

# Géométrie dynamique et manipulation directe

Eric HAKENHOLZ

Novembre 2006

## Résumé

Les sections 1 et 2 peuvent se lire indépendamment, mais on ne peut pas affirmer qu'elles sont indépendantes. Dans un premier temps, je donne quelques éléments visant à placer la géométrie dynamique dans son contexte informatique pour ensuite, dans la partie B, donner quelques caractéristiques du logiciel que je développe : CaRMetal<sup>1</sup>.

## 1 La géométrie dynamique dans son contexte

### 1.1 Retour aux sources informatiques

Le concept de géométrie dynamique peut difficilement être séparé de son lieu d'implémentation, à savoir l'environnement informatique. Au milieu des années 80 apparaissait sur macintosh un logiciel particulièrement innovant pour l'enseignement des mathématiques qui permettait de créer dans une fenêtre de travail, à l'aide uniquement de la souris, des droites et des cercles liés éventuellement entre-eux par des relations géométriques. Ce périphérique de pointage - la souris -, nouveau venu dans le monde de l'informatique personnelle, permettait aussi de déplacer directement un à un les objets libres<sup>2</sup> ou semi-libres<sup>3</sup> dans le plan tout en conservant les relations définies pendant la phase de construction. La géométrie dynamique était née, et le logiciel s'appelait Cabri-Géomètre.

Mais revenons au macintosh. Le fait que Cabri-Géomètre fût développé sur cette plate-forme - dans ce système d'exploitation - avant d'être porté sur une autre, n'est pas anodin. Dans les années 83-84, le laboratoire dans lequel est né Cabri (LSD2 : Laboratoire de Structures Discrètes et de Didactique) fait l'acquisition de LISA d'Apple et ensuite de Macintosh :

*« Il apparaît en effet que les aspects (révolutionnaires pour l'époque) de convivialité et d'interactivité de ces ordinateurs correspondent exactement aux principes de CABRI. »<sup>4</sup>*

Tout n'était bien sûr pas donné au départ, et le logiciel comme certains de ses « principes » ont beaucoup évolué ensuite. Mais à l'époque, les principes de Cabri sont en même temps les principes d'un grand nombre de logiciels qui n'ont strictement aucun rapport avec la géométrie : ils sont très étroitement liés à la notion générale de manipulation directe.

Ce terme, introduit et défini pour la première fois en 1983 par Ben Shneidermann, apparaît dans un contexte où dix années de recherches ont déjà été menées dans le cadre de l'interaction homme-machine, sur des systèmes d'exploitations de type graphique. Dans les années soixante-dix, les travaux du Xerox PARC (Palo Alto Research Center) sur la question des interfaces graphiques constituent le big-bang d'un univers maintenant familier à tous ceux qui utilisent de près ou de loin l'outil informatique. La métaphore du bureau, les icônes, les dossiers, l'édition de texte WYSIWYG (What You See Is What You Get) sont autant d'idées novatrices, implémentées par les équipes de chercheurs du PARC dans le premier ordinateur personnel basé sur une interface graphique : le Star (1981). Les pionniers du PARC n'ont toutefois pas réussi à livrer en 81 une interface qui soit véritablement en manipulation directe : beaucoup d'actions comme un simple déplacement d'icône, l'ouverture d'un fichier, ou une suppression nécessitaient des actions combinées souris/clavier,

---

<sup>1</sup><http://db-maths.nuxit.net/CaRMetal/>

<sup>2</sup>Un « point » crée sans contraintes pourra être déplacé librement dans le plan

<sup>3</sup>Un « point sur objet » peut se déplacer sur une droite, un segment, un cercle, etc.

<sup>4</sup>lire la source : <http://www-leibniz.imag.fr/GRAPH/francais/histocabri.html>

sans compter que les fenêtres ouvertes ne pouvaient se déplacer.

