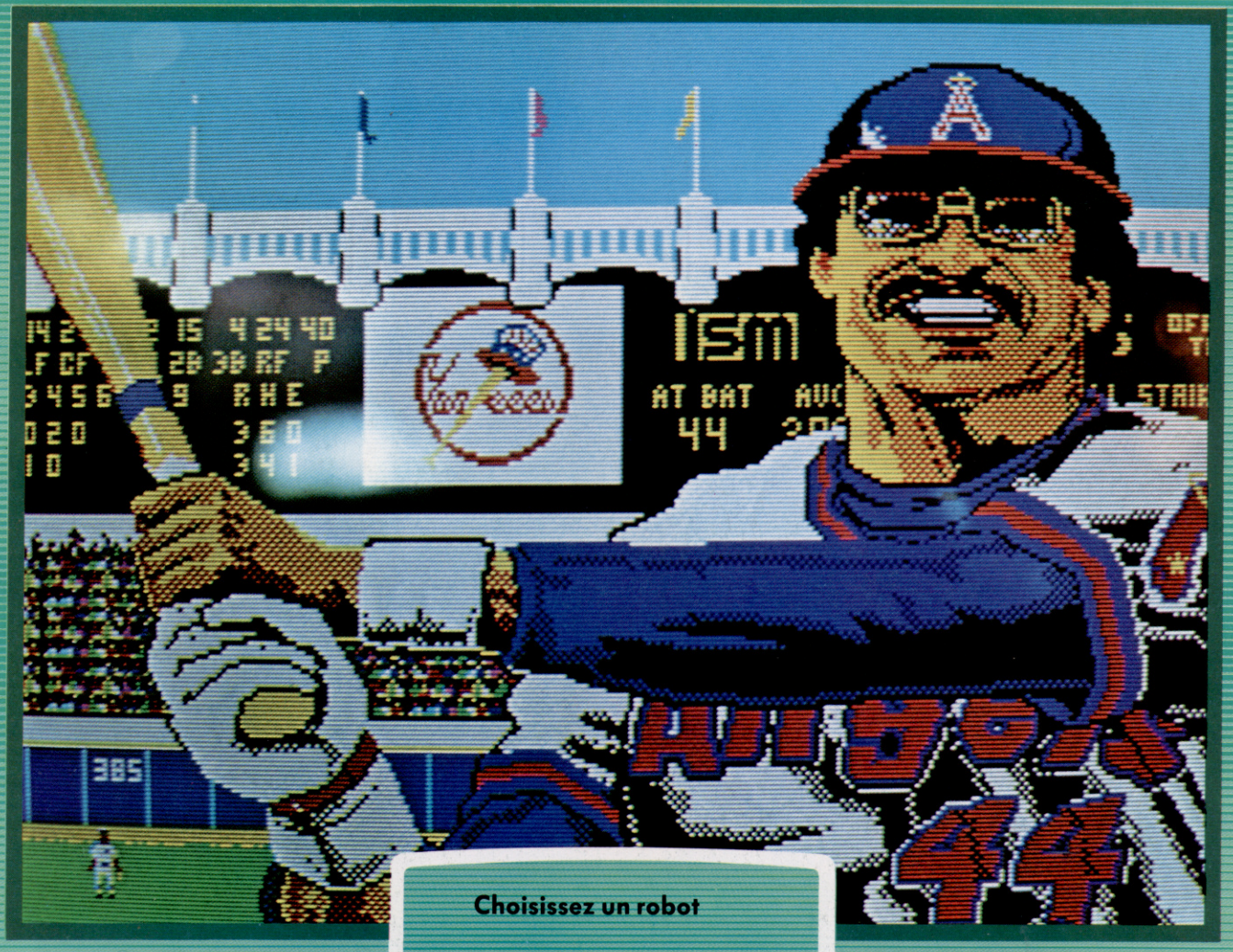
ABC INFORMATIQUE

N°67

COURS D'INFORMATIQUE PRATIQUE ET FAMILIALE



Choisissez un robot

Logiciel "vertical"

L'Osborne Encore

Faire des organigrammes

EDITIONS ATLAS

M 6062-67-12,50 F 90 FB - 3,80 FS - $ 1.95

Tenir l'affiche

Parmi les bras de robot haut de gamme disponibles sur le marché, deux catégories se dégagent : les robots utilisés à des fins didactiques et ceux qui représentent le nec plus ultra de la technique.

Si la plupart des robots décrits ici sont chers — certains dépassent 10 000 F —, ils ne sont pas pour autant des robots industriels; ils gardent comme vocation les utilisations domestique et pédagogique. Un premier groupe comprend les robots de conception très avancée qui mettent en œuvre de nombreuses caractéristiques des bras de robots industriels. La différence principale vient de leur vocation à être avant tout un outil à objectif didactique.

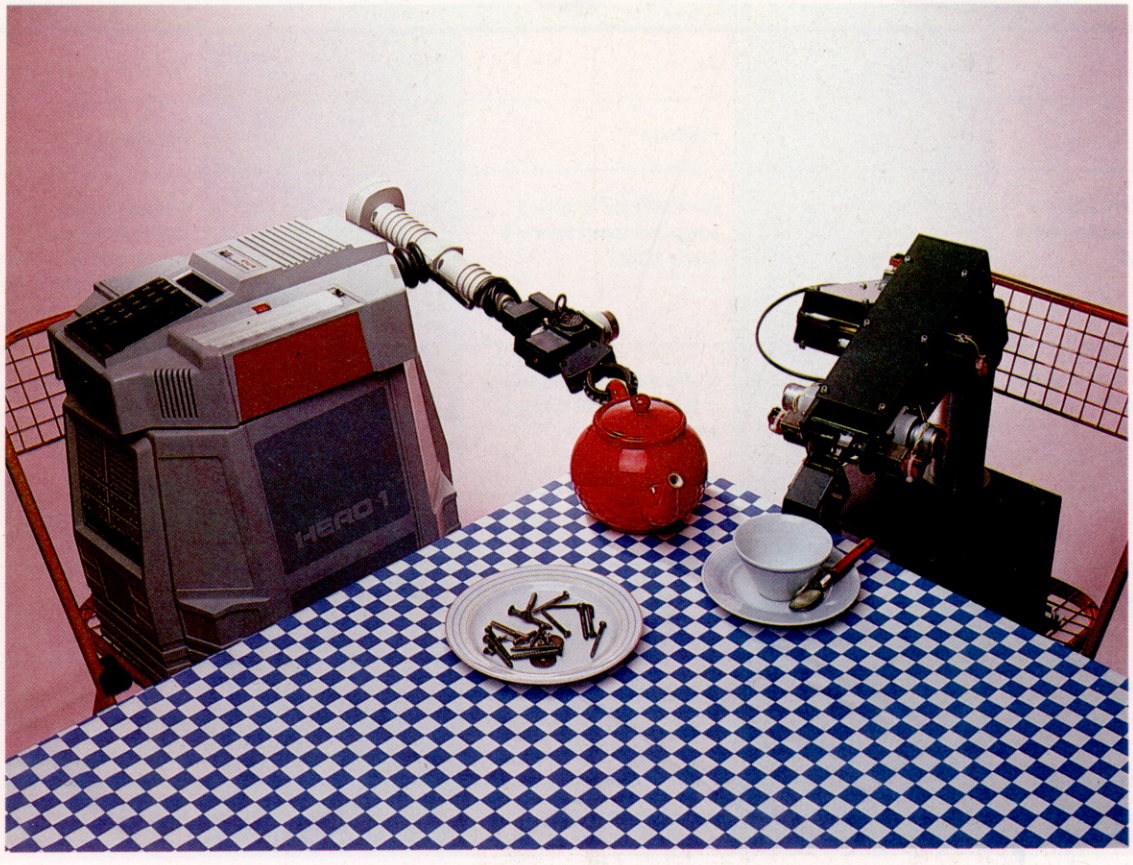
Généralement, ces bras « éducatifs » sont plus petits et incapables de manipuler de grosses charges. Les marchés industriels et éducatifs se rejoignent néanmoins lorsque les applications supposent un bras de robot léger et de faible encombrement. Par exemple, un robot industriel sera nécessaire pour déplacer de grosses barres d'acier pesant plusieurs tonnes. De même, un robot industriel spécifique sera utilisé pour assembler des composants électroniques sur des cartes à circuits imprimés, tâche pour laquelle un bras de robot puissant et massif ne convient pas. Les robots de cette catégorie peuvent donc avoir à la fois des applications industrielles et éducatives.

La seconde catégorie se rapporte à des robots qui mettent en œuvre les derniers développements en matière de recherche. Parmi eux, de nombreux robots sont équipés des systèmes de capteurs que nous avons décrits dans les articles précédents sur la robotique.

Bras d'enseignement

Mentor est un bras de robot de chez Cybernetic Applications d'un prix relativement abordable. Il est vendu en kit. Il comporte six degrés de liberté pour les articulations de la taille, de l'épaule, du coude, ainsi que trois axes de rotations pour le poignet. Il est dirigé par trois moteurs et peut être contrôlé depuis le Vic-20 de Commodore ou encore le Spectrum de Sinclair.

De la même société, mais à l'autre extrémité de la gamme, les Neptune 1 et Neptune 2 coûtent cinq fois plus cher. Ils sont également livrés en kit, et leur bras peut lever jusqu'à 2,500 kg.



A votre service
Parce que Hero est à la fois mobile et doté d'une pince pour prendre des objets, il est capable d'apporter à son propriétaire une tasse de café au petit déjeuner (pourvu que la chambre ne soit pas à l'étage...) La photographie montre Hero et Mentor en train de prendre ensemble une tasse de thé.
(Cl. Ian McKinnell.)

Ils sont mus hydrauliquement, bien qu'ils utilisent de l'eau à la place du liquide hydraulique habituel à ce type de matériel. Ils sont également contrôlés depuis les mêmes micro-ordinateurs. Le Neptune 2 a deux vitesses de fonctionnement, ce qui permet des déplacements rapides du bras pour les mouvements de plus grande amplitude et un déplacement lent pour un travail plus précis.

Le long bras d'apprentissage

Les deux domaines les plus significatifs de la recherche en micro-électronique sont la conception des composants et la robotique. S'il reste difficile d'expérimenter de nouveaux microcomposants avec un micro familial, il en va tout autrement avec les robots. La première utilité d'un bras de robot est d'améliorer la compréhension de la robotique. A cette fin, la meilleure façon d'accroître ses connaissances en la matière est d'étudier les systèmes en cours de fonctionnement et d'essayer de les améliorer. Les bras montrés ici, le Micro Grasp de Powertran Cybernetics et l'Armdroid de Cone Robotics sont programmables par micro-ordinateur et comportent cinq degrés de liberté. On peut les acheter en kit ou déjà assemblés pour des prix variant dans une fourchette de 3 500 à 6 000 F. (Cl. Chris Stevens.)

Nom	NEPTUNE 1	NEPTUNE 2	MENTOR	GENESIS P101	HERO
Type	Bras	Bras	Bras	Bras	Robot au sol
Fonction principale	Pédagogique	Pédagogique	Pédagogique	Pédagogique	Pédagogique
Capteurs	Potentiomètre pour enregistrer la position ; main pouvant apprécier la distance de l'objet	Potentiomètre pour enregistrer la position ; main pouvant apprécier la distance à l'objet	Potentiomètre pour enregistrer la position ; main pouvant apprécier la distance à l'objet	Retour de position	A ultrasons, permettant la détection du mouvement. Les capteurs distinguent 256 niveaux de lumière et de son. Capteurs tactiles sur la main-pince
Degrés de liberté	6 : taille, épaule, coude et élévation du poignet, mouvement tournant du poignet et main-pince	7 : taille, épaule, coude, élévation du poignet, mouvement tournant du poignet, inclinaison du poignet et main-pince	6 : taille, épaule, coude, élévation du poignet, mouvement tournant du poignet et main-pince	6 : taille, épaule, coude, élévation du poignet, mouvement tournant du poignet et main-pince	4 : épaule, coude, poignet et main-pince
Prix (ordre de grandeur)	1 300 F	1 900 F	3 600 F	18 000 F	21 000 F
Alimentation	Hydraulique : pompe à eau sur secteur	Hydraulique : pompe à eau sur secteur	Sur secteur	Hydraulique : pompe à eau sur secteur	Accus rechargeables
Connexion	Spectrum, Vic-20	Spectrum, Vic-20	Spectrum, Vic-20	Programmé par module de contrôle, interface standard RS232 Spectrum, Vic-20	Avec un assembleur croisé, Hero peut être utilisé sur tout micro doté d'un port série
Constructeur	Cybernetics Applications Ltd.	Cybernetics Applications Ltd.	Cybernetics Applications Ltd.	Powertran Cybernetics	Zenith Data Systems

Powertran Cybernetics construit le Genesis P101, qui est doté de six degrés de liberté et est vendu en kit. Motorisé hydrauliquement, le modèle présenté ici est livré avec un module de contrôle pour la programmation du robot, et avec l'interface standard RS232 qui permet la liaison avec la plupart des micro-ordinateurs du marché. Le prix du bras livré préassemblé et testé est nettement plus élevé.

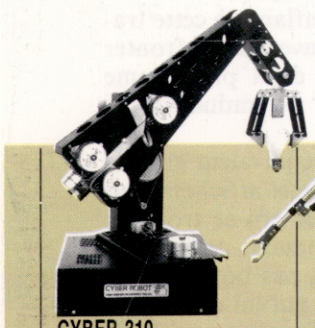
Changer de vitesse

A un prix moyen, on trouve le bras Cyber 310 de Cyber Robotics. Il existe en versions Jupiter Ace, Apple II, Commodore PET 3000/4000 et 8000, Hector HRX, etc. Il est alimenté par des moteurs pas-à-pas mais ne peut lever que des charges maximales de 250 g, ce qui fait assez poids plume. Néanmoins, la gamme d'options possibles est surprenante.

Outre les cinq degrés de liberté, il permet à l'utilisateur d'accélérer et de ralentir son déplacement du bras; ce qui signifie qu'il peut simuler les mouvements de l'homme. Il peut ainsi modifier sans cesse sa vitesse selon les changements de nature de la tâche effectuée et simuler les effets d'inertie. Les articulations peuvent jouer simultanément et la position du bras peut être déterminée soit de manière relative (en demandant par exemple au bras d'avancer de *x* unités à partir de sa position courante), soit de manière absolue (en déterminant un déplacement sur une position référencée par rapport à un point de départ). Il peut être programmé en BASIC et dans une version du FORTH appelée Robo FORTH, développée par Cyber Robotics.

