur qui confèrent à l'ensemble sa vocation musicale.

Le synthétiseur vu par l'utilisateur de l'interprète se réduit aux 3 seuls éléments qui peuvent actuellement être commandés par programme :

- l'oscillateur 2 (VCO2)
- l'amplificateur de sortie 1 (VCA1)
- le filtre.

Seules 3 sorties sur 8 du DAC sont utilisées, celles qui permettent de contrôler les éléments précédemment cités du synthétiseur :

- sortie 1 : contrôle du VCA1
- sortie 10 : contrôle du VCO2
- sortie 20 : contrôle du filtre.

L'interprète étant chargé en mémoire, il doit être lancé par la commande :G164E↵ . L'interprète répond par ####↵ et est alors prêt à recevoir des commandes à partir du clavier de la TTY.

Un programme syntaxiquement correct est composé de 5 instructions élémentaires dans n'importe quel ordre, mais obligatoirement suivies de l'instruction FIN. Soit I_1, I_2, I_3, I_4, I_5, les instructions élémentaires et I_6 l'instruction FIN.

I_1 : <u>Instruction permettant de générer un son</u>

$I_1 \Rightarrow$ {1,2,3,4} {DO,RE,...,SI} {♯,♭} {01,...,99,;} {1,...,9,;} {.,;}

1 2 3 4 5 6

Cette instruction est composée de 6 champs.

Le premier champ permet de sélectionner une octave parmi quatre, le deuxième une fréquence particulière dans cette octave. La note ainsi choisie peut être altérée ou non grâce au troisième champ. Si un dieze est frappé au clavier la fréquence précédemment choisie est augmentée d'un ½ton,si c'est un blanc elle reste inchangée. Le quatrième champ permet d'affecter une longueur à la note. L'unité de longueur étant environ de 250

millisecondes. L'utilisateur peut donc choisir explicitement entre 99 longueurs de notes différentes ou frappe ";" pour indiquer plus brièvement une longueur de 1 unité. Le cinquième champ permet de contrôler l'amplitude du signal. Le choix est donné entre 9 amplitudes différentes, plus une option par défaut : le ";" qui correspond à l'amplitude maximum.

Le choix est également donné entre deux types d'enveloppes . Le signal peut présenter une chute assez brusque d'amplitude dès que sa durée est écoulée et on l'indique grâce à un ";" dans le champ 6. Si l'utilisateur préfère que le signal ait une amplitude constante jusqu'à l'apparition de la note suivante, il faut l'indiquer par un "." dans le champ 6.

I_2 , I_3 : <u>Instructions de définition d'étiquette et d'itération</u>

$I_2 \Rightarrow$ {E} {1,...,9}

Défini un numéro d'étiquette. Dans l'ensemble d'un programme on a donc droit à 9 étiquettes au plus.

$I_3 \Rightarrow$ {I} {1,...,9} {01,...,99}
1 2 3

Le premier champ notifie que c'est une instruction d'itération.
Le deuxième précise un numéro d'étiquette obligatoirement définie par une instruction du type I_2 .
Le troisième indique le nombre d'itérations.

Exemple :

```
E   6
1DO#;;;
   :
séquence
d'instructions
   :
I 603
```

Dans ce cas l'effet de l'instruction :I 603 , sera de faire exécuter 3 fois la séquence d'instructions entre l'étiquette 6 et elle-même.

I_4 : Instruction de temporisation

$I_4 \Rightarrow$ {T} {01,...,99}

Si la note précédant une instruction de temporisation avait un ";" dans le champ 6, ou un ".", on peut programmer un silence ou une prolongation de la note de 1 à 99 unités de temps. Celle-ci peut être fixée en modifiant la mémoire à partir du pupitre.

I_5 : Contrôle de la fréquence du filtre

$I_5 \Rightarrow$ {W} {-9,...,-1,1,...,9}

Cette instruction permet de modifier la position de la bande passante du filtre dans le spectre par incrément de 1 pour des valeurs allant de -9 à +9. La tension initiale de contrôle est fixée à une valeur médiane : 7F. On peut en choisir une autre en modifiant la mémoire FFE.

I_6 : Instruction de fin

$I_6 \Rightarrow$ {F}

Elle termine puis lance l'exécution d'un programme. Quand l'interprète exécute cette instruction il rend la main au moniteur.

Les instructions suivantes sont en cours d'implémentation.

I_7 : Instruction de sélection

$I_7 \Rightarrow$ {S} {1,2}

Les mélodies programmées après cette instruction sortent sur le VCO. Sélectionner jusqu'à la prochaine instruction de sélection.

I_8 : Appel d'un sous-programme, retour d'un sous-programme

$I_8 \Rightarrow$ {CAL} {n}

n est une étiquette.

Cette instruction effectue un branchement à l'instruction suivant la définition de l'étiquette n.

$I_9 \Rightarrow$ {RET}

effectue un branchement à l'instruction suivant la derniere instruction CAL effectuée.

I_{10} : Instruction de saut

I_{10} ⇒ {J} {n}

n étant une étiquette.

Effectue un saut inconditionnel à l'instruction suivant la définition de l'étiquette n.

L'interprète détecte quelques erreurs de syntaxe. Il les signale en faisant un HALT qui empêche l'entrée de nouvelles instructions. On peut repartir en appuyant sur la touche RESET du pupitre et continuer le programme à partir de l'instruction ayant provoqué l'arrêt de la machine.

Un programme peut comporter au maximum 256 instructions dans lesquelles il ne faut pas compter les instructions de définition d'étiquettes qui sont plutôt des directives à l'interprète.

Sauvegarde d'un programme

On peut sauvegarder un programme après son exécution.

Si le programme a n instructions, il faut sauver sur ruban les mémoires :

1100 → 1100 + n-1
1200 → 1200 + n-1
1300 → 1300 + n-1
1700 → 1700 + n-1
1800 → 1800 + n-1
1900 → 1900 + n-1

Accord du synthétiseur

Il suffit d'écrire un programme qui sort deux notes séparées par un intervalle d'une octave et agir conjointement sur le VCO2 et le gain en sortie du convertisseur jusqu'à obtenir le son et l'intervalle désiré.

Réglage du filtre

L'effet des incréments de tension sortis par le calculateur dépend des réglages initiaux, du gain en sortie du convertisseur, de la position des boutons "RESPONSE" et "FREQUENCY" situés sur le tableau du synthétiseur.
Le bouton "LEVEL" du canal utilisé doit être à la position 0 et le potentiomètre du convertisseur 1 au maximum.