Ce sont les informaticiens d'Apple qui, tout en reprenant la métaphore du bureau du PARC, ajoutent à l'édifice la possibilité de manipuler directement - par un seul clic - les composants graphiques de l'interface. Le LISA d'Apple (83) est ainsi le premier ordinateur personnel qui permet à l'aide de la « souris seule » de déplacer des icônes, les pré-détruire (invention de la corbeille), déplacer les fenêtres (avec rafraîchissement d'arrière-plan), sélectionner plusieurs fichiers par un rectangle de sélection et les ouvrir (avec gestion multi-tache), etc...

## 1.2 La manipulation directe - l'engagement direct

Dans les interfaces à manipulation directe, l'utilisateur a l'impression que c'est lui-même qui agit sur des objets dotés de réactions spécifiques, plutôt que de communiquer avec un interlocuteur auquel il demande des informations, ou ordonne d'exécuter des actions. Les quatre principes retenus par Ben Shneidermann pour définir la notion de manipulation directe sont :

1. Représentation permanente des objets d'intérêt
2. Utilisation d'actions physiques au lieu de commandes textuelles : « voir et montrer au lieu de se rappeler et taper »
3. Le résultat des actions physiques doit être immédiatement visible.
4. Ces actions doivent être rapides, incrémentales et réversibles.

La géométrie dynamique, dès sa création, essaie de proposer à l'utilisateur un type d'interaction conforme à ces règles, que ce soit pendant la phase de création de la figure ou pendant celle de la manipulation des constituants :

1. Les objets non cachés volontairement sont représentés à l'écran
2. Les objets se créent et se déplacent à la souris
3. Le déplacement d'un point libre ou semi-libre entraîne une modification sans délai de l'aspect de la figure
4. Lors de la phase de création, on évite la présence d'intermédiaires comme l'arrivée de boîtes à dialogues (actions rapides). Si cela est justifié géométriquement, le déplacement des objets se fait pour l'oeil de façon continue (actions incrémentales), et à tout moment la possibilité est donnée de revenir en arrière (actions réversibles).

Le principe de la rapidité des actions (règle 4) est souvent malmené. Certains logiciels utilisent, pour des actions courantes et simples, des boîtes de dialogues bloquantes qui attendent de l'utilisateur une modification des attributs du composant sélectionné (un mot dans un traitement de texte, une cellule dans un tableur, un point dans un logiciel de géométrie, etc...). On est alors dans des situations dites « modales », qui font que l'utilisateur ressent la manipulation comme étant INdirecte. On s'aperçoit presque toujours qu'il est possible, avec certes plus d'efforts de conception et de programmation, de supprimer ces boîtes à dialogues modales et ainsi faire en sorte que l'action soit directe sur l'objet, sans intermédiaire.

La question des menus est sensiblement du même ordre : un logiciel dans lequel des actions courantes ne peuvent s'effectuer que par déroulement de menus augmente considérablement le temps de réalisation d'un projet, et diminue énormément pour l'utilisateur l'impression de manipulation directe.

La notion de manipulation directe a énormément progressé avec les années et la définition de Shneidermann, quelque peu rigide et très centrée sur l'informatique, a donné lieu à de multiples compléments basés sur une approche plus cognitive, plus centrée sur l'utilisateur. Le concept d'engagement direct est l'une des ces approches.

*« L'impression d'engagement direct correspond à ce que l'utilisateur ressent lorsqu'il peut agir directement et librement sur les représentations des objets de son propre monde et percevoir de façon immédiate leurs réactions. » - Nanard, 1990*

Prenons l'exemple d'une construction de figure : certains logiciels de géométrie dynamique indiquent précisément à l'utilisateur ce qu'il est en train de faire ou ses intentions par l'intermédiaire d'un système de retour d'informations textuelles ou graphiques. Ce sont là des éléments importants visant à améliorer l'impression d'engagement direct : en phase d'apprentissage, la présence dans le logiciel d'un mécanisme de "feedbacks" approprié sera pour beaucoup dans la rapidité d'appropriation de l'outil.