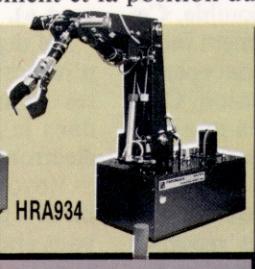
En montant dans l'échelle des prix, nous en venons aux HRA933 et HR934 de Feedback Instruments, qui sont livrés assemblés. Tous les deux sont alimentés hydrauliquement, comportent cinq degrés de liberté et peuvent soulever une charge de 1,350 kg avec une précision de positionnement de 3 mm. Outre des capteurs de position pour les articulations du bras, ils possèdent également des capteurs tactiles à leurs extrémités. Ces derniers indiquent que le bras a pris quelque chose et permettent de contrôler la force nécessaire pour soulever un objet.

Le contrôle se fait par l'intermédiaire de l'interface RS232, et des instructions spéciales doivent être données pour utiliser l'Apple II, le Tandy TRS-80, le Commodore PET ou d'autres micro-ordinateurs.



CYBER 310

HRA933



HRA934

Bras	Bras	Bras
Pédagogique •	Pédagogique	Pédagogique
Aucun	Capable de déterminer sa position; main pouvant apprécier la distance	Peut déterminer sa position; main pouvant apprécier la distance
6 : rotation de la base, de l'épaule (jusqu'à 180°), coude, élévation du poignet, rotation du poignet et main-pince	5 : rotation de la base, coude, rotation du poignet, abaissement du poignet et main-pince	5 : rotation de la base, coude, rotation du poignet, abaissement du poignet et main-pince
7 000 F	22 500 F	28 000 F
Sur secteur	Sur secteur	Sur secteur
Jupiter Ace, Apple II, Commodore PET et Cie, et Hector HRX	Apple II, Tandy TRS80, Commodore PET et Cie, AIM 65	Apple II, Tandy TRS80, Commodore PET et Cie, AIM 65
Cyber Robotics	Feedback Systems Ltd.	Feedback Systems Ltd.

Dernier cri

Hero-1 de Zenith Data Systems est vendu en kit; il présente des caractéristiques remarquables. Il est mobile et dispose d'un bras doté d'un système de coordonnées sphériques, grâce auquel le bras peut s'allonger et se rétracter de manière télescopique.

Hero est équipé d'une vaste gamme de capteurs pour détecter le mouvement, le son et la lumière, et même d'un capteur de distance à ultrasons pour éviter les collisions, ainsi que d'un synthétiseur de parole qui confère au robot un vocabulaire sans limite. Son bras offre cinq degrés de liberté. Le temps de montage de Hero-1 est considérable, aussi est-il possible de le commander assemblé. Mais le prix monte alors de manière étourdissante...

Tous ces robots ne sont guère mieux que de coûteux amusements. Leur utilisation principale, jusqu'ici, a été la promotion publicitaire de certaines sociétés qui les ont utilisés pour distribuer des prospectus sur leur stand d'exposition, lors de salons professionnels... Ils n'en représentent pas moins la technologie la plus avancée à l'heure actuelle.

Tous utilisent intelligemment des capteurs, se déplacent de manière organisée et sont dotés de bras intelligents. Aucun n'a de système de vue, mais ils peuvent parler, percevoir des signaux acoustiques et leur répondre.

Il va sans dire que leur prix arrêtera beaucoup d'acheteurs éventuels, mais ils restent représentatifs de ce que vous pouvez essayer de réaliser, et de ce vers quoi vous pouvez tendre en construisant votre propre robot. Mais ils ont également une autre signification d'ordre historique : il y a quelques années, de tels robots auraient tout simplement été impensables, même à n'importe quel prix...

Décollage vertical

« Logiciel vertical » est un mot du jargon informatique donné aux programmes spécialisés écrits pour des catégories socio-professionnelles bien déterminées.

Un restaurateur utilise un tableur pour se faire une idée de ce que ses clients commandent, afin de mieux planifier ses futurs menus. Une crèche utilise le même programme fonctionnant sur quatre micro-ordinateurs Commodore 64, pour gérer les coûts énergétiques intervenant dans son budget. Un chirurgien utilise une base de données pour étayer ses recherches sur les causes et les traitements possibles du cancer.

Bientôt, les pharmaciens utiliseront des micro-ordinateurs pour traiter les ordonnances médicales, qui devront être imprimées et non écrites à la main.

Un architecte utilise un programme pour agencer au mieux les éléments d'une cuisine ou les meubles des chambres à coucher, en tirant le meilleur parti possible de la place disponible. Cet outil informatique lui est devenu tellement indispensable qu'il s'est associé avec les concepteurs du programme pour en assurer la diffusion. Chaque logiciel, particulièrement adapté à une profession, est appelé « vertical » par opposition aux logiciels standards ou « horizontaux ».

Tous ces exemples ne sont que quelques applications nouvelles de l'ordinateur, des réponses à la question que l'on ne cesse de poser, « que peut-il faire? ». N'oublions pas qu'il a été calculé que l'utilisateur moyen d'un ordinateur n'utilise sa machine qu'à 10 % de ses capacités, et ce taux semble encore surévalué. Dépenser 2 000 F pour une machine dont l'usage est ensuite limité à une seule application, qu'il s'agisse de jeux ou de comptabilité, ne constitue pas un investissement vraiment justifié, alors que la modularité de l'ordinateur n'est pas pleinement exploitée.

Il existe deux manières d'exploiter au mieux les possibilités de l'ordinateur : la première est de trouver de nouvelles applications pour les logiciels standards. Un photographe professionnel pourra utiliser le module gestion de stock d'un progiciel de comptabilité pour assurer l'inventaire et l'exploitation de ses négatifs et diapositives provenant de commandes passées. De la même manière, une antenne de l'Agence nationale pour l'emploi pourrait utiliser une base de données pour mettre en rapport les besoins en personnel de ses entreprises clientes et le fichier de ses demandeurs d'emploi. Les données relatives à ces derniers pourraient être consignées sur disques durs, et listées automatiquement sous forme de courriers personnalisés aux entreprises. Cette utilisation revient bien moins cher que celle de progiciels spécifiques.

Conception sur mesure

L'autre possibilité est de rechercher un progiciel pour l'utilisation spécifique que vous recherchez. Supposons que vous soyez un étudiant en théologie dont le bureau est surchargé de pesants infolio, glossaires, dictionnaires, commentaires bibliques, etc. Il se peut alors que vous soyez intéressé par tel progiciel du marché qui comprend la totalité de la traduction de la Bible, avec des fonctions complètes de recherche pour la création de références croisées sur disque.

Et, si vous ne faites pas confiance à cette traduction de la Bible, vous pouvez la confronter à l'original grec au moyen d'un programme appelé Greek Transliterator (« traducteur de grec »).

Ce dernier vous donnera l'équivalent grec de n'importe quels mot ou phrase et affichera toutes les occurrences concernées qui se trouvent dans le texte traduit. Évidemment, cette étude électronique de la Bible n'est pas bon marché.

Voyons un autre progiciel qui lui aussi semblera à première vue tout à fait inconcevable. Si vous étiez un ingénieur chargé de faire les calculs concernant la mise au point de drains et d'égouts susceptibles de faire face aux pires orages... alors vous apprécieriez certainement beaucoup MIDUSS (*McMaster Interactive Design of Stormwater Systems,* « conception interactive de systèmes d'écoulement d'eau »). Ce dernier vous permet de calculer le diamètre des diverses canalisations d'eau à mettre en place, ainsi que l'importance des bassins de retenue.

Vous pourrez en outre dresser des représentations hydrographiques et effectuer des recopies d'écran de votre travail en mode haute résolution. Il vous faudra pour cela avoir par exemple un Sirius 256 K, et mettre approximativement 8 000 F dans le progiciel.

Les systèmes d'écoulement d'eau ont, semble-t-il, énormément inspiré les programmeurs... L'ordinateur portable Husky est en effet au centre d'un système anglais d'inspection des conduites d'eau, CAMIL (*Computer-Aided Manhole Inspection Location*).

Ce programme conçu à Southampton a été étendu à toute la Grande-Bretagne par les autorités compétentes. Il permet à des techniciens-vérificateurs de saisir sur place les données recueillies et de les transmettre par téléphone à des ordinateurs centraux, qui sont alors à même de sortir des états sur la disposition des systèmes d'écoulement d'eau ainsi vérifiés. Le programme

peut également répondre à des recherches telles que : « Communiquez la liste de tous les systèmes d'écoulement en brique construits avant 1900. » Et il y en a beaucoup...

Oui ou non

Les voyageurs de commerce trouveront très utiles les logiciels Travelling qui sont exploités sur un autre ordinateur portable, le NEC PC-8201. Outre les progiciels de base tels que Travelling Writer, traitement de texte pour établir des rapports et doté de fonctions de fusion de documents (courrier personnalisé), cette gamme comprend Travelling Project Manager (planification des déplacements), Travelling Appointment Manager (planification des rendez-vous), Travelling Sales Manager (gestion des ventes), et, surtout, Travelling Expense Manager (gestion des frais de déplacement).

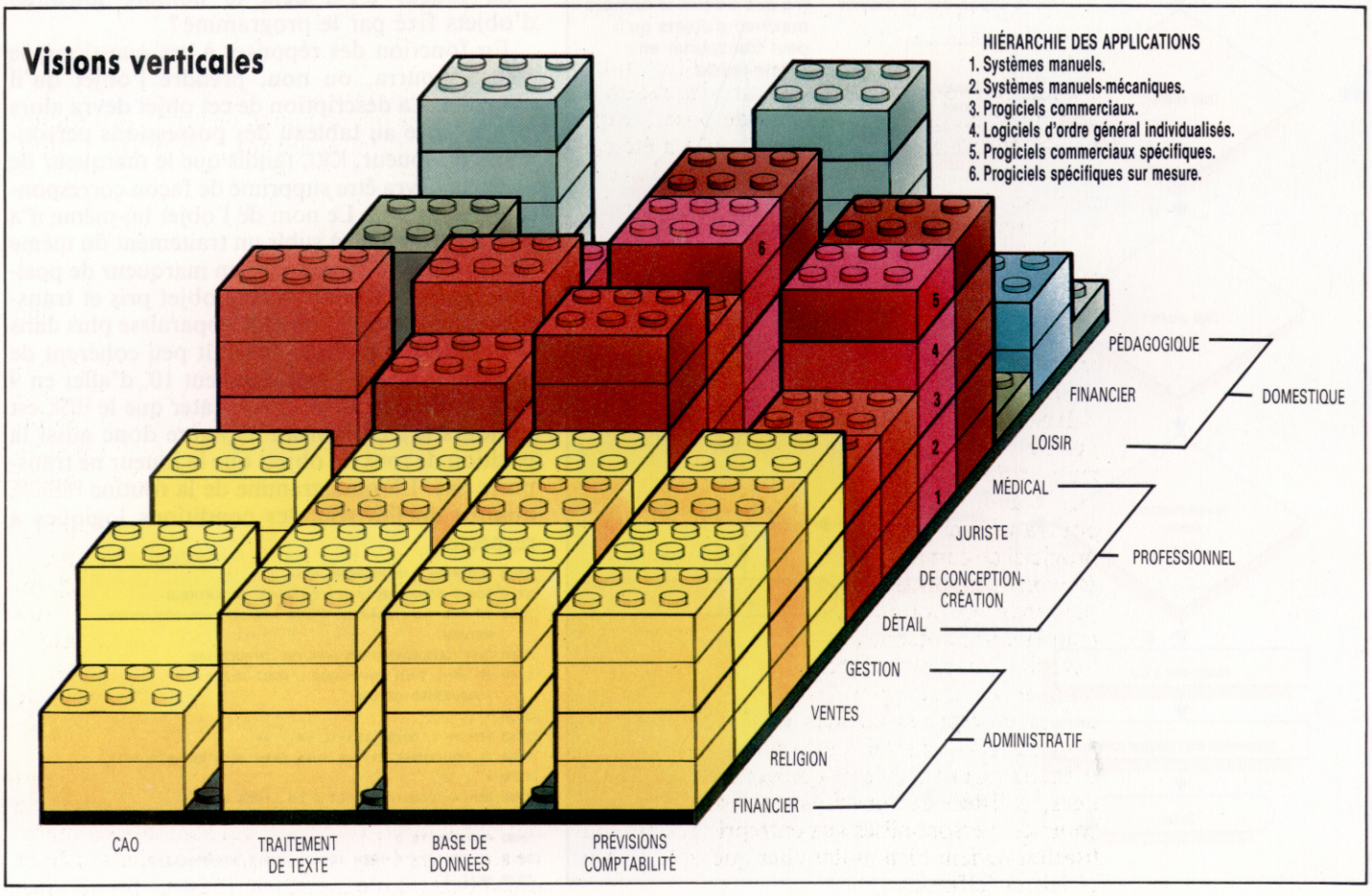
Les vendeurs peuvent également améliorer leurs méthodes de vente avec le module Sales Edge (approches de la vente), qui fait partie des programmes Human Edge (le facteur humain). Ces programmes sont conçus par Thorn EMI pour l'IBM PC et l'Apricot. Le module Sales Edge s'efforce de déterminer les points forts et les points faibles de son utilisateur, en lui posant une série de questions et de réponses auxquelles il doit répondre seulement par oui ou par non. Après une série de questions du même genre concernant le client, il suggère une politique de vente avec des « manœuvres » d'« entrée de jeu » et de « conclusion » pour une affaire. Ce programme n'est, à nouveau, pas très bon marché. Le programme complet Human Edge, qui comprend la gestion, la négociation et la communication, coûte près de 10 000 F. Si vous préférez jouer sur les marchés monétaires, vous serez peut-être tenté d'investir dans des programmes qui vous permettent d'étudier et d'analyser ce qui arrive aux principales monnaies du monde.

De tels programmes peuvent générer trente-sept tableaux comparatifs, calculer des indicateurs sur les forces relatives des monnaies, des taux d'intérêt et des indices d'activité économique. Le coût de ces programmes dépasse 10 000 F. C'est cher, mais négligeable par rapport aux pertes qu'un manque d'informations financières peut entraîner lorsqu'on joue sur les monnaies !

Les programmes de planification d'agenda sont également en pleine expansion. La plupart des ordinateurs comportent une horloge incorporée ; mais il y en a peu qui soient capables de vous interrompre dans l'exécution d'une tâche pour vous rappeler où vous devriez être à ce moment précis.

C'est pourtant exactement ce que fait, entre autres, le planificateur d'emploi du temps résidant en mémoire morte du HP-75C de chez Hewlett-Packard.

Nous verrons prochainement BrainStorm, programme visant à organiser votre pensée de la même manière que les traitements de texte organisent vos phrases.