PROGRAMME

```
0900  LLI 0E
0902  RET
0903  LLI 1A
0905  RET
0906  LLI 2D -
0908  RET
0909  LLI 39 9
090B  RET
090C  LLI 45 E
090E  RET
090F  LLI 51 Q
0911  RET
0912  CPI 01
0914  JTZ 0900
0917  CPI 02
0919  JTZ 0903
091C  CPI 03
091E  JTZ 0906
0921  CPI 04
0923  JTZ 0909
0926  CPI 05
0928  JTZ 090C
092B  CPI 06
092D  JTZ 090F
0930  INB
0931  JFZ 0930
0934  LAC
0935  ADI 01
0937  LCA
0938  CPE
0939  JFZ 1673
093C  JMP 15D2
093F
0940  LHI 18
0942  LLI 00
0944  LMI E8
0946  INL
0947  JFZ 0944
094A  LHI 19
094C  LLI 00
094E  LMI 00
0950  INL
0951  JFZ 094E
0954  CAL 3F44
0957  JMP 0900
095A  JMP 0996
095D
095E
095F
0960
0961
0962
0963
0964
0965  SUI 01
0967  JTZ 15EC
096A  OUT 0B
096B  LCA
096C  LAI 01
096E  OUT 09
096F  LAI 00
0971  OUT 09
0972  LAC
0973  JMP 0965
0976  CAL 3C4B
0979  CAL 3F44
097C  NDI 7F
097E  LHI 14
0980  LLI 00
0982  LLM
0983  CPI 3B ;
0985  JFZ 098E
0988  CAL 3CC7
098B  JMP 0954
098E  DCL
098F  LHI 19
0991  LMI 01
0993  JMP 0988
0996  LHI 19
0998  LAM
0999  LHI 0F
099B  LBL
099C  LLI FF
099E  LMA
099F  LLB
09A0  CPI 00
09A2  LHI 18
09A4  LAM
09A5  JTZ 0965
09A8  JMP 15EC
09AB  LHI 0F
09AD  LBL
09AE  LLI FF
```

```
09B0  LAM
09B1  CPI 00
09B3  LLB
09B4  JTZ 166E
09B7  LCI F6
09B9  LEI 00
09BB  INE
09BC  JFZ 09BB
09BF  INC
09C0  JFZ 09B9
09C3  LHI 18
09C5  LAM
09C6  OUT 0B
09C7  LAI 01
09C9  OUT 09
09CA  LAI 00
09CC  OUT 09
09CD  JMP 15D2
09D0  CPI 57 W
09D2  JFZ 1444
09D5  LHI 14
09D7  LLI 00
09D9  LLM
09DA  LHI 11
09DC  LMI 05
09DE  LEL
09DF  CAL 3C4B
09E2  CAL 3F44
09E5  LLE
09E6  CPI 2D -
09E8  JTZ 0A70
09EB  NDI 0F
09ED  LHI 13
09EF  LMA
09F0  LHI 14
09F2  LLI 00
09F4  LBM
09F5  INB
09F6  LMB
09F7  CAL 3CC7
09FA  JMP 0954
09FD
09FE
09FF
0A00
0A01
0A02
0A03
0A04
0A05
0A06  CPI 03
0A08  JTZ 160D
0A0B  CPI 05
0A0D  JTZ 0A1B
0A10  JMP 0A54
0A13  NDI 0F
0A15  LBA
0A16  SUB
0A17  SUB
0A18  JMP 09ED
0A1B  LEL
0A1C  LHI 13
0A1E  LAM
0A1F  LHI 0F
0A21  LLI FE
0A23  LBM
0A24  ADB
0A25  LMA
0A26  LLE
0A27  OUT 0B
0A28  LAI 20
0A2A  OUT 09
0A2B  LAI 00
0A2D  OUT 09
0A2E  INL
0A2F  JMP 15BF
0A32
0A33
0A34
0A35
0A36
0A37
0A38
0A39
0A3A
0A3B
0A3C
0A3D
0A3E
0A3F  LHI 00
0A41  LLI 00
0A43  LMI 44 D
0A45  INL
0A46  LMI 39 9
0A48  INL
0A49  LMI 0A
0A4B  LHI 0F
0A4D  LLI FE
0A4F  LMI 7F
0A51  JMP 1420
0A54  LHI 00
0A56  LLI 01
0A58  LMI 00
0A5A  LLI 02
0A5C  LMI 38 8
0A5E  JMP 0000
0A61  LBM
0A62  LHI 10
0A64  LAI 26 &
0A66  ADB
0A67  LLA
0A68  LBM
0A69  INB
0A6A  LAB
0A6B  CPI 0C
0A6D  JFZ 163B
```