Cabri et CaRMetal sont deux logiciels dans lesquels les étapes d'une construction sont jalonnées par des retours d'informations systématiques. Dans le même temps, les choix faits par l'un et l'autre dans ce domaine sont de natures différentes : Cabri privilégie l'indication textuelle, et CaRMetal renvoie exclusivement des réponses de type graphique. Ce dernier logiciel possède une couleur strictement réservée aux retours d'information : le jaune fluorescent.

Voici quelques copies écran réalisées dans ces deux logiciels qui illustrent quelques-unes des nombreuses situations d'engagement direct :

**Exemple 1.** *Ce qui se passe après le premier clic, lors de la création d'un objet « à la volée » (ici d'un cercle) :*

**Avec Cabri :** voir FIG.1. Le centre « clignote », le cercle apparaît et « suit » la souris jusqu'au second

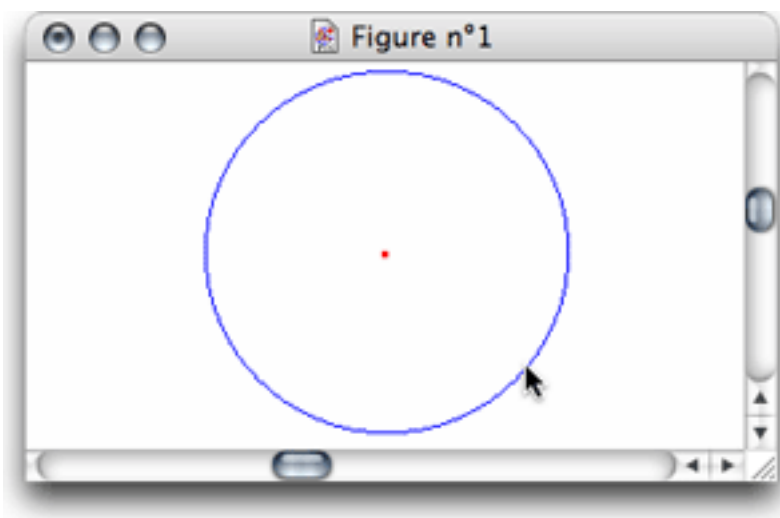


FIG. 1 – Cercle avec Cabri

*clic.*

**Avec CaRMetal** voir FIG.2.

*Le cercle apparaît, suit la souris, mais en jaune fluo jusqu'au second clic.*

Dans ce cas-là les différences sont d'ordre graphique uniquement et ne sont pas aussi prononcées que dans les exemples qui suivent :

**Exemple 2.** *Placement d'un point sur un objet : Après avoir choisi l'outil point, on s'approche d'un objet :*

**Avec Cabri :** voir FIG.3 et FIG.4.

**Avec CaRMetal** voir FIG.5 et FIG.6.

Les concepteurs de Cabri ont fait le choix du texte pour le suivi des intentions de l'utilisateur. Ils ont poussé très loin cette approche textuelle et on est souvent très étonné par la précision des messages de Cabri. Ainsi, par exemple, si on fait passer une conique par cinq points de la courbe  $y = 1/x$ , le message affiché au survol de cet objet sera « Cette hyperbole équilatère ». Le message sera bien entendu « Cette parabole » pour la courbe  $y = x^2$ . Force est de reconnaître que ce travail de programmation est remarquable.