Profils de marchés
Nous avons montré dans notre diagramme des domaines d'application, avec les besoins des diverses catégories représentatives d'utilisateurs de micros. L'axe vertical représente le niveau de complexité de l'application, depuis les méthodes manuelles (les moins complexes) jusqu'au niveau le plus élevé occupé par les logiciels « sur mesure », ou encore personnalisés. Le profil d'utilisation qui en résulte montre clairement que les logiciels d'ordre général satisfont les besoins en matière de gestion, mais que les utilisateurs spécialisés doivent personnaliser les logiciels existants ou créer les leurs. Les courbes irrégulières haut-bas indiquent une mauvaise correspondance entre les besoins et les ressources ; les courbes régulières et à mi-hauteur signifient que le marché parvient à satisfaire les utilisateurs. (Cl. Kevin Jones.)

Prenez tout

Nous avons d'abord tracé un plan du monde imaginaire de notre jeu d'aventures. Nous y avons ensuite placé les objets. Voici maintenant les routines pour prendre et transporter ces objets.

Précédemment, nous avons passé en revue l'analyse des commandes, et étudié quelques-unes de celles-ci. Il y avait ainsi PRENDRE et LAISSER (avec des variantes comme POSER). Une fois cette commande identifiée par l'ordinateur, le programme doit mettre en œuvre les routines qui lui permettent d'obéir aux injonctions de l'utilisateur. Considérons d'abord PRENDRE.

Pour voir comment elle fonctionne, revenons à la méthode, employée par le programme, pour gérer les objets présents dans notre univers imaginaire.

Dans la première partie, nous avons créé des DATA pour chaque emplacement, avec une description de l'endroit, le nom des objets présents, et la liste des sorties possibles. Après la lecture des DATA, l'ordinateur crée un tableau, IV$, qui, dans la Forêt hantée, a le contenu suivant :

N	IV$(N,1)	IV$(N,2)
1	FUSIL	10
2	LAMPE	9
3	CLÉ	5

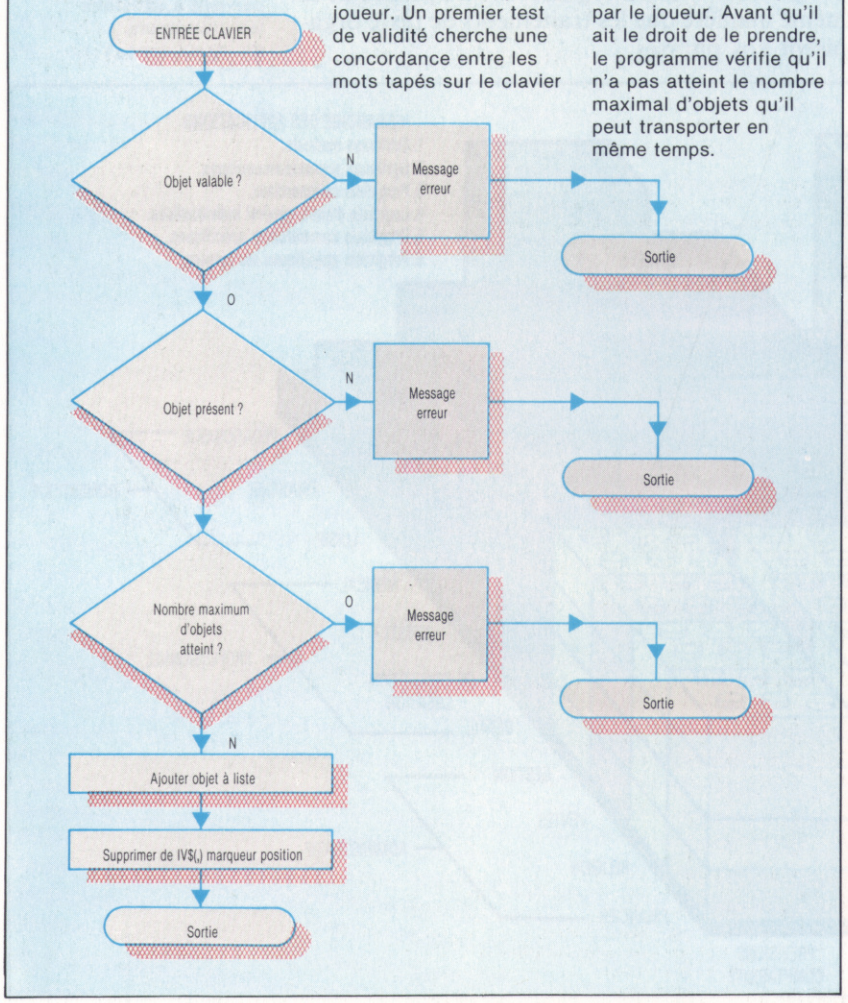
La première colonne (N,1) garde en mémoire le nom de l'objet, et la deuxième sa position en début de partie. A chaque description d'un lieu particulier, cette seconde colonne est interrogée pour voir si l'un des objets se trouve là où est le joueur. Lorsque celui-ci veut s'emparer de quelque chose, au moyen d'une commande de type PRENDRE OBJET, plusieurs facteurs doivent être pris en considération :

- L'objet est-il acceptable? (C'est-à-dire fait-il partie du tableau IV$(,)?)
- Est-il bien là où le joueur se trouve?
- Le joueur a-t-il déjà le nombre maximal d'objets fixé par le programme?

En fonction des réponses à ces questions, le joueur pourra, ou non, prendre l'objet qu'il convoite. La description de cet objet devra alors être ajoutée au tableau des possessions personnelles du joueur, IC$(), tandis que le marqueur de position devra être supprimé de façon correspondante dans IV$(,). Le nom de l'objet lui-même n'a pas, en revanche, à subir un traitement du même ordre; il suffit d'employer un marqueur de position égal à −1 pour chaque objet pris et transporté, pour que cet objet n'apparaisse plus dans la description du lieu. Il serait peu cohérent de prendre le FUSIL à l'emplacement 10, d'aller en 9 et de revenir en 10 pour constater que le FUSIL est toujours là! Le tableau IV$(,) gère donc aussi la position de tous les objets que le joueur ne transporte pas. L'organigramme de la routine PRENDRE montre quelles sont les conditions logiques à respecter.

Un test de validité

L'organigramme de la routine PRENDRE montre quels tests la routine effectue sur la commande entrée par le joueur. Un premier test de validité cherche une concordance entre les mots tapés sur le clavier et les objets de l'inventaire. Un second vérifie que l'objet en question est bien à l'emplacement occupé par le joueur. Avant qu'il ait le droit de le prendre, le programme vérifie qu'il n'a pas atteint le nombre maximal d'objets qu'il peut transporter en même temps.



Gary Capps-Jenner

```
3700 REM **** S/B PRENDRE ****
3710 GOSUB 5300:REM L'OBJET EST-IL ACCEPTABLE
3720 IF F=0 THEN SN$="IL N'Y A PAS DE "+W$:GOSUB5500:
     RETURN
3730 OV=F:GOSUB5450:REM ANALYSE INVENTAIRE
3740 IF HF=1 THEN SN$="VOUS L'AVEZ DEJA"
     :GOSUB5500:RETURN
3750 :
3755 REM ** L'OBJET EST-IL LA ? **
3760 IF VAL(IV$(F,2))<>P THEN SN$="JE N'EN VOIS PAS"
3770 :
3780 REM ** AJOUTER OBJET A LA LISTE **
3790 A=0
3800 FOR J=1 TO 2
3810 IF IC$(J)="" THEN IC$(J)=IV$(F,1):AF=1:J=2
3820 NEXT J
```

```
3830 :
3840 REM ** MAXIMUM **
3850 IF AF=0 THEN PRINT "VOUS AVEZ DEJA 2 OBJETS"
     :RETURN
3860 :
3870 SN$= C'EST FAIT:GOSUB5500
3880 IV$(F,2)="-1":REM DELETE INVENTORY ENTRY
3890 RETURN
```

Un test de validité

Le *test de validité* est le plus important et le plus complexe des trois. Réduit à sa plus simple expression, ce n'est qu'une simple routine qui prend en charge la seconde partie de la réponse donnée par le joueur, et la compare avec le tableau IV$(,).

Mais, sous cette forme, la commande PRENDRE ne pourrait avoir qu'une forme unique du type PRENDRE OBJET. En effet, une phrase comme PRENDRE LE FUSIL ne serait pas acceptée, puisque la routine comparerait LE FUSIL avec l'inventaire, qui ne comprend que FUSIL. Il nous faut donc mettre au point une méthode d'analyse un peu plus sophistiquée.

Le procédé le plus évident consiste à subdiviser en constituants de base la seconde partie de la réponse, et à comparer chacun d'eux avec l'inventaire. Le problème est ainsi contourné, mais de nouvelles difficultés surgissent. Si l'inventaire contient une description du type GRAND COUTEAU, une commande du genre PRENDRE LE GRAND COUTEAU ne sera pas acceptée. En effet, notre routine comparerait successivement LE, GRAND et COUTEAU, de façon séparée. Mais il y a une solution, plus élaborée encore : au lieu de chercher une concordance parfaite entre la phrase tapée par le joueur et les descriptions de l'inventaire, la routine examinera le nom de l'objet (dans le tableau) lettre par lettre, jusqu'à ce qu'elle parvienne à un accord, ou qu'elle atteigne la fin du mot. L'encadré ci-contre explique la méthode de façon plus approfondie.

Mots abrégés

Un des avantages est qu'il devient possible d'entrer tel ou tel mot sous forme abrégée. C'est ainsi que PRENDRE LE COU serait tout à fait accepté, à condition bien sûr qu'aucun objet précédant COUTEAU dans l'inventaire ne commence par COU ; si cela arrivait, l'ordinateur donnerait une réponse inexacte. Des problèmes de ce genre sont inévitables, dès lors qu'on veut se servir d'une routine « souple ». On peut les éliminer en faisant très attention aux noms d'objets insérés dans l'inventaire.

Si deux d'entre eux commencent par les mêmes lettres, ou si l'un est contenu dans l'autre, le plus

```
5300 REM **** VALIDATION OBJET ****
5310 NN$=NN$+" ":LN=LEN(NN$):C=1:F=0
5315 FOR K=1 TO LN
5320 IF MID$(NN$,K,1)<>" " THEN NET K:RETURN
5325 W$=MID$(NN$,C,K-C):C=K+1
5330 LW=LEN(W$)
5335 FOR J=1 TO 3
5340 LI=LEN(IV$(J,1)):REM LONGUEUR OBJET
5350 FOR I=1 TO LI-LW+1
5360 IF MID$(IV$(J,1),I,LW)=W$ THEN F=J:I=LI:J=3:K
=LN
5370 NEXT I,J,K
5380 RETURN
```

Concordance


```
  10REM **** DEMO CONCORDANCE BBC ****
  40MODE 5:COLOUR 2:DIM V$(3)
  60FOR I=1 TO 3:READ V$(I):NEXT I
  90A$="LE" :B$="COUTEAU":C$="COU"
  95$="                 "
 110M$=A$:GOSUB 1000:REM CONCORDANCE LE
 130M$=B$:GOSUB 1000:REM CONCORDANCE COUTEAU
 150M$=C$:GOSUB 1000:REM CONCORDANCE COU
 160END
1000REM **** S/P CONCORDANCE ****
1010CLS:F=0:LW=LEN(M$)
1030FOR J=1 TO 3:LI=LEN(V$(J))
1042X=1:Y=6:GOSUB2000:PRINT S$
1045X=1:Y=6:GOSUB2000:PRINT V$(J)
1047FOR I=1 TO LI-LW+1
1050X=1:Y=5:GOSUB2000:PRINT S$
1060X=1:Y=5:GOSUB2000:PRINT M$
1070IF MID$(V$(J),I,LW)=M$ THEN F=I:I=LI:J=3
1075FOR D=1 TO 300:NEXT D,I:REM RETARD
1086IF F<>0 THEN GOSUB 2500
1087NEXT J:RETURN
2000REM **** POSITION CURSEUR EN X,Y ****
2010PRINT TAB(X,Y);:RETURN
2500REM **** CONCORDANCE POSITIVE ****
2502COLOUR 1:X=F:Y=6:GOSUB2000:PRINT M$
2505FOR K=1 TO 5:X=1:Y=10:GOSUB2000:PRINT S$
2508FOR D=1 TO 500:NEXT:REM RETARD
2510X=1:Y=10:GOSUB2000:PRINT "CONCORDANCE POSITIVE"
2512FOR D=1 TO 500:NEXT D,K:REM RETARD
2520A$=GET$:COLOUR 2:RETURN
3000REM **** DATA INVENTAIRE$ ****
3005DATA"PETITE FOURCHETTE","PORTE ROUGE","GRAND COUTEAU"
```

```
  10 REM ** SPECTRUM **
  40 INK 6:DIM V$(3,20)
1070 IF V$(J,I TO I+LW-1)=M$ THEN F=I:I=LI:J=3
2010 PRINT AT (Y,X);:RETURN
2502 INK 2:X=F:Y=6:GOSUB2000:PRINT S$
2520 A$=INKEY$:IF A$="" THEN 2520
2525 INK 6:RETURN
```

```
  10 REM ** CBM 64 **
  40 PRINT CHR$(158):DIM V$(3)
  50 DN$=CHR$(17): FOR K=1 TO 5:DN$=DN$+DN$:NEXT:
     DN$=CHR$(19)+DN$
2010 PRINT LEFT$(DN$,Y)TAB(X);:RETURN
2502 PRINTCHR$(28);:X=F:Y=6:GOSUB2000:PRINT S$
2520 GETA$:IF A$="" THEN 2520
2525 PRINT CHR$(158):RETURN
```

La sous-routine de validité employée dans le cadre de la routine PRENDRE vérifie la phrase entrée au clavier, à raison d'un mot à la fois, en s'efforçant de trouver une concordance avec un des éléments de l'inventaire. Ce court programme, purement démonstratif, montre comment on procède. Son inventaire ne comporte que trois objets, et il entreprend de chercher des correspondants à LE, COUTEAU et COU. Chaque fois qu'il trouve une concordance, il attend que le joueur appuie sur une touche pour continuer.

court des deux doit être placé avant dans l'inventaire. De plus, il ne faudra pas faire usage de descriptions différentes comportant le même mot : on n'emploiera pas à la fois COUTEAU et GRAND COUTEAU.

Lorsqu'elle a détecté une concordance, la routine affecte à la variable F une valeur qui est celle de l'élément de l'inventaire correspondant à l'objet spécifié dans la commande. Dans le cas contraire, F garde une valeur de zéro, indiquant que l'objet en question n'existe pas.

L'objet est-il présent?