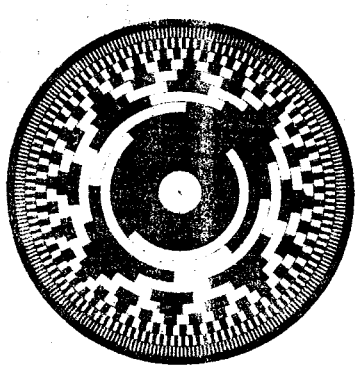
```
0A70  LEL
0A71  CAL 3F44
0A74  LLE
0A75  JMP 0A13
0A78
0A79
```

```
1420  LLI 5D I
1422  LHI 10
1424  LMI 00
1426  CAL 3DEB
1429  LAH
142A  CPI 14
142C  JFZ 1424
142F  LAL
1430  CPI 1F
1432  JFZ 1424
1435  LHI 17
1437  LLI 00
1439  LMI FF
143B  INL
143C  JFZ 1439
143F  JMP 0940
1442
1443
1444  NDI 7F
1446  CPI 31 1
1448  JTC 1464
144B  CPI 37 7
144D  JTC 1465
1450  CPI 45 E
1452  JTZ 154D
1455  CPI 49 I
1457  JTZ 156E
145A  CPI 46 F
145C  JTZ 15AB
145F  CPI 54 T
1461  JTZ 1599
1464  HLT
1465  NDI 0F
1467  LLI 00
1469  LHI 14
146B  LLM
146C  LHI 11
146E  LMA
146F  CAL 3F44
1472  JMP 1628
1475
1476  LLM
1477  LHI 10
1479  CPM
147A  JTZ 148B
147D  LLI 01
147F  LHI 14
1481  LAM
1482  ADI 02
1484  LMA
1485  CPI 0E
1487  JFZ 1630
148A  HLT
148B  LLI 01
148D  LHI 14
148F  LAM
1490  ADI 01
1492  LMA
1493  CAL 3F44
1496  LLI 01
1498  LHI 14
149A  LLM
149B  LHI 10
149D  NDI 7F
149F  LCA
14A0  CPM
14A1  JTZ 14B6
14A4  LAL
14A5  CPI 09
14A7  JTZ 14AC
14AA  LAA
14AB  HLT
14AC  LLI 01
14AE  LHI 14
14B0  LMI 0D
14B2  LAC
14B3  JMP 1496
14B6  LLI 01
14B8  LHI 14
14BA  LAM
14BB  SUI 01
14BD  RRC
14BE  LMI 00
14C0  LLI 02
14C2  LMA
14C3  CAL 3F44
14C6  LLI 02
14C8  LHI 14
14CA  NDI 7F
14CC  CPI 23 #
14CE  JFZ 14DB
14D1  JMP 0A61
14D4
14D5
14D6
14D7
14D8
14D9
14DA
14DB  LBM
14DC  JMP 1634
14DF  LHI 11
14E1  LAM
14E2  LEL
14E3  CAL 0912
```

```
14E6  LDL
14E7  LLE
14E8  LMI 01
14EA  LLD
14EB  LHI 10
14ED  JMP 14F6
14F0
14F1
14F2
14F3
14F4
14F5
14F6  LAL
14F7  ADB
14F8  LLA
14F9  LAM
14FA  LHI 14
14FC  LLI 00
14FE  LLM
14FF  LHI 12
1501  LMA
1502  CAL 3F44
1505  NDI 7F
1507  CPI 3B ;
1509  JFZ 1511
150C  LAI 01
150E  JMP 1534
1511  NDI 0F
1513  CPI 00
1515  JTC 151D
1518  CPI 0A
151A  JTC 151E
151D  HLT
151E  LLI 1E
1520  LHI 14
1522  LMI 00
1524  RLC
1525  LAA
1526  LMA
1527  RLC
1528  RLC
1529  ADM
152A  JMP 1643
152D  NDI 0F
152F  LLI 1E
1531  LHI 14
1533  ADM
1534  LLI 00
1536  LHI 14
1538  LLM
1539  LHI 13
153B  LMA
153C  LAA
153D  LAA
153E  LAA
153F  LLI 00
1541  LHI 14
1543  LBM
1544  INB
1545  LMB
1546  LAB
1547  CPI 7F
1549  JMP 169C
154C
154D  CAL 3C4B
1550  CAL 3C4B
1553  CAL 3F44
1556  NDI 0F
1558  CPI 00
155A  JFC 155E
155D  HLT
155E  LLI 00
1560  LHI 14
1562  LBM
1563  LLI 03
1565  ADL
1566  LLA
1567  LMB
1568  CAL 3CC7
156B  JMP 16EA
156E  CAL 3C4B
1571  CAL 3C4B
1574  CAL 3F44
1577  NDI 0F
1579  CPI 00
157B  JFC 157F
157E  HLT
157F  CPI 0A
1581  JTC 1585
1584  HLT
1585  LLI 03
1587  ADL
1588  LLA
1589  LHI 14
158B  LBM
158C  LLI 00
158E  LLM
158F  LHI 11
1591  LMI 03
1593  LHI 12
1595  LMB
1596  JMP 1502
1599  CAL 3C4B
159C  CAL 3C4B
159F  LHI 14
15A1  LLI 00
15A3  LLM
15A4  LHI 11
15A6  LMI 02
15A8  JMP 1502
15AB  CAL 3CC7
15AE  LHI 14
15B0  LLI 00
15B2  LLM
```

```
15B3  LHI 11
15B5  LMI 04
15B7  LHI 14
15B9  LLI 00
15BB  LMI 00
15BD  LLI 00
15BF  LHI 11
15C1  LAM
15C2  CPI 01
15C4  JTZ 09AB
15C7  CPI 02
15C9  JTZ 15F0
15CC  JMP 0A06
15CF
15D0
15D1
15D2  LHI 12
15D4  LAM
15D5  LHI 13
15D7  LDM
15D8  LCI 01
15DA  JMP 168A
15DD  INB
15DE  JFZ 168C
15E1  DCC
15E2  JFZ 168A
15E5  DCD
15E6  JFZ 15D8
15E9  JMP 095A
15EC  INL
15ED  JMP 15BF
15F0  LHI 13
15F2  LEM
15F3  LDI 01
15F5  LCI 4F O
15F7  LBI 01
15F9  INB
15FA  JFZ 15F9
15FD  DCC
15FE  JFZ 15F7
1601  DCD
1602  JFZ 15F5
1605  DCE
1606  JFZ 15F3
1609  INL
160A  JMP 15BF
160D  LHI 17
160F  LBM
1610  INB
1611  LAB
1612  LHI 13
1614  CPM
1615  LHI 17
1617  JTZ 1621
161A  LMA
161B  LHI 12
161D  LLM
161E  JMP 15BF
1621  LAI 00
1623  LMA
1624  INL
1625  JMP 15BF
1628  LBA
1629  LHI 14
162B  LLI 01
162D  JMP 1476
1630  LAB
1631  JMP 1476
1634  LHI 10
1636  LAI 26 &
1638  ADB
1639  LLA
163A  LBM
163B  LHI 14
163D  LLI 00
163F  LLM
1640  JMP 14DF
1643  LMA
1644  CAL 3F44
1647  JMP 152D
164A
164B
164C
164D
164E  CAL 3CC7
1651  LHI 16
1653  LLI 4A J
1655  LBM
1656  CAL 3809
1659  LLI 4B K
165B  CAL 3809
165E  LLI 4C L
1660  CAL 3809
1663  LLI 4D M
1665  CAL 3809
1668  CAL 3CC7
166B  JMP 0A3F
166E  LHI 18
1670  LEM
1671  LAI 3F ?
1673  LBI FF
1675  OUT 0B
1676  LCA
1677  LAI 01
1679  OUT 09
167A  JMP 0930
167D
167E
167F
1680
1681
1682
1683
```

```
1684
1685
1686
1687
1688
1689
168A  LBI 3F 7
168C  LEA
168D  LAI 00
168F  OUT 09
1690  LAE
1691  OUT 08
1692  LAI 10
1694  OUT 09
1695  LAI 00
1697  OUT 09
1698  LAE
1699  JMP 15DD
169C  LHI 14
169E  LLI 00
16A0  LLM
16A1  DCL
16A2  LHI 11
16A4  LAM
16A5  CPI 01
16A7  JTZ 16B0
16AA  JMP 0988
16AD
16AE
16AF
16B0  CAL 3C4B
16B3  CAL 3F44
16B6  CPI 3B ;
16B8  JTZ 0976
16BB  LHI 14
16BD  LLI 00
16BF  LLM
16C0  DCL
16C1  LHI 18
16C3  NDI 0F
16C5  CPI 00
16C7  JFZ 16CB
16CA  HLT
16CB  CPI 0A
16CD  JTC 16D1
16D0  HLT
16D1  LBA
16D2  LAI 88
16D4  DCB
16D5  JTZ 16DD
16D8  ADI 0A
16DA  JMP 16D4
16DD  LMA
16DE  JMP 0976
16E1
16E2
16E3
16E4
16E5
16E6
16E7
16E8
16E9
16EA  CAL 3F44
16ED  JMP 09D0
16F0
```



ACTE DE NAISSANCE DE SUCELLUS

JEAN-FRANÇOIS DEGREMONT

ARTINFO/MUSINFO #28

Début de conception : Juin 1977

Début de construction (1ère version) : novembre 1977
(2ème version) : 15 janvier 1978

Objectifs recherchés :

- suppléer au manque de terminaux spécialisés pour des applications pratiques de l'enseignement en Intelligence Artificielle à Paris 8
- tenter de résoudre par des solutions inhabituelles de quelques problèmes sensori-moteurs en robotique
- élargir les activités du Département (Art et Informatique, Musique et Informatique, etc.)

Les moyens informatiques du Département sont : un T1600, un terminal PDP10, plusieurs microprocesseurs, un gros stock de pièces détachées d'origines diverses (CAB 500, Gamma 30,...)
Ceci a permis une construction à très bas prix (de l'ordre de 500Frs tout compris) en vue d'une connexion sur un Zilog 80.