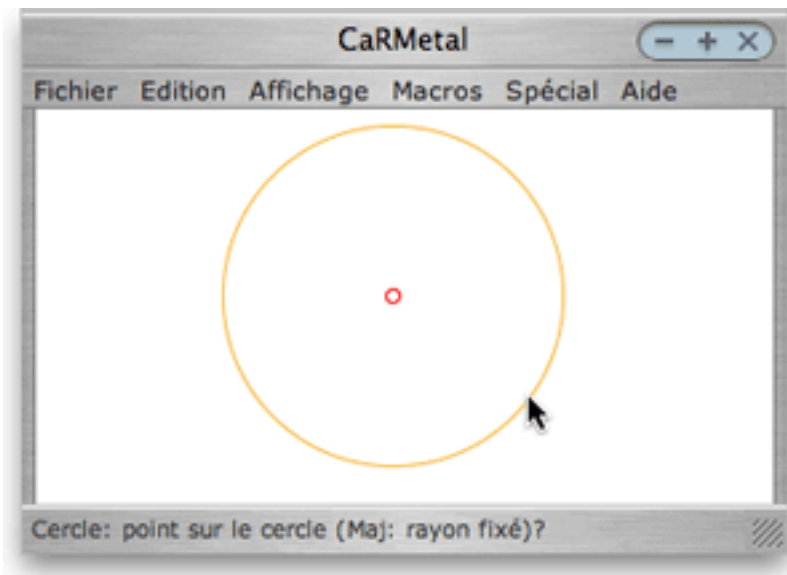


FIG. 2 – Cercle avec CaRMetal

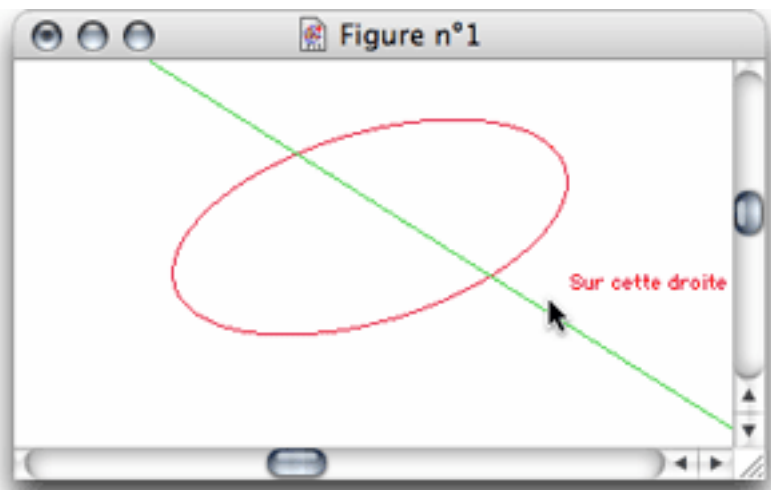


FIG. 3 – Point sur une droite avec Cabri : Le message « sur cette droite » apparaît

**Exemple 3.** *Placement d'un point à une intersection d'objets :Après avoir choisi l'outil point, on s'approche de l'intersection.*

**Avec Cabri :** voir FIG.7.

**Avec CaRMetal** voir FIG.8.

*CaRMetal est lui complètement « muet » d'un point de vue textuel, et ne répond que de manière graphique (comme le fait aussi le logiciel GeoGebra).*

*Avec Cabri, la non-lecture des messages peut mener à des erreurs de construction : si on attend juste l'arrivée d'un texte sans faire attention à son sens, on peut placer un point « sur cet objet » au lieu de le créer comme on le souhaite « à cette intersection ». Pour cette raison le formateur doit insister sur le rôle des messages qu'il faut lire, en faisant comprendre en substance que le message texte n'est pas un indicateur graphique binaire qui apparaît ou disparaît, comme une sorte de bascule « bon /pas bon ».*

*Il y a dans cet objectif de formation (la prise en compte du texte), un travail à « contre-culture » à mener qui, s'il est absolument nécessaire dans d'autres situations, peut tout à fait être évité dans un logiciel comme CaRMetal. Pour reprendre l'exemple précédent, la réponse de type graphique dit immédiatement, sans*