Une fois la routine de validation menée à bien, l'emplacement de l'objet peut être sans difficulté établi grâce à la valeur de la variable P (position où se trouve le joueur). L'objet à prendre est en IV$(F,1), et sa position est en IV$(F,2). La ligne 3760 de la routine PRENDRE de la Forêt hantée compare cette valeur à celle de P.

Toutefois, cela peut donner lieu à des erreurs subtiles, le programme annonçant que l'objet n'est pas là, alors que le joueur le tient ! Il faut donc savoir ce qu'il possède déjà, avant d'émettre un message d'erreur (du type « VOUS AVEZ DÉJÀ l'OBJET »). Cette condition est vérifiée par une analyse des éléments du tableau ; la sous-routine qui suit en vérifie les éléments et donne à un drapeau la valeur 1 si l'objet recherché est déjà en possession de l'objet recherché, ce qu'indique un −1 dans la partie correspondante du tableau.

```
5450 REM ***** L'OBJET EST-IL EN POSSESSION
DU JOUEUR *****
5460 HF=0
5470 IF IV$(OV,2)="-1" THEN HF=1
5480 RETURN
```

Le joueur et ses bagages

Il est possible de combiner simultanément deux tâches différentes : ajouter des objets aux possessions du joueur, et vérifier s'il a ou non atteint le nombre maximal d'objets. Le tableau IC$() contiendra la liste de ce qu'il possède déjà, et une boucle FOR...NEXT servira à repérer le premier espace libre dans ce tableau, de façon à y insérer le nouvel objet.

Dans la Forêt hantée, le joueur ne peut en transporter que deux à la fois, et la boucle ne sera donc exécutée que deux fois. S'il reste un espace, il est possible de prendre l'objet ; sinon un message d'erreur est affiché.

Enfin, nous devrons supprimer le marqueur de position de l'objet que nous venons de retirer de l'inventaire. Il suffit pour cela de donner à IV$(F,2) la valeur −1.

Un aide-mémoire utile

Le joueur aura donc la possibilité de prendre les objets qu'il rencontre. Il est intéressant d'inclure dans le jeu une commande lui permettant d'obtenir la liste de ses possessions, car il peut très bien se heurter à une porte fermée sans se souvenir qu'un quart d'heure auparavant il a ramassé une clé. Une sous-routine très simple lui rafraîchira la mémoire en lui permettant de lister tous les objets déjà rencontrés. Une boucle FOR...NEXT affichera le contenu du tableau gérant les objets possédés par le joueur, IC$().

```
4100 REM ***** LISTE DES OBJETS POSSEDES *****
4110 PRINT "VOUS AVEZ EN VOTRE POSSESSION:"
4120 FOR I=1 TO 2
4130 PRINT "  "; IC$(I)
4140 NEXT I
4150 RETURN
```

Listages Digitaya

```
2140 REM **** S/P PRENDRE ****
2145 IV$(4,1)="TICKET TO TRI-STATE"
2150 GOSUB5730:REM IS OBJECT VALID
2160 IF F=0 THEN PRINT"JE N'EN VOIS PAS":RETURN
2170 REM ** "OBJET DEJA PRIS ? **
2180 OV=F:GOSUB5830
2190 IFHF=1 THEN SN$="VOUS L'AVEZ DEJA"
     GOSUB5880:RETURN
2200 :
2210 REM ** OBJET PRESENT **
2220 IF VAL(IV$(F,2))<>P THEN SN$="IL N'Y EN A PAS ICI !"
     :GOSUB5880:RETURN
2230 :
2240 REM ** AJOUTER OBJET A LISTE **
2250 AF=0:FOR J=1TO4
2260 IFIC$(J)=""THENIC$(J)=IV$(F,1):AF=I:J=4
2270 NEXTJ
2280 :
2290 REM ** NOMBRE MAXIMUM **
2300 IF AF=0THENPRINT"VOUS AVEZ DEJA 4 OBJETS" :
RETURN
2310 :
2320 SN$="C'EST FAIT":GOSUB5880
2330 IV$(F,2)="-1":REM DELETE POSITION ENTRY
2340 RETURN


5730 REM **** S/P VALIDATION OBJET ****
5740 NN$=NN$+" ":LN=LEN(NN$):F=0:C=1
5745 FOR K=1 TO LN
5750 IF MID$(NN$,K,1)<>" " THEN NEXTK:RETURN
5755 W$=MID$(NN$,C,K-C):C=K+1:LW=LEN(W$)
5760 FORJ=1 TO 8
5770 LI=LEN(IV$(J,1)):REM LONGUEUR OBJET
5780 FORI=1TO LI-LW+1
5790 IFMID$(IV$(J,1),I,LW)=W$THENF=J:I=LI:J=8:K=LN
5800 NEXT I,J,K
5810 RETURN
5820 :
5830 REM **** S/P OBJET TRANSPORTE ? ****
5840 HF=0
5850 IFIV$(OV,2)="-1"THEN HF=1
5860 RETURN

2540 REM **** LIST INVENTORY S/R ****
2550 PRINT"VOUS AVEZ:"
2560 FORI=1TO4
2570 PRINT"  ";IC$(I)
2580 NEXTI
2590 RETURN
```

Variantes de basic

Spectrum :

Pour la Forêt hantée procédez aux changements suivants :

Remplacer SN$ par S$, IV$(,) par IC$() par I$() et NN$ V$(,) par R$.

5320 IF R$(K TO K) <> « » THEN NEXT K:RETURN
5325 LET W$ = R$(C TO K − 1)
5360 IF V$(I TO I + LW − 1) = W$ THEN LET F = J: LET I = LI:LET J = 3:LET K = LN

Pour Digitaya, les changements sont les mêmes, sauf pour les lignes 5750, 5755 et 5790.

Osborne est de retour

La compagnie américaine Osborne Corp. fut la première à lancer un micro vraiment portable, l'Osborne-1. Après bien des ennuis, elle sort l'Osborne Encore, lui aussi compatible avec l'IBM PC.

Le lancement de l'Osborne-1 constitua une véritable révolution : ce portable, le premier du genre, comportait un moniteur intégré, un double lecteur de disquettes, des interfaces modem et imprimante, et disposait du fameux système d'exploitation CP/M. Il pesait 10,500 kg, ce qui était peut-être un peu excessif pour un « portable », mais il avait un potentiel énorme que la concurrence comprit aussitôt. Adam Osborne, son créateur, qui avait littéralement inventé un nouveau marché, donna bien des idées aux autres firmes au point de se retrouver face à de nombreux concurrents.

Le coup de grâce vint en 1981, lorsque IBM annonça le lancement de son premier ordinateur personnel. Tous les hommes d'affaires des États-Unis se ruèrent instantanément sur le nouvel appareil. Osborne annonça très vite la mise en vente d'un compatible, l'Osborne Executive. Il devait être équipé de deux microprocesseurs, ce qui lui aurait permis de faire tourner à la fois les logiciels sous CP/M et ceux sous MS/DOS. Mais les puces 8086 étaient alors presque introuvables dans le monde entier; l'Executive fit donc son apparition sans être aucunement compatible. Les ventes furent si mauvaises qu'Osborne Corp dut se résoudre à une liquidation volontaire au cours de l'été 1983.

Osborne remet ça

Pourtant Adam Osborne parvint à remonter la pente, et il a aujourd'hui mis sur pied une nouvelle compagnie de recherche et de développement, un peu analogue à Sinclair Research, en Grande-Bretagne.

L'Osborne Encore est enfin arrivé sur le marché malgré toutes ces difficultés. Il est le premier-né de la nouvelle firme. C'est, de la part de celle-ci, un pari audacieux. Ce n'est pas un ordinateur aussi compact que l'Epson PX-8, mais il a pour but de combler une lacune dans la gamme des machines consacrées à la gestion, et se situe entre les portables et les appareils de bureau.

Il pèse environ 6 kg, ce qui est un gros progrès par rapport à son prédécesseur. Le clavier se replie sur l'écran, et le tout a à peu près les dimensions de trois annuaires téléphoniques empilés. Le boîtier est moulé dans un plastique bleuté très solide et le clavier est retenu par une double attache.

Le clavier est de type machine à écrire. Il comporte dans sa partie supérieure une série de touches de fonction à membrane, ainsi que des « icônes » (signes conventionnels représentant les logiciels intégrés dans l'appareil). L'écran, de 23,7 × 8 cm, est à cristaux liquides.

C'est un vrai clavier professionnel, pourvu de touches de contrôle situées des deux côtés, et de touches curseur (en bas et à droite). Peut-être est-il malgré tout un peu trop ramassé, mais c'est là une conséquence inévitable, dès lors que le constructeur s'efforce de doter son modèle de toutes les caractéristiques de l'IBM PC.

Choisir l'icône

Ainsi, ce dernier est doté d'un clavier numérique séparé qui, sur l'Encore, a été placé au sein du clavier lui-même, les touches correspondantes étant bleues (et non blanches). Pour accéder aux fonctions de calcul, il suffit simplement d'appuyer sur l'icône qui leur est réservée (les autres étant affectées au modem, au lecteur de disquettes et au calendrier).

Cet ensemble de touches prête le flanc à la critique, en particulier de la part des professionnels. Les touches de fonction qui, sur l'IBM PC, sont séparées des autres sur la gauche du clavier, évoquent un peu le petit ZX81... On a, en les pressant, le sentiment ambigu, assez désagréable, de n'être jamais tout à fait sûr d'avoir bien appuyé. La mise en place d'un écran à cristaux liquides

Un appareil compact
L'Osborne Encore est le premier compatible IBM équipé d'un écran à cristaux liquides (et non d'un tube cathodique), sur lequel le clavier vient se replier lorsque l'appareil n'est pas en service. L'ensemble est très compact et peut être transporté grâce à la bretelle qu'on aperçoit au sommet de l'ordinateur. (Cl. Chris Stevens.)

permettait de grosses économies d'énergie, puisqu'un dispositif de ce genre consomme beaucoup moins qu'un tube cathodique. Mais cela crée de gros problèmes de compatibilité avec l'IBM PC. On peut même dire que c'est là que se situe le principal handicap.

Ce n'est pas seulement une question de manque de couleurs, auquel on peut toujours remédier par l'emploi de différents niveaux de luminosité ; c'est surtout la taille de l'écran qui devient le principal handicap.

L'écran de l'IBM PC comporte 25 lignes de 80 caractères. Les concepteurs japonais de l'écran à cristaux liquides de l'Encore n'ont pu reproduire un affichage du même type, et il est, sur l'Osborne, réduit à 16 lignes, ce qui veut dire que bien des logiciels prévus pour le PC seront inutilisables. En effet, beaucoup d'entre eux fonctionnent par « pages », en bas desquelles les messages sont affichés. L'utilisateur, s'il veut les faire tourner, devra donc faire face à bien des difficultés.

Sur le côté droit de l'ordinateur, un espace est ménagé pour deux lecteurs de disquettes 5 pouces (mais un seul est fourni dans la version de base). De l'autre côté, on remarque une prise pour le transformateur, un commutateur de mise en route, un bouton pour régler le contraste de l'écran, et un boîtier pour les piles (de type nickel cadmium), ce qui permet à l'ordinateur de se passer de prise de courant.

Mise en route

Les ports d'entrée/sortie sont placés à l'arrière de l'appareil. On distingue (de gauche à droite) un jack téléphonique qui se raccorde au modem intégré de l'Encore, un port Centronics qui sert d'interface imprimante, et un port série RS232 (permettant l'emploi d'imprimantes et de modems).

Pour charger le système d'exploitation MS-DOS, il suffit d'appuyer sur l'icône « disque » du clavier. Tous les programmes intégrés peuvent être lancés à tout moment, quelle que soit la tâche accomplie alors par l'ordinateur.

Toutes les disquettes IBM chargées en mémoire peuvent être lues par l'Encore. Mais les problèmes d'écran empêchent souvent de vérifier si les programmes tournent bien, car il faut entrer beaucoup de commandes à l'aveuglette.

Toutefois, il faut noter que le 1-2-3 de Lotus ne pose aucune difficulté particulière, alors que, d'habitude, il donne du fil à retordre aux micro-ordinateurs compatibles IBM en raison de sa façon très particulière d'accéder aux routines ROM de l'IBM PC.

On ne peut encore dire si l'Encore connaîtra le même succès que l'Osborne-1. Comme tous les affichages à cristaux liquides, l'écran a besoin d'une source de lumière très forte pour que les caractères soient vraiment lisibles. Ils sont d'ailleurs de taille restreinte, et on peut se demander si les utilisateurs potentiels (hommes d'affaires, gestionnaires) feront l'effort d'adaptation nécessaire.

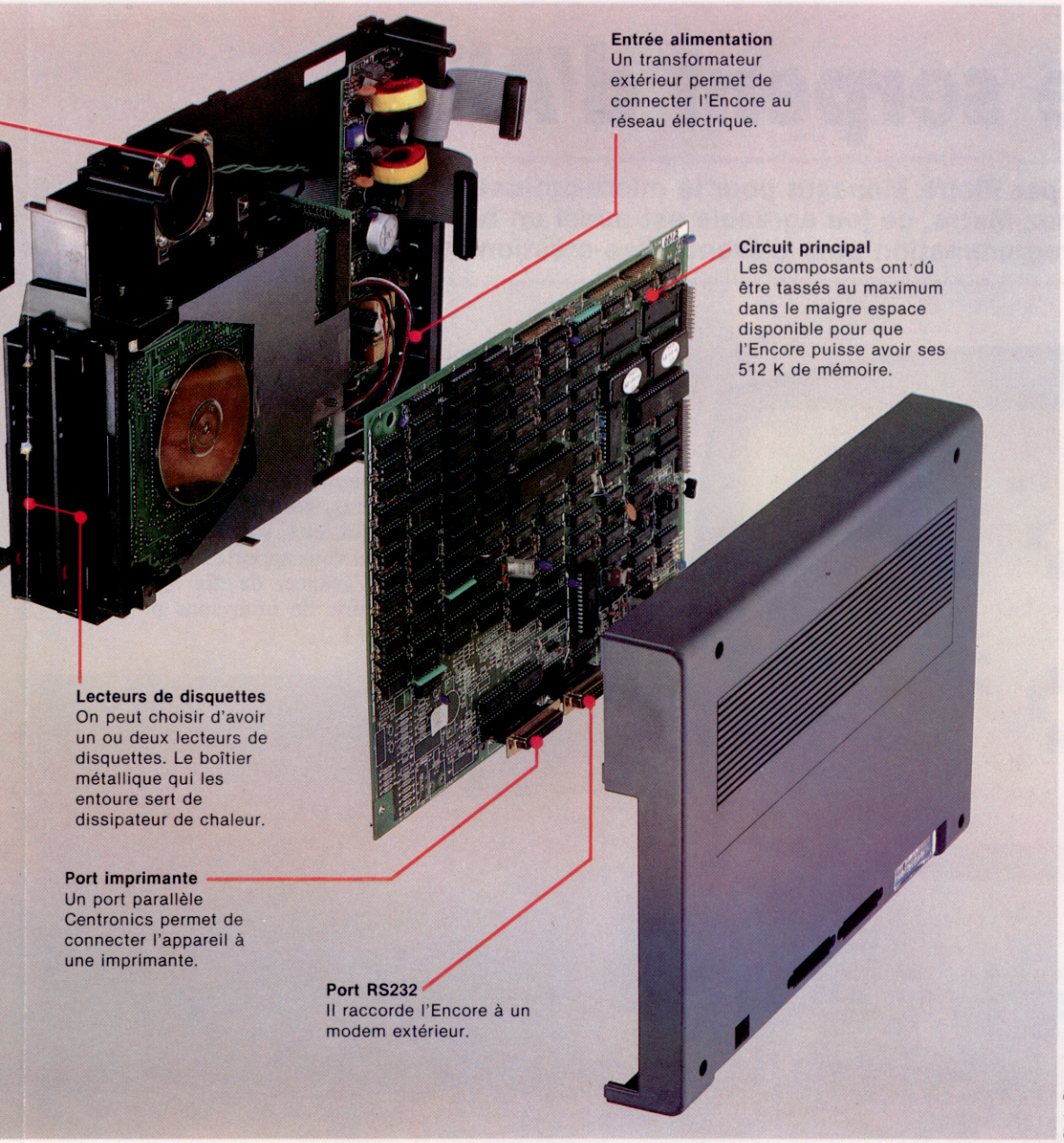
Haut-parleur
Les annonces écran sont accompagnées d'un « bip » audible.