La réalisation de ce terminal fait actuellement appel aux restes de :

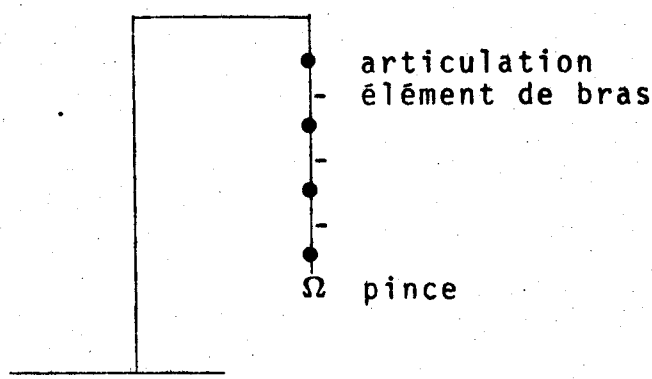
- 1 dérouleur de bandes magnétiques Gamma 30
- 1 photocopieuse
- 1 bicyclette
- 1 moteur d'essuie-glace
- 1 pompe à vide
- 1 lecteur de rubans perforés Gamma 30
- 1 machine à laver
- 1 tambour magnétique de CAB 500
- diverses tôleries

Son hardware est de conception simple et solide.

Son software est extrêmement développé et développable.

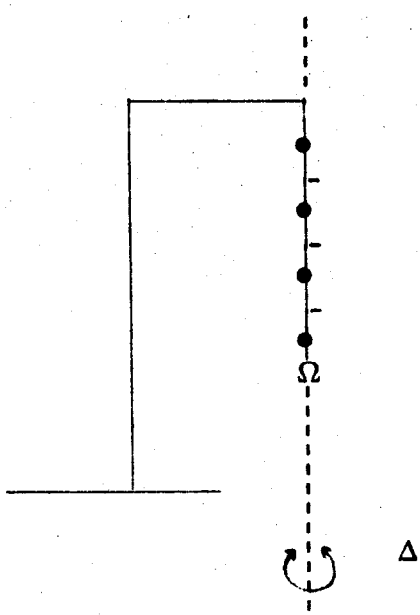
I - LA MÉCANIQUE

Principe

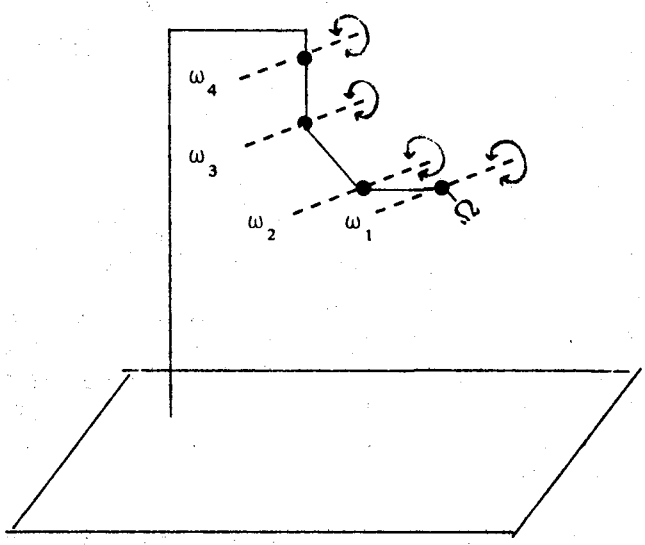


Déplacements

a) autour de l'axe Δ

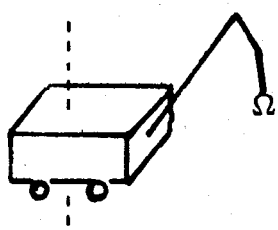


b) chaque élément peut tourner autour de l'axe ω_i par rapport à l'élément précédent. L'ensemble des éléments se déplace donc dans un même plan.

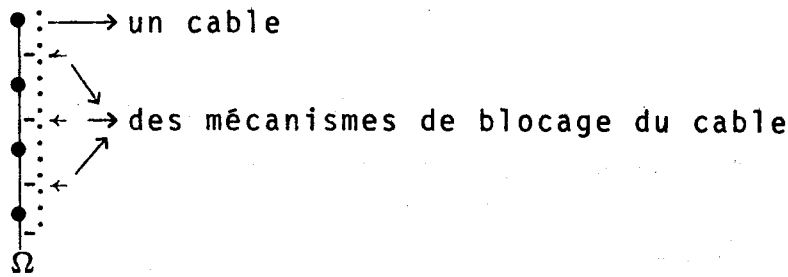


Analogie

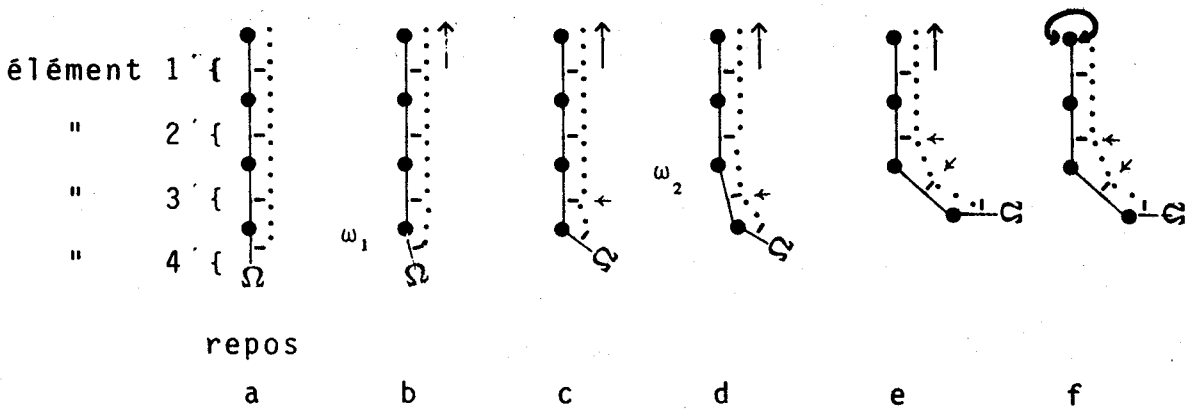
La pelle mécanique



L'idée de base



Mouvement



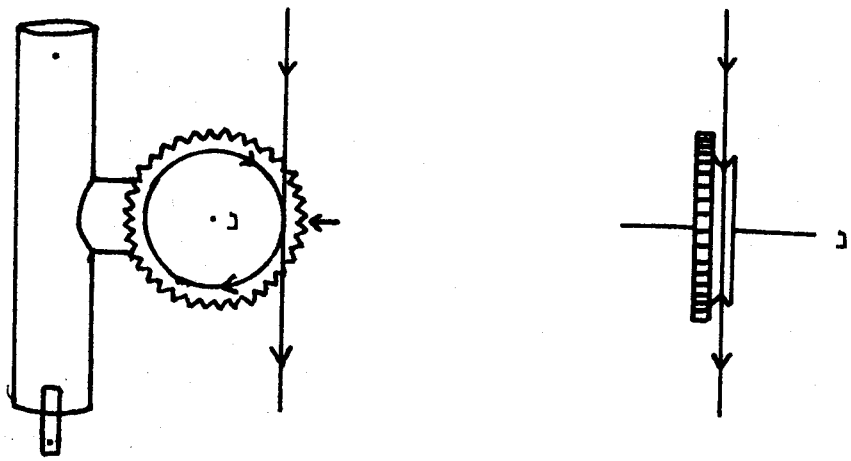
si on tire sur le câble (b), l'élément 4 tourne autour de ω_1. Lorsque la position désirée est atteinte, blocage (c). Si on continue à tirer sur le câble, c'est alors l'élément 3 qui tourne autour de ω_2 (d). On positionne ainsi successivement tous les éléments (e). Cependant qu'une rotation autour de Δ (f) permet de couvrir l'espace.