Écran à cristaux liquides
Il consomme beaucoup moins d'énergie qu'un tube cathodique, ce qui permet d'obtenir un compatible IBM fonctionnant sur piles.

Double emploi
Le clavier de l'Encore a moins de touches que celui du PC, mais les possibilités sont les mêmes. Certaines — indiquées en bleu — font donc aussi fonction de touches calculatrices.

Problème de disquettes

Les lecteurs de disquettes — on peut en installer un ou deux — sont placés sur le côté de l'appareil. La compacité de celui-ci s'en trouve accrue, mais l'utilisateur devra parfois insister pour insérer une disquette, et surtout pour s'assurer qu'il l'a bien mise dans le bon lecteur !



Entrée alimentation
Un transformateur extérieur permet de connecter l'Encore au réseau électrique.

Circuit principal
Les composants ont dû être tassés au maximum dans le maigre espace disponible pour que l'Encore puisse avoir ses 512 K de mémoire.

Lecteurs de disquettes
On peut choisir d'avoir un ou deux lecteurs de disquettes. Le boîtier métallique qui les entoure sert de dissipateur de chaleur.

Port imprimante
Un port parallèle Centronics permet de connecter l'appareil à une imprimante.

Port RS232
Il raccorde l'Encore à un modem extérieur.

Chris Stevens

OSBORNE ENCORE

PRIX

Varie selon les 4 modèles.
★ ★ ★ ★ ou
★ ★ ★ ★ ★

DIMENSIONS

241 × 325 × 141 mm.

UC

Microprocesseur 16 bits 8086.

MÉMOIRE

Maximum de 512 K.

ÉCRAN

A cristaux liquides. 16 lignes de 80 caractères. La firme a promis pour bientôt un écran de 25 lignes. Le changement devrait être peu coûteux.

INTERFACES

Interface série RS232, jack téléphone RJ11, port parallèle Centronics.

LANGAGES

BASIC Microsoft fourni sur disquette.

CLAVIER

62 touches de type machine à écrire, plus 10 touches de fonction et 4 icônes.

DOCUMENTATION

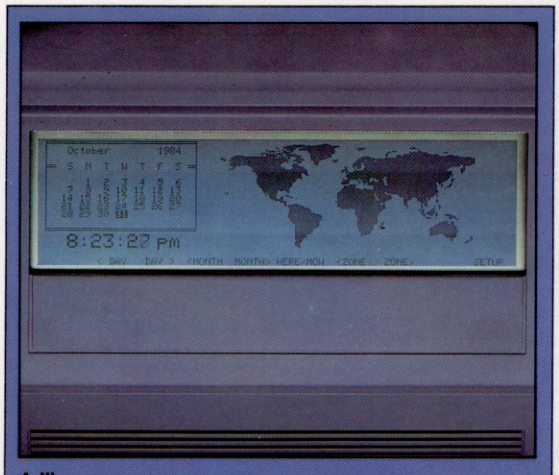
Comme tous les manuels de chez Osborne, les deux guides utilisateur sont très bien écrits, très complets, et contiennent de très nombreux exemples.

FORCES

L'Encore est un appareil compact et puissant, qui peut fonctionner sur piles pendant quatre ou cinq heures.

FAIBLESSES

L'écran n'est pas réellement compatible avec IBM, et sa lecture est parfois difficile.

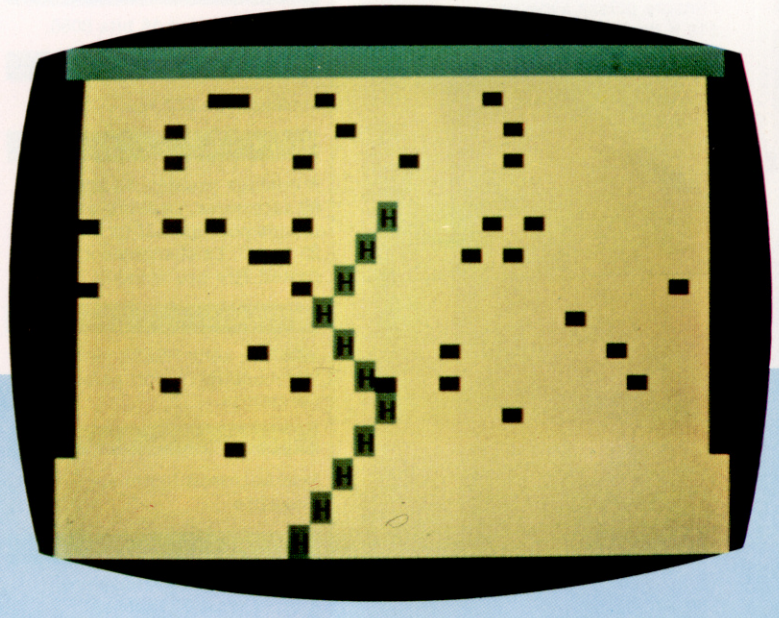


A l'heure
On voit ici l'écran à cristaux liquides de 16 lignes de 80 caractères. Les rainures que l'on remarque en dessous doivent permettre l'installation future d'un écran comportant 25 lignes. L'image représentée est l'affichage agenda/calendrier.

Du bout des doigts
Un clavier à membrane est installé au-dessus du clavier proprement dit. Il comporte dix touches de fonction et des icônes qui permettent l'appel des programmes intégrés implantés en ROM.

Le serpent d'Alice

Écrit par Pierre Monsaut pour le micro-ordinateur Alice de chez Matra, ce jeu agréable est aussi un bon exercice de programmation pour d'éventuelles créations de jeux.



Dans ce jeu, vous êtes un serpent qui se déplace en ondulant sur l'écran. Le changement de direction s'effectue en tapant n'importe quelle touche. Pour pouvoir vous déplacer, vous devez vous nourrir. Heureusement, vous êtes entouré par un grand nombre de champignons. Mais attention ! Si les bleus sont excellents, vous devez absolument éviter les rouges qui, eux, sont vénéneux. Chaque champignon bleu vous apporte suffisamment de calories pour avancer de dix lignes. Essayez de ne pas mourir de faim sans pour autant finir empoisonné !

```
10 REM SERPENT
15 REM T=POSITION INITIALE
16 REM       DU SERPENT
20 T=239
25 REM H=SCORE
30 H=0
35 REM R$=CARRE ROUGE
40 R$=CHR$(191)
45 REM B$=CARRE BLEU
50 B$=CHR$(175)
55 REM T$=CARACTERE SERPENT
60 T$="S"
65 REM S=CALORIES
70 S=100
80 CLS
85 REM D=DIRECTION (1 OU -1)
90 D=1
95 REM
96 REM BOUCLE PRINCIPALE
97 REM
99 REM CHANGEMENT DIRECTION
100 D$=INKEY$
110 IF D$<>"" THEN D=-D
120 T=T+D
125 REM POSITION VALIDE?
130 IF T<225 THEN T=225
140 IF T>254 THEN T=254
150 T1=T+32
155 REM CALCUL COORDONNEES
160 Y=INT(T1/32)*2
170 X=(T1-16*Y)*2
175 REM CHAMPIGNON ROUGE?
180 IF POINT(X,Y)=4 THEN 300
185 REM CHAMPIGNON BLEU?
190 IF POINT(X,Y)=3 THEN S=S+10:
H=H+10:SOUND 1,1
200 PRINT@ 511," ";
205 REM AFFICHAGE CHAMPIGNONS
210 PRINT@ 480+RND(30),B$;
220 IF RND(2)=1 THEN PRINT@ 480+
RND(30),R$;
225 REM AFFICHAGE SERPENT
230 PRINT@ T,T$;
235 REM DECOMPTE CALORIES
240 S=S-1
245 REM MORT DE FAIM?
250 IF S=0 THEN 300
255 REM SCORE: +1
260 H=H+1
270 GOTO 100
295 REM
296 REM FIN
297 REM
300 SOUND 1,1
310 PRINT@ 511," ";
320 PRINT@ T,T$;
330 FOR I=1 TO 5
340 SOUND 1,1
350 FOR J=1 TO 50
360 D$=INKEY$:NEXT J
370 NEXT I
375 REM RECORD BATTU?
380 IF H>R THEN R=H
390 PRINT@ 66,"SCORE :";H,"RECOR
D :";R;
400 PRINT@ 138,"UNE AUTRE ?";
410 D$=INKEY$
420 IF D$="" THEN 410
430 IF D$<>"N" THEN 20
440 CLS
450 END
```

Bis repetita

Dans ce dernier article sur la programmation LOGO, nous montrons comment ajouter de nouvelles structures de commande au langage et des procédures capables d'écrire elles-mêmes d'autres procédures.

La primitive EXÉCUTE prend une liste en entrée et l'exécute comme s'il s'agissait d'une ligne de procédure. Cela peut servir à ajouter de nouvelles structures de commande au langage selon la manière et le moment voulus. Nous pouvons définir ainsi une procédure PENDANTQUE :

```
POUR PENDANTQUE :CONDITION :ACTION
  SI NON (EXÉCUTE :CONDITION) ALORS STOP
  EXÉCUTE :ACTION
  PENDANTQUE :CONDITION :ACTION
FIN
```

Voici un exemple d'utilisation. PUISSANCE affiche toutes les puissances pour les valeurs entrées inférieures à 1 000 :

```
POUR PUISSANCE :X
  FAIS «P :X
  PENDANTQUE [:P < 1000] [AFFICHE :P FAIS «P :P* :X]
FIN
```

PENDANTQUE (WHILE), RÉPÈTE (REPEAT) et POUR (FOR) sont des commandes habituelles dans d'autres langages. Elles ne sont pas vraiment nécessaires avec LOGO. Une manière plus naturelle à LOGO d'écrire PUISSANCE serait :

```
POUR PUISSANCE :P
  SI NON :P < 1000 ALORS STOP
  AFFICHE :P
  PUISSANCE P* :P
FIN
```

RÉPÈTE est présent dans toutes les versions LOGO mais n'est pas strictement nécessaire puisque l'on peut définir un mot équivalent, REPT, de la manière suivante :

```
POUR REPT :NON :LISTE
  SI :NON = 0 ALORS STOP
  EXÉCUTE :LISTE
  REPT :NON – 1 :LISTE
FIN
```

EXÉCUTE est une primitive très utile pour un travail plus avancé avec LOGO. Un programme peut assembler une liste et la transmettre à EXÉCUTE pour qu'elle soit exécutée. Nous l'étudierons prochainement.

Prendre les procédures séparément

Nous devons d'abord définir une procédure pour tracer normalement un triangle :

```
POUR TRI
  AV 50 DR 120 AV 50
  DR 120 AV 50 DR 120
FIN
```

Tapez maintenant AFFICHE TEXTE «TRI. Le résultat sera :

```
[AV 50 DR 120 AV 50 DR 120 AV 50 DR 120]
```

Le texte de la procédure est donné sous la forme d'une liste de listes, dont chaque liste interne est une ligne de procédure. Pour comprendre la présence d'une liste vide au départ, définissez cette procédure de remplacement pour l'addition :

```
POUR AJOUTE :A :B
  AFFICHE :A + :B
FIN
  AFFICHE TEXTE «AJOUTE donnera :
  [:A :B] [AFFICHE :A :B]
```

La première liste contient les entrées de la procédure. TEXTE nous permet donc d'accéder à l'intérieur d'une procédure et de voir ce qu'elle comporte. DÉFINIS, pour sa part, fait le contraire : elle nous permet de définir une procédure comme une liste de listes sans avoir à passer par l'éditeur. Essayons maintenant DÉFINIS «L [[:A] [AV :A] [DR 90] [AV :A/2]] et exécutons L avec par exemple L 30. Utiliser DÉFINIS en mode immédiat de la sorte n'a aucun intérêt par rapport à l'utilisation de l'éditeur. L'avantage de DÉFINIS est qu'une procédure peut créer une autre procédure.

En grandissant

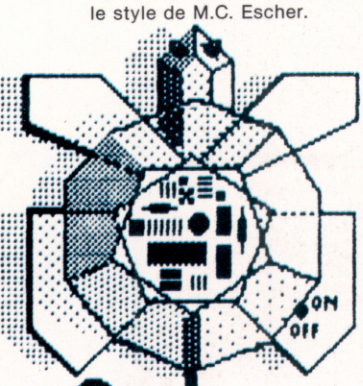
Nous en venons à développer un petit système pour contrôler la taille des formes. Les commandes de base seront DEMANDE, qui détermine la forme concernée, et AGRANDISSEMENT, qui modifie la taille de la forme retenue. Par exemple, DEMANDE «CARRÉ dessinera un carré, et AGRANDISSEMENT [*10] effacera le carré existant pour le retracer avec des côtés 10 fois plus grands.

Afin que le programme reste simple, il nous faut accepter quelques restrictions pour ces commandes. D'abord, les procédures de formes données en entrée à DEMANDE ne doivent pas comporter RÉPÈTE ni appeler de sous-procédures. Ensuite, le système s'effondrera si vous obtenez des résultats négatifs. Ces problèmes ne sont pas insolubles pour autant que l'on cherche à améliorer ce que nous proposons ici.

DEMANDE fonctionne en assignant le nom de la forme à la variable globale «COURANT, puis en exécutant la procédure. Elle crée à cette fin une liste d'un article — le nom de la procédure — et utilise ensuite EXÉCUTE.

```
POUR DEMANDE :OBJET
  CACHETORTUE
  FAIS «COURANT :OBJET
  EXÉCUTE (LISTE :OBJET)
FIN
```

Dessine-moi une tortue
On ne peut avancer dans l'étude de LOGO sans rencontrer le thème de la récursivité. Nous avons vu des exemples tels que des procédures qui s'appellent elles-mêmes, des listes définies en termes de listes. Nous voyons maintenant des procédures qui écrivent d'autres procédures. Avec un peu d'imagination, il serait facile avec LOGO de créer des dessins dans le style de M.C. Escher.



AGRANDISSEMENT efface d'abord le dessin existant (LOGO sur Commodore utilise COULEURCRAYON-1 pour l'effacement), puis utilise DÉFINIS pour définir la procédure courante comme celle réécrite. La couleur du crayon retourne à sa valeur normale et la nouvelle forme est dessinée. Remarquez que les entrées à AGRANDISSEMENT sont sauvegardées dans la variable OUVLISTE, que nous devrons utiliser par la suite.

```
POUR AGRANDISSEMENT :OUVLISTE
   COULEURCRAYON – 1
   EXÉCUTE (LISTE :COURANT)
   DÉFINIS :COURANT RÉÉCRIRE.PROC TEXTE :COURANT
   COULEURCRAYON 1
   EXÉCUTE (LISTE :COURANT)
FIN
```

RÉÉCRIRE.PROC divise le texte en lignes et les transmet une par une à RÉÉCRIRE.LIGNE :

```
POUR RÉÉCRIRE.PROC :TEXTE
   SI VIDE? :TEXTE ALORS RÉSULTAT []
   RÉSULTAT FMETS RÉÉCRIRE.LIGNE PREMIER :TEXTE
      RÉÉCRIRE.PROC SAUFPREMIER :TEXTE
   FIN
```

RÉÉCRIRE.LIGNE cherche une commande AV ou AVANCE sur la ligne. Si elle en trouve une, elle transmet le reste de la ligne à MODIFIE qui s'en chargera.

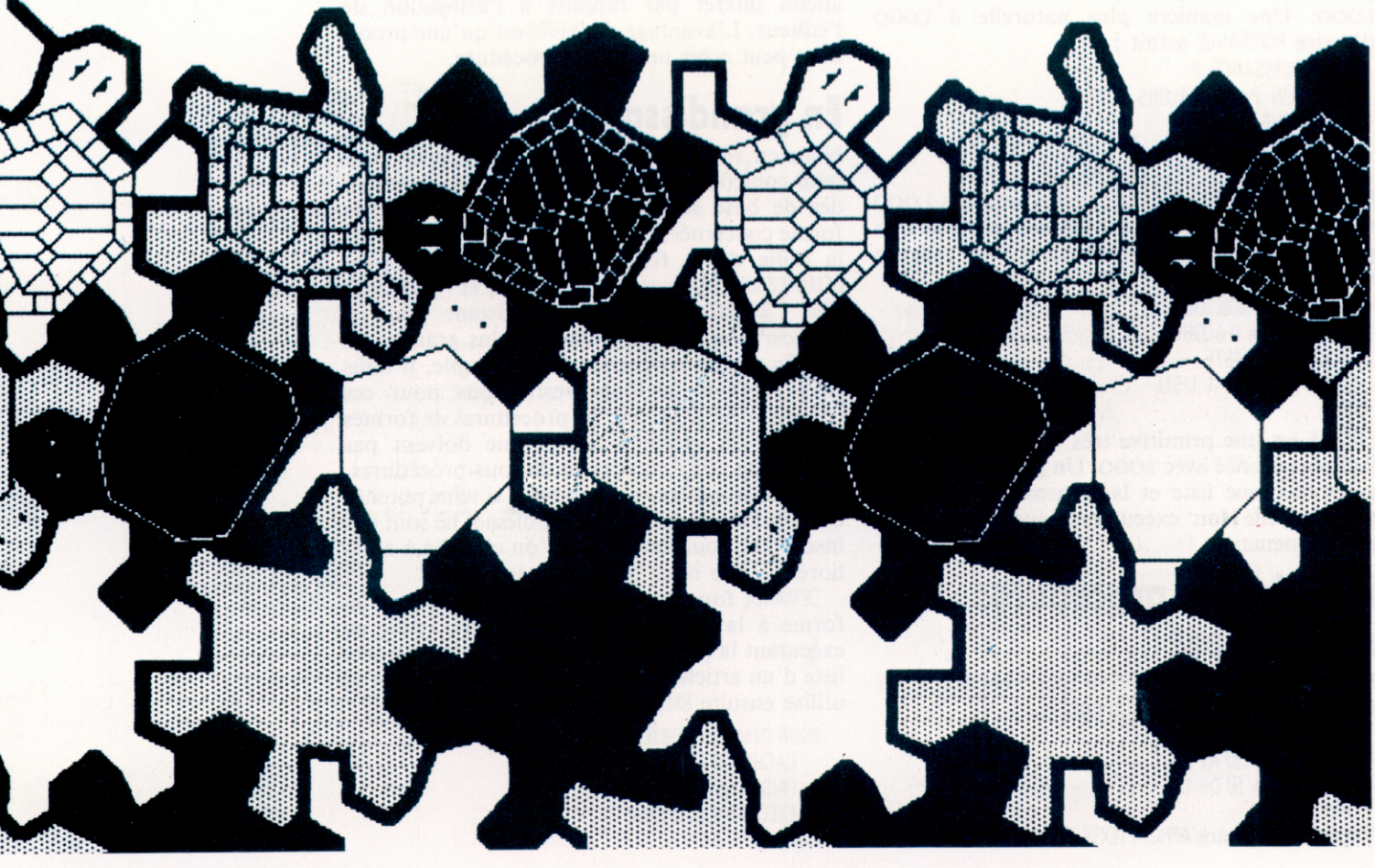
```
POUR RÉÉCRIRE.LIGNE :LIGNE
   SI VIDE? :LIGNE ALORS RÉSULTAT []
   SI AU MOINS PREMIER :LIGNE = «AV
      PREMIER :LIGNE = «AVANCE ALORS RÉSULTAT
      MODIFIE SAUFPREMIER :LIGNE
   RÉSULTAT FMETS PREMIER :LIGNE
      RÉÉCRIRE.LIGNE SAUFPREMIER :LIGNE
FIN
```

MODIFIE élabore la ligne réécrite. Le premier article de LISTE, en entrée à MODIFIE, aurait été l'entrée à AVANCE dans la procédure initiale. Supposons qu'il s'agisse de 50, si OUVLISTE contient [*2], PHRASE PREMIER :LISTE :OUVLISTE correspondrait à [50*2]. MODIFIE utilise maintenant EXÉCUTE pour évaluer cette liste (obtenant un résultat de 100). Enfin, une liste est constituée, composée de AV, de la nouvelle valeur calculée, et de la réécriture du reste de la ligne.

```
POUR MODIFIE :LISTE
   RÉSULTAT (PHRASE « AV (EXÉCUTE PHRASE
      PREMIER :LISTE :OUVLISTE)
      RÉÉCRIRE.LIGNE SAUFPREMIER : LISTE)
FIN
```

Copieuse...

Il est souvent utile de pouvoir faire une copie d'une procédure. Aussi, définissons une procédure COPIEDEF, de telle sorte que COPIEDEF«NOUVEAU NOM «ANCIENNOM définisse NOUVEAUNOM comme une



Ian McKinnell

copie de ANCIENNOM (ANCIENNOM n'étant pas modifié). Une définition évidente serait :

```
POUR COPIEDEF :NOUVEAU :ANCIEN
    DÉFINI :NOUVEAU TEXTE :ANCIEN
FIN
```

L'ennui avec cette définition, c'est que si ANCIEN n'existe pas, la procédure passe à NOUVEAU, et la définit comme ne contenant rien. Il convient donc de résoudre ce problème. La définition suivante sera plus efficace :

```
POUR COPIEDEF :NOUVEAU :ANCIEN
    SI NON PROCÉDURE? :ANCIEN ALORS
      (AFFICHE [PAS DE PROCÉDURE]
      :ANCIEN) STOP DÉFINIS :NOUVEAU
      TEXTE :ANCIEN
FIN
```

Cette définition utilise une procédure appelée PROCÉDURE? qui donne en résultat VRAI lorsque son entrée est une procédure, et FAUX dans le cas contraire. PROCÉDURE? et sa contrepartie PRIMITIVE? sont des tests très utiles qui n'existent pas dans LOGO MIT. Aussi nous développerons ici des versions de PROCÉDURE? et de PRIMITIVE? qui fonctionneront aussi bien pour les versions Apple et Commodore de LOGO.

```
POUR PROCÉDURE? :NOM
    SI NOMBRE? :NOM ALORS RÉSULTAT «FAUX
    SI LISTE? :NOM ALORS RÉSULTAT «FAUX
    TEST MOT? :NOM
    SIVRAI SI MOT? TEXTE :NOM ALORS
      RÉSULTAT «FAUX SINON SI NON (TEXTE :NOM = [])
      ALORS RÉSULTAT «VRAI
    RÉSULTAT «FAUX
FIN
POUR PRIMITIVE? :NOM
    SI APPARTIENT? :NOM ALORS RÉSULTAT «FAUX
    SI LISTE? :NOM ALORS RÉSULTAT «FAUX
    TEST MOT? :NOM
    SIVRAI SI MOT? TEXTE :NOM ALORS RÉSULTAT
      «VRAI SINON RÉSULTAT «FAUX
FIN
```

Bibliographie

Nous avons maintenant traité de tous les principaux aspects du langage LOGO standard, et couvert une vaste gamme d'applications. Si vous voulez en savoir plus sur LOGO, voici quatre livres que vous pourrez avantageusement lire :

- *Learning with Logo,* de Daniel Watt (McGraw-Hill). Livre d'initiation idéal, particulièrement efficace avec les enfants.
- *Logo,* de Harold Abelson (McGraw-Hill). Livre constituant le « standard » sur ce langage.
- *Turtle Geometry,* de Harold Abelson et Andrea diSessa (MIT Press). Traite de la géométrie de la tortue. Les notions de mathématiques qu'il suppose sont du niveau de terminale et de la fac. Un des derniers chapitres développe un simulateur de la Relativité Générale avec LOGO !

Ces trois ouvrages, destinés avant tout à un public que l'anglais n'effraie point, sont cependant accessibles au plus grand nombre. Nous donnerons prochainement des titres de publications en langue française sur LOGO.

Vertus du logo

- Est interprété comme le BASIC, aussi est-il facile d'exécuter et de modifier des programmes.
- Est structuré et doté de vraies procédures, contrairement à la plupart des versions du BASIC.
- Est extensible comme le FORTH, c'est-à-dire qu'il est possible de définir de nouveaux mots qui s'intègrent alors dans le vocabulaire de l'ordinateur.
- Peut traiter des listes comme LISP, aussi est-il utile pour explorer des domaines tels que l'intelligence artificielle.

LOGO n'est pas vraiment conçu pour élaborer des algorithmes parfaits et aboutis, mais est idéal pour dégrossir une question. Nous avons écrit la plupart de nos programmes en partant d'une simple procédure qui exécutait seulement une partie des tâches. Nous l'avons ensuite modifié pour finalement la mettre au point et approfondir en même temps notre compréhension du problème. En fin de compte, nous obtenions un algorithme bien pensé.

Vices du logo

- Les problèmes essentiels demeurent le manque d'espace de travail et la lenteur d'exécution.
- Certaines fonctions seraient bien utiles; LOGO manque notamment de fonctions de dépistage d'erreurs, de traitement de tableaux et de fichiers. Ces caractéristiques existent seulement pour certains LOGO.

Variantes de logo

Utilisez pour toutes les versions LCSI :

NOMBREP pour NOMBRE?
LISTP pour LISTE?
MOTP pour MOT?
VIDEP pour VIDE?

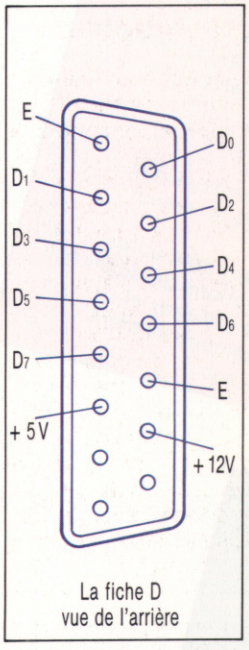
Le LOGO du Spectrum comporte la primitive COPIEDEF et PRIMITIVEP (ce qui correspond à PRIMITIVE?), et DÉFINIP (pour PROCÉDURE?).

Sur Atari, utilisez PC pour COULEURCRAYON, PH pour PHRASE, CT pour CACHETORTUE. DÉFINIS et TEXTE n'existent pas pour LOGO Atari, bien que le manuel indique une manière de les définir.

Les couleurs du crayon changent d'une machine à l'autre, PC-1 est utilisé ici pour effacer des lignes, mais certaines versions LOGO ont plutôt une primitive EC (EffaceCrayon).

Au travail!

Après l'installation des moteurs et la construction du circuit, nous allons connecter les moteurs à la fiche D et mettre au point une interface très simple avec le port utilisateur. Nous rédigerons aussi un programme permettant de tester le robot.



La fiche D vue de l'arrière

Connexions

Les diagrammes nous montrent comment faire les connexions suivant les modèles. A noter que les 11 fils nécessaires, à l'extrémité libre du câble, devront être dénudés, étamés et installés sur l'interface suivant les connexions appropriées. Les possesseurs d'un Commode 64 auront à se servir d'un connecteur plat à 24 voies et de câble-ruban à 12 fils. Il leur faudra faire bien attention en effectuant les connexions avec la fiche (comparez les diagrammes). C'est d'autant plus nécessaire que le connecteur peut être mis en place dans un sens ou dans l'autre; attribuez-lui arbitrairement un « haut » (TOP sur le diagramme), ainsi qu'à la fiche.
Une fois tout cela accompli, nous serons en mesure de raccorder notre interface au robot et au port utilisateur. Une source d'alimentation en courant continu de 12 V doit aussi être fixée sur la prise de 2,1 mm installée sur le circuit de l'interface.