On voit tout de suite apparaître le premier inconvénient de ce principe : lorsqu'on a commencé à chercher une position, il n'est plus possible de changer sans repasser par l'état de repos. Plus exactement si on veut changer la position d'un élément du bras, tous les éléments précédents doivent revenir au repos. Sur le prochain modèle, ce problème sera résolu.

Avantages de ce principe

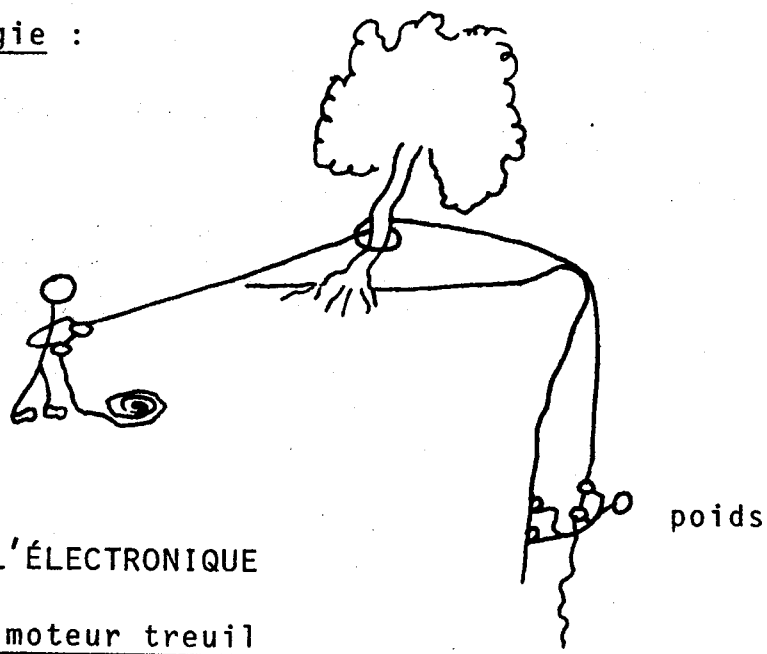
- ★ chaque point peut être atteint de différentes façons
  ⇒ effet tentaculaire
- ★ grande rigidité (limitée par l'élasticité du câble de traction)
  ⇒ moins d'oscillations lors des variations de vitesse ou des approches lentes
- ★ puissance limitée par la résistance du câble et par les blocages de segments. ce problème est mécaniquement simple
- ★ pas d'élément moteur dans le bras qui peut sans danger être soumis à des conditions de travail sévères
- ★ possibilités de construction légère du bras
  ⇒ accroissement de la charge utile
- ★ bonne précision, assez simple à obtenir par contrôle des points de blocage du câble.

Elément blocant



Le câble fait une boucle dans une poulie à gorge tournant librement sur l'axe ɔ. Cette poulie est solidaire d'une roue dentée légèrement plus grande. Un couteau basculant par électro-aimant peut venir se loger entre deux dents et ainsi bloquer la rotation de l'ensemble. Un couple - frein très important (blocage) est donc appliqué au câble.

Analogie :



II - L'ÉLECTRONIQUE

a) Le moteur treuil

- 1 commande de marche-arrêt, d'embrayage et d'inversion de sens de rotation. L'ordre est capté directement sur une des sorties de l'Intel 8080 et commande un relais (triac) après amplification.

- 1 photo-transistor qui envoie un pulse chaque fois qu'une dent passe devant lui : c'est une roue dentée

identique à celles montées sur le bras mais située en sortie du treuil. Ces pulses sont envoyés sur une entrée de l'Intel 8080 et permettent de contrôler indépendamment des à-coups moteurs la longueur de cable déroulée (cela permet d'éviter les problèmes de côut et de puissance posés par les moteurs pas-à-pas).

b) Le moteur de rotation

C'est un moteur muni d'un frein sur le rotor. Il est très démultiplié. En contrôlant les durées d'alimentation, on contrôle l'angle de rotation dont il est possible d'inverser le sens.

c) Electro-aimants

Une commande de blocage pour chaque élément de bras. Un relais transistorisé actionne, après amplification, un électro-aimant. Une commande de serrage de la pince agit de la même façon.

d) Des senseurs tactiles et de proximité

- les senseurs tactiles sont des micro-interrupteurs en bout de pince.
- les senseurs de proximité sont :
  - une cellule photo-électrique qui prévient lorsqu'un objet se trouve entre les deux branches de la pince
  - un circuit oscillant en limite de rupture,par capacité variable) prévient de l'approche d'un objet du bout de la pince.
- de nombreux autres capteurs ("moustaches", sonar,...) sont à l'étude et seront installés sur de prochaines versions de la pince.

e) Des alarmes diverses

Ce sont des micro-interrupteurs.
Toute la logique d'alarme est faite par software.

III - LE SOFTWARE

a) Le software de base

Sur le Zilog 80 il y a une horloge et un système d'entrées/sorties programmables. Ceci permet :

- de déclencher un événement extérieur (i.e. la mise en marche d'un moteur) dont la durée est contrôlée et arrêtée par un interrupt de l'horloge
- de ne pas explorer constamment les bits d'alarme. Les entrées masquables déclenchent un interrupt si nécessaire, ce qui provoque l'appel d'une routine d'intervention d'urgence ou l'incrémentation du compteur de dents.

Un temps maximum peut ainsi être consacré aux calculs divers.