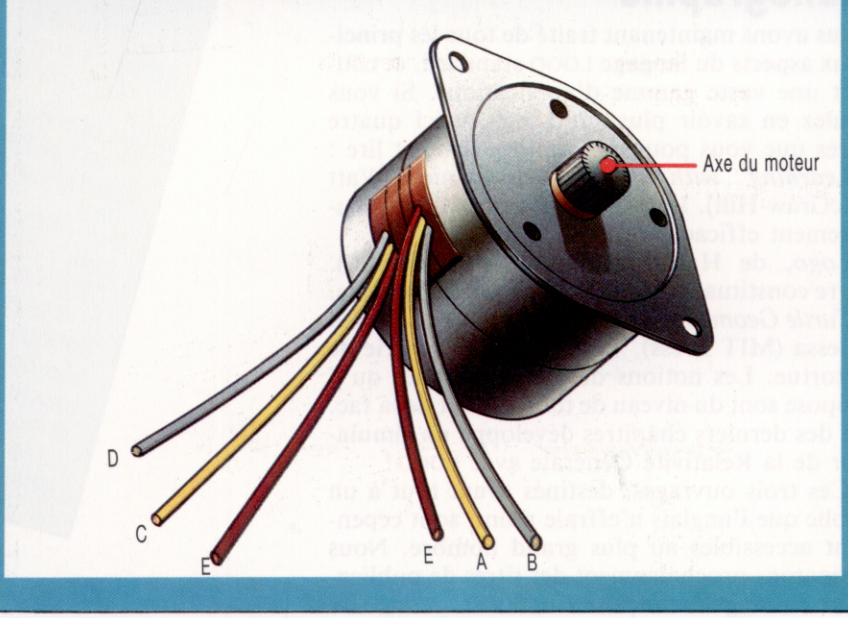
Un programme de contrôle

La première phase de la construction étant achevée, nous pouvons d'ores et déjà rédiger un programme permettant de contrôler le robot à partir du clavier. Les bits 0 à 3 du registre de données du port utilisateur contrôlent les moteurs. Le bit 0 commande la remise à zéro; sa valeur normale est de 1. Les bits 1 et 2 contrôlent respectivement le déplacement du moteur vers la droite et vers la gauche. Le bit 3 est le bit d'impulsion, qui ordonne aux moteurs d'accomplir un nouveau pas. Notre programme utilise les touches T, B, F et H pour contrôler la direction, et une boucle répétitive pour impulser les moteurs. (Cl. Kevin Jones.)

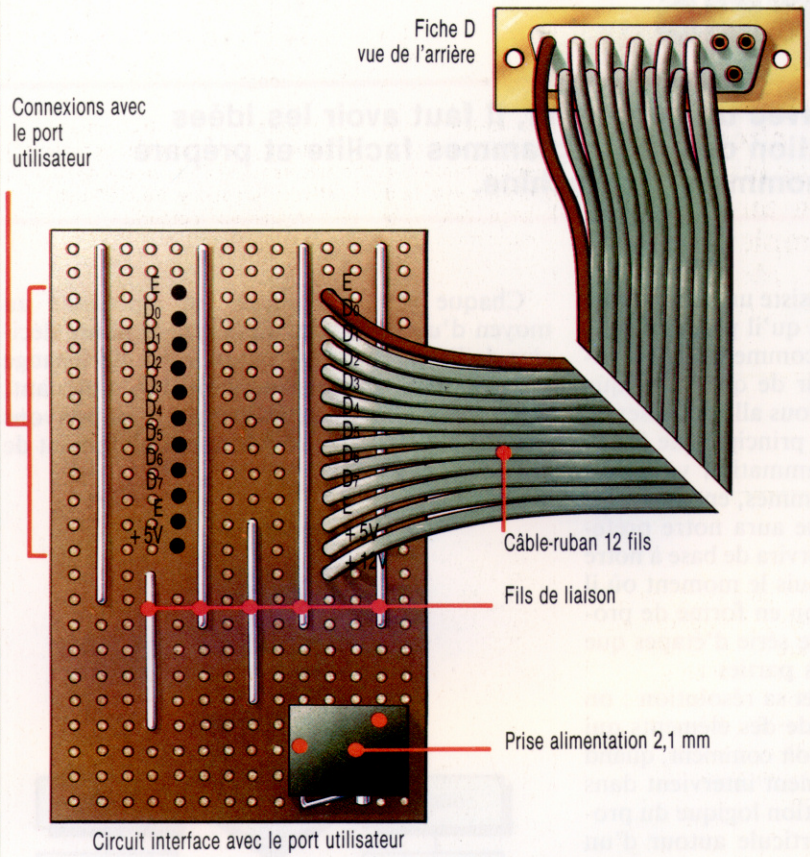
```
1000 REM **** CONTROLEUR ROBOT BBC ****
1010 DDR=&FE62:DATREG=&FE60:?DDR=15:REM:SORTIE LIGNES 0-3
1020 PROCinitialisation:REPEAT
1030 A$=INKEY$(1):IF A$<>"" THEN PROCtest_clavier
1040 PROCpulse(10)
1050 UNTIL A$="X":?DATREG=0:END
1060 DEF PROCinitialisation
1070 avant=4:arrière=2:gauche=6:droite=0
1080 dir=avant:?DATREG=dir+1:ENDPROC
1100 DEF PROCimpulsion
1110 FOR c=1 TO tm
1120 ?DATREG=(?DATREG OR 8):PROCretard(2)
1130 ?DATREG=(?DATREG AND 247):PROCretard(2)
1140 NEXT c:ENDPROC
1150 DEF PROCretard (n)
1160 FOR I=1 TO n:NEXT I
1170 ENDPROC
1180 DEF PROCtest clavier
1190 IF A$="T" THEN dir=avant
1200 IF A$="B" THEN dir=arrière
1210 IF A$="F" THEN dir=gauche
1220 IF A$="H" THEN dir=droite
1230 ?DATREG=((?DATREG AND 249)OR dir)
1240 ENDPROC


10   REM **** CONTROLEUR ROBOT CBM 64 ****
20   DDR=56579:DATREG=56577:POKEDDR,15
30   GOSUB1000:REM INITIALISATION
40   GETA$:IFA$<>"" THEN GOSUB3000:REM TOUCHE
50   M=10:GOSUB1500:REM IMPULSION
60   IFA$<>"X" THEN 40
70   POKEDATREG,0:END
1000 REM **** S/P INITIALISATION ****
1010 AVA=4:ARR=2:GAU=6:DRO=0
1020 DR=AVA:POKEDATREG,DR+1:RETURN
1500 REM **** S/P IMPULSION ****
1510 FOR C=1 TO M
1520 POKEDATREG,(PEEK(DATREG)OR8):GOSUB2000:REM RETARD
1530 POKEDATREG,(PEEK(DATREG)AND247):GOSUB2000:REM RETARD
1540 NEXT C:RETURN
2000 REM **** S/P RETARD ****
2010 FOR I=1 TO N:NEXT I:RETURN
3000 REM **** S/P TEST CLAVIER ****
3010 IF A$="T" THEN DR=AVA
3020 IF A$="B" THEN DR=ARR
3030 IF A$="F" THEN DR=GAU
3040 IF A$="H" THEN DR=DRO
3050 POKE DATREG,((PEEK(DATREG)AND249)ORDR)
3060 RETURN
```

Qu'est-ce qui manque?

Avant de connecter les moteurs et la fiche au circuit, reportez-vous au diagramme déjà donné. Il vous montre comment souder les fils. D'abord les moteurs. De chacun d'eux émergent 6 fils, qui sont répartis en deux groupes de trois. Les fils jaunes et gris (A et B) sont les plus proches de l'axe du moteur. Soudez chaque fil bien à sa place sur le circuit, en tenant compte des lettres attribuées à chacun d'eux.



Kevin Jones



Circuit interface avec le port utilisateur

Liste des pièces

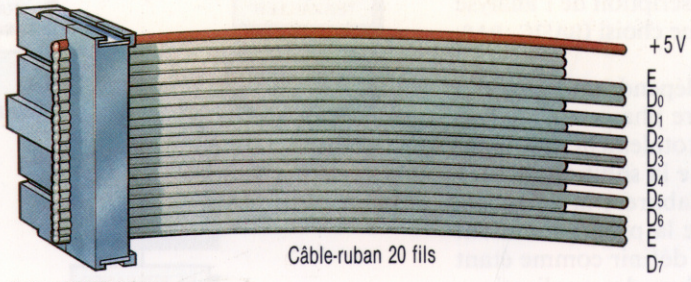
Nombre d'éléments
1 fiche D 15 voies
1 bouchon pour fiche D 15 voies
1 prise alimentation de 2,1 mm
1 connecteur IDC 20 voies (BBC)
1 connecteur plat 24 voies
1 jeu de pastilles adhésives

Accessoires
4 m de câble-ruban 12 fils
1 m de câble-ruban 20 fils
1 prise alimentation 12 V, 1 A

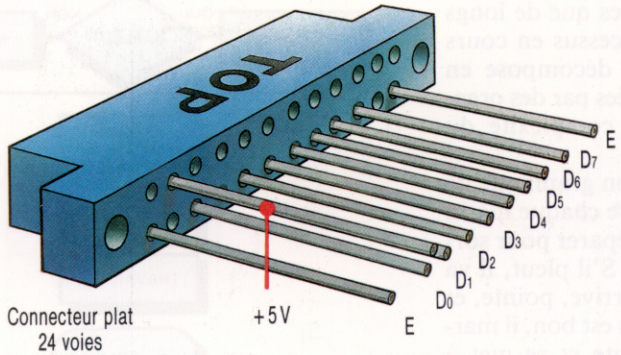
Interface avec le port utilisateur

Les connexions internes de notre robot étant réalisées, il nous faut maintenant mettre au point une interface très simple qui permette à la fois de le raccorder au port utilisateur, d'où nous le contrôlerons, et de lui fournir le courant de 12 V dont les moteurs pas à pas ont besoin. Coupez un morceau de veroboard (50 trous par 24 bandes), et fixez-y 3 m de câble-ruban 12 voies, de la façon indiquée sur le diagramme. Le fil rouge installé sur l'un des côtés du câble vous servira de repère. Soudez les 12 fils sur la fiche. Montez la prise d'alimentation de 2,1 mm sur le circuit, en prenant garde à l'orientation des broches (le pôle central est négatif). Ensuite mettez en place les fils de liaison comme indiqué. Reste à installer les connexions avec le port utilisateur, mais ici possesseurs de BBC Micro et de Commodore 64 doivent recourir à des procédés différents, puisque leurs ports utilisateurs ne sont pas les mêmes.

BBC Micro



Commodore 64



Qu'est-ce que je fais là?

Une fois les connexions propres aux moteurs menées à bien, reste à faire les raccordements avec la fiche D, montée sur le capot du robot. Le diagramme (page de gauche, en haut) vous montre les emplacements spécifiques de la fiche (celle-ci étant vue du dessous par rapport au capot). Grâce à un câble-ruban à 12 fils, connectez au circuit les lignes de données (D_0 à D_3), les lignes de 5 et 12 V, et la terre. Pour cela, reportez-vous une fois de plus au diagramme déjà donné, qui présente toutes les connexions de façon explicite. La ligne d'alimentation de 12 V, en particulier, doit se trouver exactement à la position qui vous est spécifiée. Faute de quoi, les circuits internes de votre ordinateur pourraient être endommagés... Les lignes de données (D_4 à D_7) n'ont pas besoin d'être connectées pour le moment, puisqu'elles ont pour fonction de transmettre les signaux émis par les capteurs que nous fixerons ultérieurement. Toutes les connexions internes au robot sont désormais en place. Choisissez un endroit approprié à l'intérieur du boîtier pour y insérer le circuit, que vous installerez à l'aide de pastilles adhésives pour éviter le jeu. Refermez le capot et vissez-le à l'aide de quatre vis de coin.

Pas à pas

Pour communiquer avec un ordinateur, il faut avoir les idées très claires. L'utilisation des organigrammes facilite et prépare ce dialogue entre l'homme et la machine.

Vous savez déjà en quoi consiste un programme et les multiples applications qu'il permet. Mais peut-être ne savez-vous pas comment ils se construisent, c'est-à-dire à partir de quelle logique interne ils se développent. Nous allons passer en revue un certain nombre de principes, de façon que la technique de programmation vous soit facilitée. L'usage des diagrammes, encore appelés organigrammes (ce terme aura notre préférence) ou ordinogrammes, servira de base à notre travail. Tout problème, depuis le moment où il est posé jusqu'à sa résolution en forme de programme, doit passer par une série d'étapes que l'on peut regrouper en trois parties :

1. Analyse du problème et sa résolution : on comprend par analyse l'étude des éléments qui forment partie du projet ; savoir comment, quand et de quelle manière tel élément intervient dans ce qui finit par être une solution logique du problème. La construction s'articule autour d'un ensemble d'idées qui se suivent de manière séquentielle jusqu'à une solution.
2. Organigramme : représentation graphique de l'analyse.
3. Codification : c'est l'écriture du programme, c'est-à-dire la transcription de l'analyse dans un langage informatique choisi (BASIC, PASCAL, etc.).

Cette troisième phase dépend toujours des deux premières. La première phase — l'analyse — est sans aucun doute totalement indispensable, puisqu'elle représente la solution logique que seule la pensée peut élaborer ; la deuxième n'est que l'outil qui facilite la programmation.

L'organigramme peut se définir comme étant le moyen qui facilite l'analyse des applications par l'utilisation de figures géométriques symboliques représentant les différentes phases de la solution d'un problème.

On utilise ces dessins graphiques, car ils se révèlent plus clairs et plus simples que de longs textes conventionnels. Tout processus en cours pour résoudre un problème se décompose en séquences élémentaires, symbolisées par des organigrammes, quelle que soit la complexité du problème.