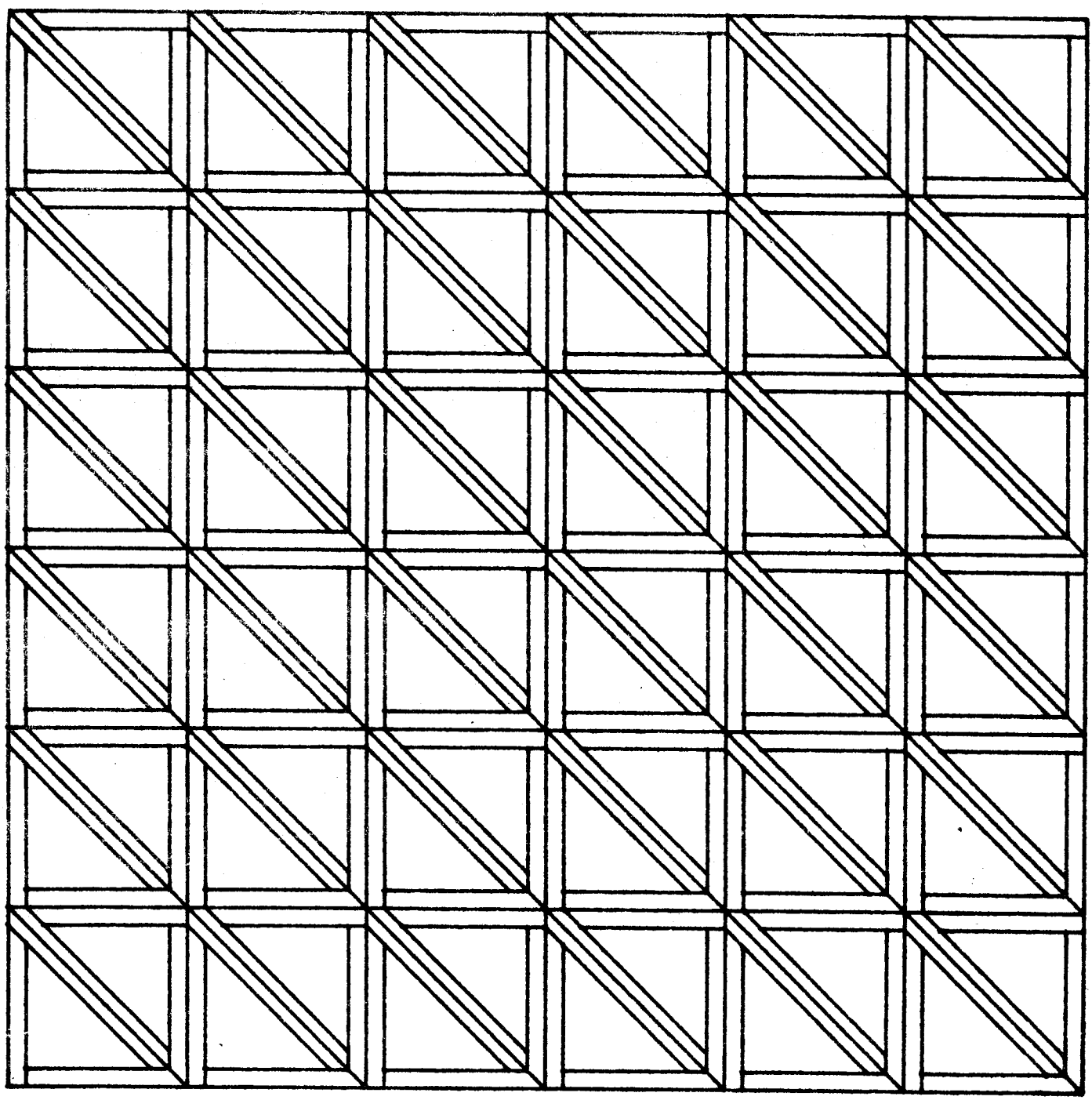
b) Le software évolué

- calcul automatique de trajectoires en modulant en fonction des obstacles
- tentative d'approximation dans la préhension d'un objet
- recherche d'objets mouvants sur une table lumineuse

Ces fonctions seront écrites en LISP ; ce langage ayant été implanté dernièrement par Jérôme CHAILLOUX sur des Zilog 80.

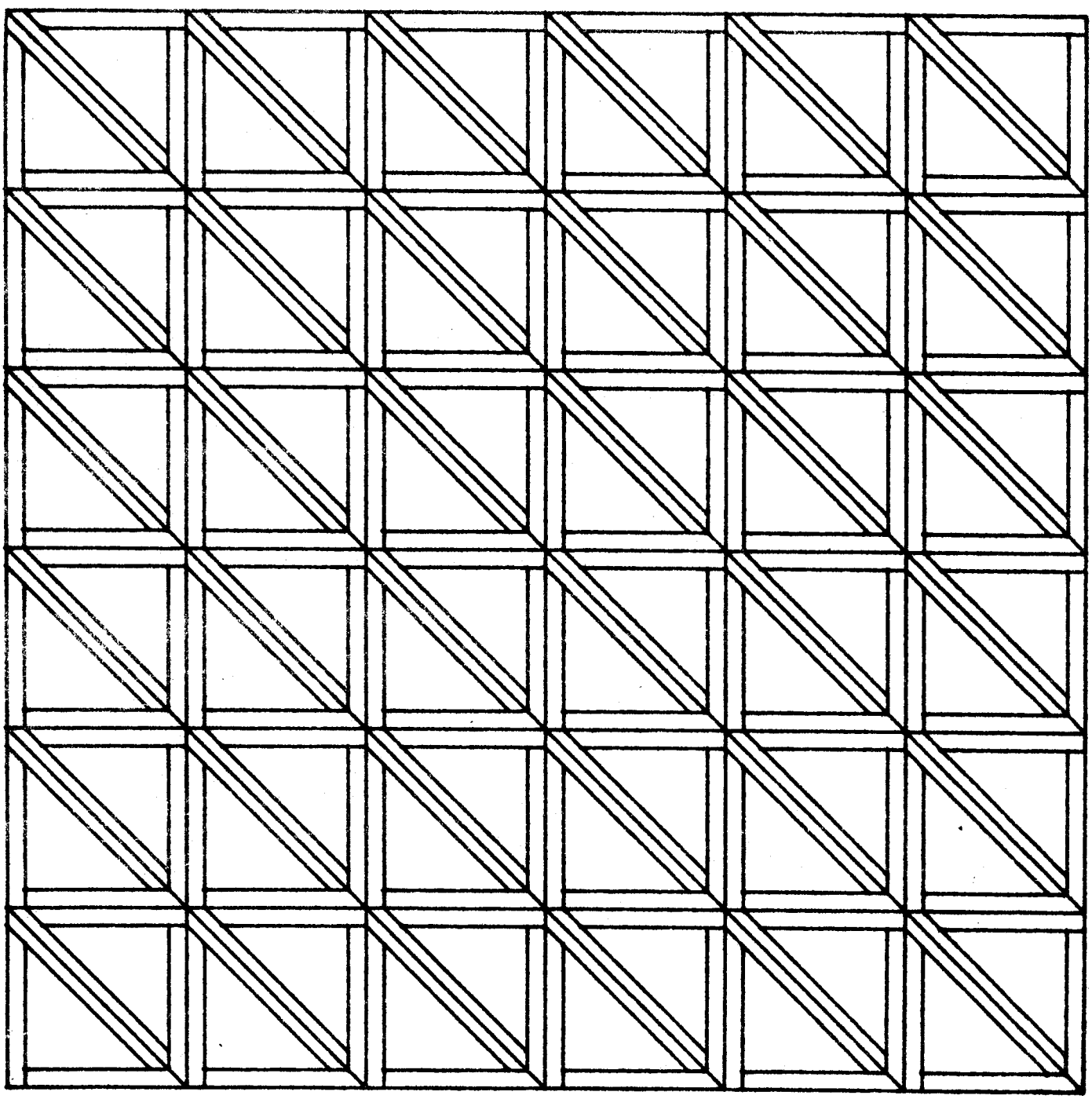
IMPOSSIBLE CRISTAL

JEAN-ERIC SCHOETTL
& HARALD WERTZ

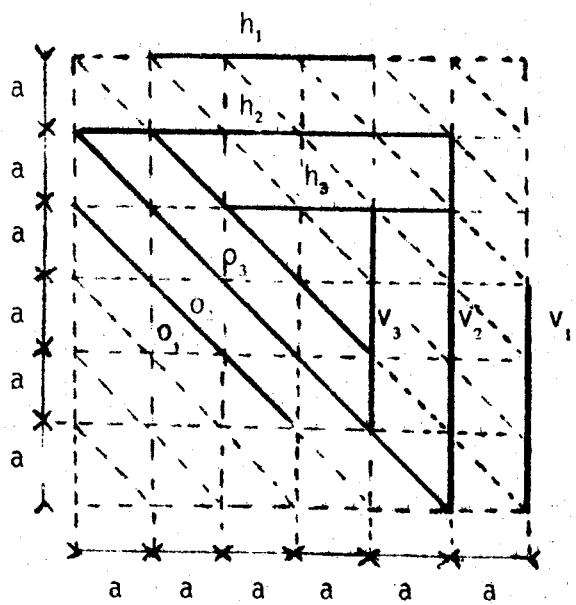


IMPOSSIBLE CRISTAL

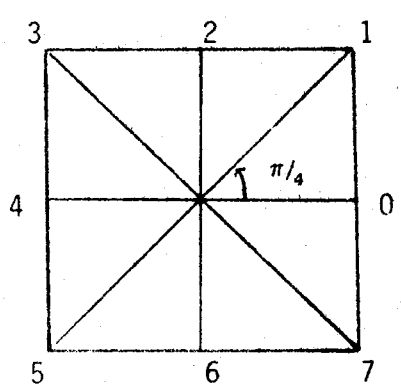
JEAN-ERIC SCHOETTL
& HARALD WERTZ



Voici une version LSE du programme qui fait tracer le dessin illusoire bien connu inspiré par M.C. Escher.



Le motif de base : 3 traits horizontaux (h_1,h_2,h_3), 3 traits verticaux (v_1,v_2,v_3), 3 traits obliques (o_1,o_2,o_3). a donne



Le pas élémentaire de la traçante et les 8 directions 0...7. 8 et 9 sont réservées aux lever et baisser de plume.

Un pas oblique : un pas droit × $\sqrt{2}$

```
 1   <<*****************************************************************
 2   <<
 3   <<                    I M P O S S I B L E   C R I S T A L
 4   <<
 5   <<*****************************************************************
 6
 7   MAIN PROCEDURE DESSIN
 8
 9   .COMMON SECTION COM
10     REF PROCEDURE PASELN;
11   WORD I,J,N,A,AP2,AP8,AP9,AP10,APN10;
12   WORD DIR,NPAS;
13   ARRAY 10 BYTE TQA=('8D,'0A,"QUEL A? ");
14   LPFILE QA=(MODE:OUTPUT,EM;EU:TS;DATA:TQA;EOE:10;CONTROL);
15   ARRAY 10 BYTE TQN=('8D,'0A,"QUEL N? ");
16   LPFILE QN=(MODE:OUTPUT,EM;EU:TS;DATA:TQN;EOE:10;CONTROL);
17
18   .KSTORE SECTION PILE
19         RES 50;
20
21   .LOCAL SECTION LOC
22         RES 1;
23   <<
24   <<*****************************************************************
25   <<
26
27    PROCEDURE LEC(DON)
28    .LOCAL SECTION CONTINUE LOC
29    POINTER WORD DON;
30   BYTE NA;
31   LPFILE LPNA=(MODE:INPUT,EM;EU:TK;DATA:NA;EOE:1;CONTROL);
32    .USING LOCAL IS LOC,COMMON IS COM;
33    READ LPNA;
34   &DON:= NA   AND 'F;
35    END; <<DE LEC.
36   <<
37   <<*****************************************************************
38   <<
39    PROCEDURE PLUME
40
41    .USING LOCAL IS LOC,COMMON IS COM;
42     RX:= NPAS;  RA:= DIR;
43     CALL PASELN;
44    END;  <<DE PLUME
45   <<
46   <<*****************************************************************
47   <<
48    PROCEDURE PENUP
49
```

```
 50        .USING COMMON IS COM,LOCAL IS LOC;
 51         RX:= 1 ; RA:= 8 ;  CALL PASELN;
 52         END;  <<  DE PENUP
 53        <<
 54        <<     ***********************************************************
 55        <<
 56        PROCEDURE PENDOWN
 57
 58        .USING COMMON IS COM,LOCAL IS LOC;
 59         RX:= 1 ; RA:= 9 ;  CALL PASELN;
 60         END;  <<  DE PENDOWN
 61        <<
 62        <<     ***********************************************************
 63        <<
 64       PROCEDURE NOB (DIR1,DIR2,DIR3)
 65
 66       .LOCAL SECTION CONTINUE LOC
 67        WORD DIR1,DIR2,DIR3;
 68
 69       .USING LOCAL IS LOC,COMMON IS COM;
 70
 71       DO FOR I:=1 STEP +1 UNTIL N;
 72       <<
 73       << LIGNE CONTINUE
 74       <<
 75         NPAS:= APN10;
 76         DIR:=DIR1;
 77         CALL PENDOWN;
 78         CALL PLUME;
 79       <<
 80       <<   RETOUR
 81       <<
 82         CALL PENUP;
 83         DIR:= DIR2;
 84         CALL PLUME;
 85        <<
 86        << SE DECALER VERS LA DROITE
 87        <<
 88         DIR:= DIR3;
 89         NPAS:= A;
 90         CALL PLUME;
 91        <<
 92        << 1 ERE LIGNE BRISEE
 93        <<
 94         DIR:= DIR1;
 95        DO FOR J:= 1 STEP +1 UNTIL N;
 96         NPAS:= AP2;
 97         CALL PLUME;
 98         CALL PENDOWN;
 99         NPAS:= AP8;
100         CALL PLUME;
101         CALL PENUP;
102             END;
103        <<
104        <<   RETOUR
105        <<
106         DIR:= DIR2;
107         NPAS:= APN10;
108         CALL PLUME;
109        <<
110        << SE DECALER VERS LA DROITE
111        <<
```

```
112        DIR:= DIR3;
113        NPAS:= AP8;
114        CALL PLUME;
115       <<
116       <<  2 EME LIGNE BRISEE
117       <<
118        DIR:= DIR1;
119       DO FOR J:= 1 STEP +1 UNTIL N;
120        NPAS:= A;
121        CALL PLUME;
122        CALL PENDOWN;
123        NPAS:= AP8;
124        CALL PLUME;
125        CALL PENUP;
126        NPAS:= A;
127        CALL PLUME;
128             END;
129       <<
130       <<   RETOUR
131       <<
132        NPAS:= APN10;
133         DIR:= DIR2;
134         CALL PLUME;
135       <<
136       << DECALER VERS LA DROITE
137       <<
138        NPAS:= A;
139        DIR:= DIR3;
140        CALL PLUME;
141
142        END;
143
144       <<
145       <<   DERNIER GRAND TRAIT
146       <<
147        CALL PENDOWN;
148        NPAS:= APN10;
149        DIR:= DIR1;
150        CALL PLUME;
151
152       END;     << DE NOB.
153       <<
154       <<      ***********************************************************
155       <<
156       PROCEDURE OB(BINF,BSUP,PAS)
157
158        .LOCAL SECTION CONTINUE LOC
159      WORD BINF,BSUP,PAS;
160
161        .USING LOCAL IS LOC,COMMON IS COM;
162
163      I:= BINF-PAS;
164      DO; I:= I+PAS;
165      IF (PAS<0) THEN IF (I<BSUP) THEN EXIT OB; END; END;
166      IF (PAS>0) THEN IF (I>BSUP) THEN EXIT OB; END; END;
167
168       <<
169       <<  AVANCER SUR LE COTE
170       <<
171        CALL PENUP;
172        IF (PAS>0) THEN
173        DIR:= 2;
```

```
174        NPAS:= AP9;
175        CALL PLUME;
176         ELSE
177      DIR:= 0;
178        NPAS:= AP10;
179      CALL PLUME;
180      DIR:= 6;
181      NPAS:= A;
182      CALL PLUME;
183      END;
184
185       <<
186       << 1 ERE LIGNE BRISEE
187       <<
188        DIR:= 7;
189
190       DO FOR J:= 1 STEP +1 UNTIL I;
191        CALL PENDOWN;
192        NPAS:=AP8;
193        CALL PLUME;
194        CALL PENUP;
195        NPAS:= AP2;
196        CALL PLUME;
197      END;
198
199      <<
200      <<      RETOUR
201      <<
202      NPAS:=I*AP10;
203        DIR:= 3;
204      CALL PLUME;
205      <<
206      << SE DECALER D'UN CRAN
207      <<
208        DIR:= 2;
209      NPAS:=A;
210      CALL PLUME;
211      <<
212      <<  LIGNE CONTINUE
213      <<
214      CALL PENDOWN;
215      NPAS:=I*AP10;
216        DIR:= 7;
217      CALL PLUME;
218      <<
219      <<   RETOUR
220      <<
221      CALL PENUP;
222        DIR:= 3;
223      CALL PLUME;
224      <<
225      <<   DECALER
226      <<
227        DIR:= 2;
228      NPAS:=A;
229      CALL PLUME;
230      <<
231      << 3IEME LIGNE BRISEE
232      <<
233        DIR:= 7;
234
235      DO FOR J:=1 STEP +1 UNTIL I;
```

```
236        NPAS:= A;
237        CALL PLUME;
238        CALL PENDOWN;
239        NPAS:=AP8;
240        CALL PLUME;
241        CALL PENUP;
242        NPAS:= A;
243        CALL PLUME;
244      END;
245
246      <<
247      <<RETOUR
248      <<
249        DIR:= 3;
250      NPAS:=I*AP10;
251      CALL PLUME;
252      <<
253      << REDESCENDRE UN BRIN
254      <<
255      DIR:= 6;
256      NPAS:= A;
257      CALL PLUME;
258
259       END;
260
261      END;        <<DE OB
262       <<
263       <<         ************************************************************
264       <<
265
266
267      .USING KSTORE=PILE,LOCAL=LOC,COMMON=COM;
268
269      WRITE QA;
270       CALL LEC(@A);
271      WRITE QN;
272       CALL LEC(@N);
273
274      A:= 10*A;
275      AP2:= 2*A;
276      AP8:= 8*A;
277      AP9:= 9*A;
278      AP10:= 10*A;
279      APN10:= AP10*N;
280
281      <<
282      <<  LIGNES HORIZONTALES ET VERTICALES.
283      <<
284      CALL NOB (6,2,0);
285      CALL PENUP;
286      NPAS:= APN10;
287      DIR:=2;
288      CALL PLUME;
289      DIR:=4;
290      CALL PLUME;
291      CALL NOB(0,4,6);
292      CALL PENUP;
293      NPAS:= APN10;
294      DIR:=4;
295      CALL PLUME;
296
297      <<
```

```
298     << LIGNES OBLIQUES.
299     <<
300     CALL OB(1,N,1);
301     CALL OB(N-1,1,-1);
302
303     END.
304          FIN
```