La figure 1 est la représentation graphique de cet exemple simple. « Juan réalise chaque matin les actes suivants : se lever, se préparer pour sortir et vérifier le temps qu'il fait. S'il pleut, il va en voiture jusqu'à son bureau, arrive, pointe, et commence à travailler. Si le temps est bon, il marche jusqu'à son bureau, il pointe et se met à travailler. »

Chaque acte, ou phase, est représenté au moyen d'un rectangle, alors que la phase décisive de l'état du temps apparaît par un losange (cela pourrait être un hexagone !). Maintenant, confrontés à un même problème, nous pouvons aboutir à différentes conclusions au moment de

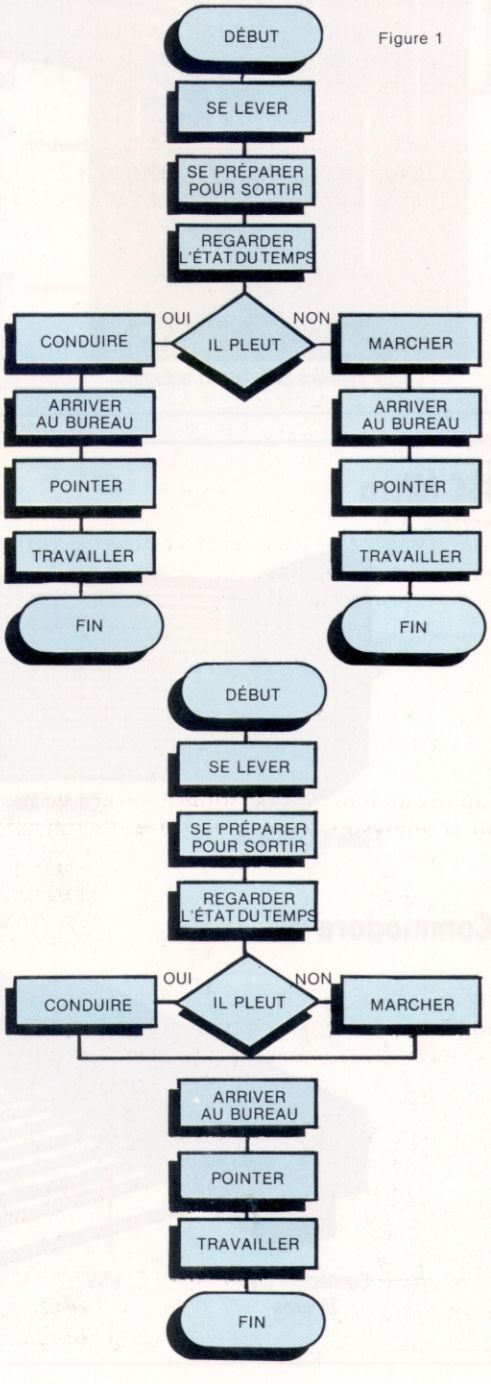


Figure 1

présenter une solution. Si toutes sont bonnes, laquelle choisirons-nous? Il apparaît toujours plus valable de choisir la solution dans laquelle il n'est pas nécessaire de réécrire des parties identiques du programme, dénommées « routines », qui sont communes à plusieurs chemins fondamentaux.

Ainsi, la figure 2 illustre comment on peut éviter la répétition de l'espace dans lequel sont représentés les éléments arriver au bureau, pointer et travailler. A partir de ces éléments fondamentaux, on comprend qu'une des tâches confiées à l'analyste est de combiner les différents composants du système pour arriver, dans le temps le plus court possible aux meilleurs résultats. Nous devons d'abord analyser comment circulent les données, puis ordonner ces flux d'informations en séquences afin d'introduire un système de codification. Il est clair que cette partie de l'analyse ne permet pas, par elle-même, de confectionner un programme.

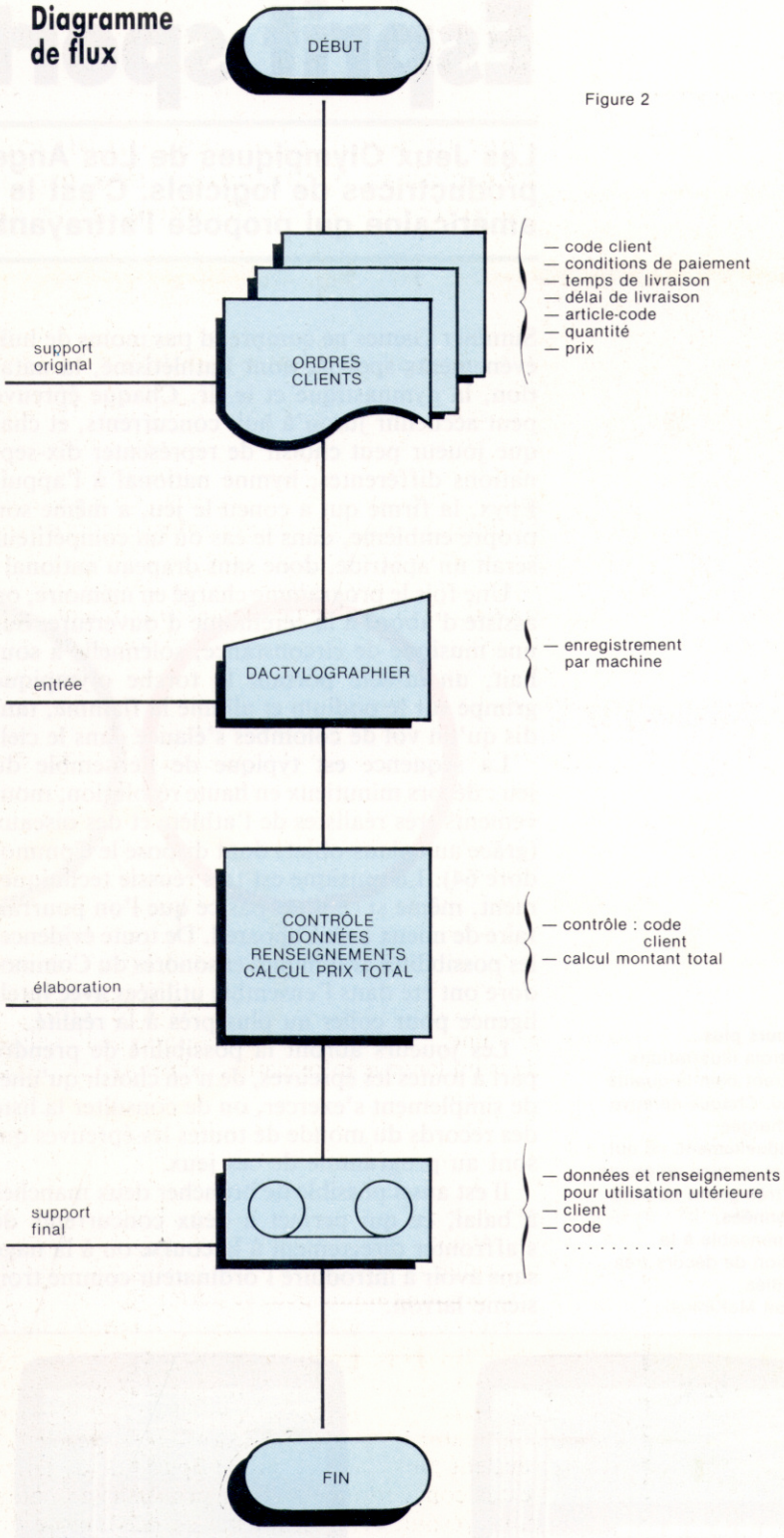
L'usage des organigrammes, appliqué à l'automatisation de l'information, est, en soi, trop général. Nous allons distinguer deux types de diagrammes : le « diagramme de flux » appelé aussi de système, et le « diagramme de blocs » ou de programme. Un diagramme de flux cherche à représenter graphiquement le passage des données en expliquant, à grands traits, la résolution et son processus. Il reproduit également une configuration nécessaire pour résoudre le problème, mais à l'aide d'opérations peu élaborées au niveau des différentes étapes. Mais cette vision globale a son revers : nous manquons d'informations nécessaires, particulières, pour pouvoir traduire et créer le programme. Mais à quoi sert un diagramme de flux? Nous pouvons répondre de la manière suivante : il décrit toutes les phases d'un processus; il assigne et définit les composants du système qu'on utilisera à chacune des phases; il met en relation les différentes phases et leurs composants.

Supposons qu'une petite entreprise ait l'intention d'automatiser son système de commandes et de transformer ses archives traditionnelles à l'aide d'un ordinateur. La personne chargée de ce travail (l'analyste) reçoit une série d'explications concernant la manière dont s'était déroulé le travail jusqu'alors : « Les clients font leurs commandes que nous recueillons dans des fichiers. Ceux-ci contiennent des renseignements tels que des numéros de code, le nom du client, les articles, les nombres d'unités, etc. Nous cherchons ensuite dans les archives la fiche du client et une fois les informations rassemblées, nous lui appliquons les conditions de paiement, remise et autres détails qui figurent sur sa fiche. Puis nous calculons le montant de sa commmande. Enfin, nous archivons la fiche corrigée. »

Notre analyste, qui pourrait bien être vous-même, ayant compris la méthode exposée et appris que l'entreprise ne souhaitait pas introduire des variations d'aucun type, étudie le problème et aboutit à la conclusion générale suivante : selon lui, les étapes essentielles à développer sont :

Diagramme de flux

Figure 2



• Recevoir le support original des ordres de commande.
• Introduire les données dans l'ordinateur.
• Réaliser les vérifications nécessaires et, une fois ce contrôle établi, calculer à combien se monte le total de la commande. Cette étape se fera intégralement de façon automatique.
• Procéder à l'enregistrement du dossier du client, dûment actualisé, sur le support (magnétique) final. Cette démarche est traduite par l'organigramme qui apparaît ci-contre.

Esprit sportif

Les Jeux Olympiques de Los Angeles ont donné des idées aux firmes productrices de logiciels. C'est le cas, en particulier, d'une entreprise américaine qui propose l'attrayant Summer Games.

Summer Games ne comprend pas moins de huit événements sportifs dont l'athlétisme, la natation, la gymnastique et le tir. Chaque épreuve peut accueillir jusqu'à huit concurrents, et chaque joueur peut choisir de représenter dix-sept nations différentes, hymne national à l'appui. Epyx, la firme qui a conçu le jeu, a même son propre emblème, dans le cas où un compétiteur serait un apatride, donc sans drapeau national!

Une fois le programme chargé en mémoire, on assiste d'abord à la cérémonie d'ouverture. Sur une musique de circonstance, solennelle à souhait, un athlète portant la torche olympique grimpe sur le podium et allume la flamme, tandis qu'un vol de colombes s'élance dans le ciel.

La séquence est typique de l'ensemble du jeu : décors minutieux en haute résolution, mouvements très réalistes de l'athlète et des oiseaux (grâce aux plans-objets dont dispose le Commodore 64). La musique est très réussie techniquement, même si ce n'est pas ce que l'on pourrait faire de mieux avec l'appareil. De toute évidence, les possibilités graphiques et sonores du Commodore ont été dans l'ensemble utilisées avec intelligence pour coller au plus près à la réalité.

Les joueurs auront la possibilité de prendre part à toutes les épreuves, de n'en choisir qu'une, de simplement s'exercer, ou de consulter la liste des records du monde de toutes les épreuves qui sont au programme de ces jeux.

Il est aussi possible de brancher deux manches à balai, ce qui permet à deux concurrents de s'affronter directement à la course ou à la nage sans avoir à introduire l'ordinateur comme troisième larron.

On commence par le saut à la perche, qui est sans doute le moment le plus difficile de la compétition. La première hauteur à franchir est fixée à quatre mètres (loin du record du monde, mais il faut bien commencer prudemment).

Il faut d'abord déplacer le manche à balai vers l'arrière afin de déterminer la position de départ de la perche, puis vers l'avant afin que l'athlète s'élance au-dessus de la barre; enfin, on appuie sur le bouton de mise à feu pour qu'il lâche la perche et franchisse l'obstacle sans le heurter. Tout cela doit être fait à la seconde près, et toute erreur d'appréciation vous vaudra un échec humiliant en voyant la barre tomber sans que rien ne la retienne.

Prouesses techniques

Tout cela n'a pas dû être très facile à programmer : non seulement il faut prendre en compte le défilement d'écran, mais aussi s'adapter très rapidement à tous les mouvements du manche à balai et à la pression du bouton de mise à feu. L'ordinateur doit également s'assurer que le sauteur est en mesure — ou non — de réussir un saut correct. Naturellement, le joueur ne doit se rendre compte de rien pendant qu'il manipule ses différentes manettes!

A l'issue de chaque épreuve, un tableau des résultats est affiché, tandis que retentit l'hymne national du pays vainqueur. L'ordinateur procède ensuite au chargement de l'épreuve suivante; Summer Games est en fait une suite de programmes réunis sous un seul titre, ce qui montre bien la complexité des problèmes de programmation qu'il a fallu résoudre pour obtenir une simulation réaliste.

Les autres épreuves ont la même qualité que la première. Soyez prévenus : le maniement du manche à balai varie avec chacune d'elle, et il vous faudra bien des heures de pratique avant de pouvoir faire bonne figure. Mais n'en va-t-il pas de même dans la vie?

Summer Games : pour le Commodore 64.

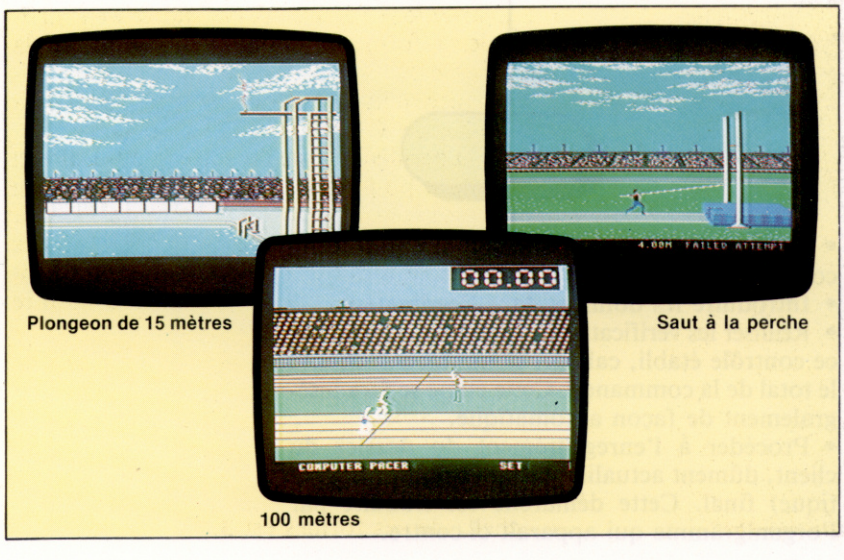
Éditeurs : Epyx Software.

Auteurs : Randy Glover, Stephen Landrum, John Leupp, Brian McGhie, Stephen Mundry, Erin Murphy, Scott Nelson.

Manche à balai : indispensable.

Format : cassette ou disquette.

Toujours plus...
Ces trois illustrations montrent bien la qualité du jeu. Chaque épreuve est chargée individuellement, ce qui permet la mise en œuvre d'un très grand nombre de données, indispensable à la création de décors très travaillés.
(Cl. Ian McKinnell.)



Plongeon de 15 mètres — Saut à la perche — 100 mètres