URGENT

NE PAS AFFRANCHIR

ERRATA

MARC BATTIER
ARTINFO/MUSINFO #27

Page 32, figure 7 : à l'adresse DAC 80
correspond le dispositif FILTRE2

Page 33, figure 9 : il manque un point en A13.

Page 33, figure 8 :

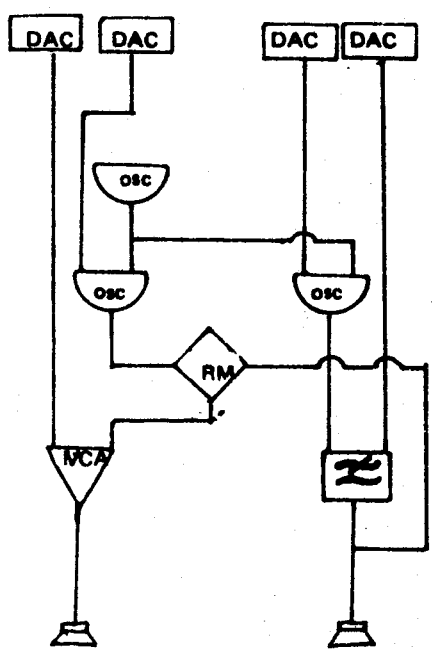


Figure 8

A V E R T I S S E M E N T

Le présent bulletin répond à une visée toute didactique : livrer sous forme accessible aux nouveaux venus dans les groupes de travail courants

- de l'information technique et bibliographique en rapport avec leurs disciplines
- des programmes commentés de tous niveaux permettant un accès relativement rapide à des techniques de programmation appropriées, ainsi qu'a une implémentation aisée.

On s'est efforcé, dans la mesure du possible, de ne pas établir de clivage trop net entre les disciplines concernées (musique, arts plastiques, poésie, architecture, logique , informatique) , mais tout au contraire de les unifier, ne serait-ce que par des techniques de programmation communes.

L'aspect pédagogique d'ARTINFO/MUSINFO reflète une préoccupation constante du Groupe, à savoir ne pas se satisfaire en dernier ressort de méthodes de programmation trop élémentaires.

ARTINFO/MUSINFO est imprimé au Département d'Informatique de l'Université Paris 8 (Vincennes). Grâces soient rendues aux soins diligents de Jacqueline BERTOUT et Philippe PINON.

Pour tous renseignements et composition des livraisons à venir, s'adresser à :

Jacques ARVEILLER
Département d'Informatique
Université Paris 8
Route de la Tourelle
75571 PARIS CEDEX 12

Pour tout envoi, s'adresser à Patrick GREUSSAY , même adresse.

EMPLACEMENT LIBRE

UNIVERSITÉ PARIS VIII

ART-INFØ
28
MUS-INFØ

GROUPE ART ET INFORMATIQUE

1978