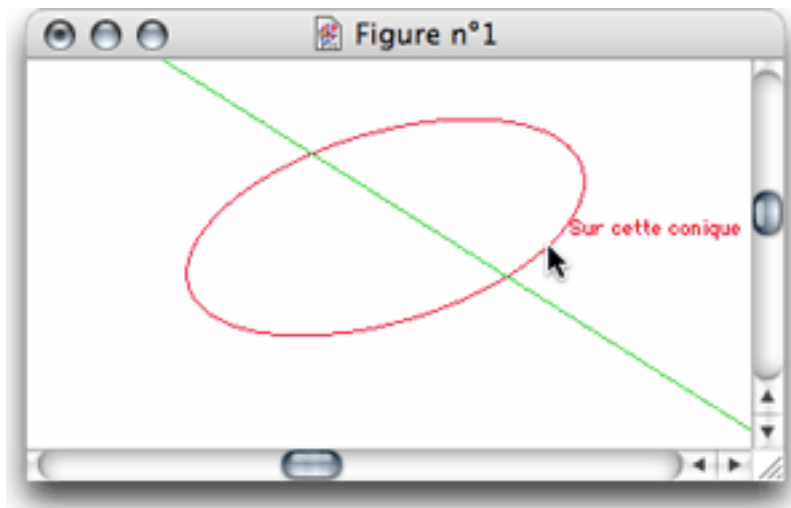


FIG. 4 – Point sur une conique avec Cabri : Le message « sur cette conique » apparaît

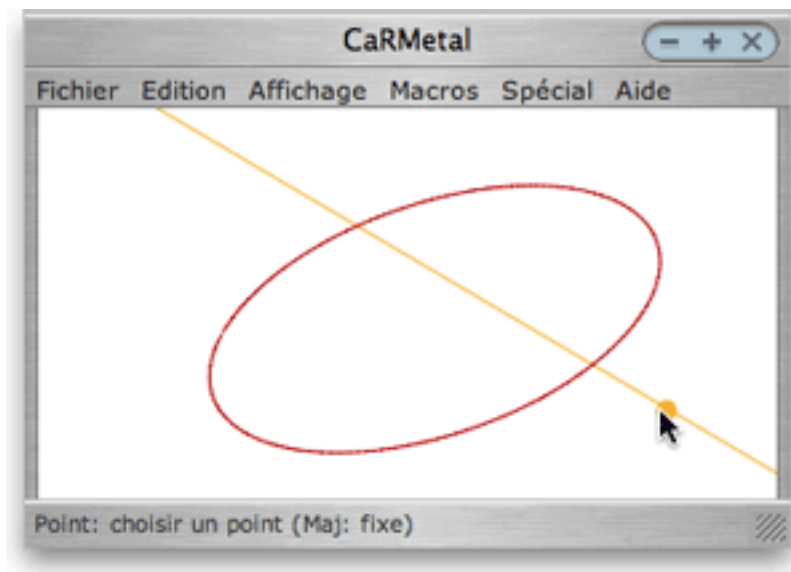


FIG. 5 – Point sur une droite avec CaRMetal : La droite passe en jaune fluo, et un point apparaît sur cet objet

*ambiguïté, sans interprétation sémantique, qu'on est sur l'objet, ou à cette intersection. Dans ce cas, c'est l'objet lui-même qui réagit.*

### 1.3 « Voir et montrer plutôt que de se rappeler et taper »

Les premiers systèmes informatiques étaient de type conversationnels. L'interaction homme-machine se faisait alors uniquement par le biais d'un langage dérivé des langues naturelles, à travers l'utilisation du clavier. Un grand inconvénient de ces systèmes est qu'ils exigent non seulement de mémoriser des commandes, mais aussi leur organisation stricte dans des phrases du langage. Cette double difficulté - lexicale et syntaxique - a conduit petit à petit à une situation où aujourd'hui 99% des ordinateurs personnels de la planète sont pilotés par l'utilisateur via une interface graphique. Une « major » de l'informatique a d'ailleurs dû sûrement comprendre - un peu tard - dans les années 90, où était son profit en reprenant l'idée phare des pionniers du PARC et d'Apple qui est que l'informatique ne doit plus être une affaire d'informaticiens<sup>5</sup>.

<sup>5</sup> « Simple things should be simple, complex things should be possible » - Alan Kay, un des fondateurs du PARC

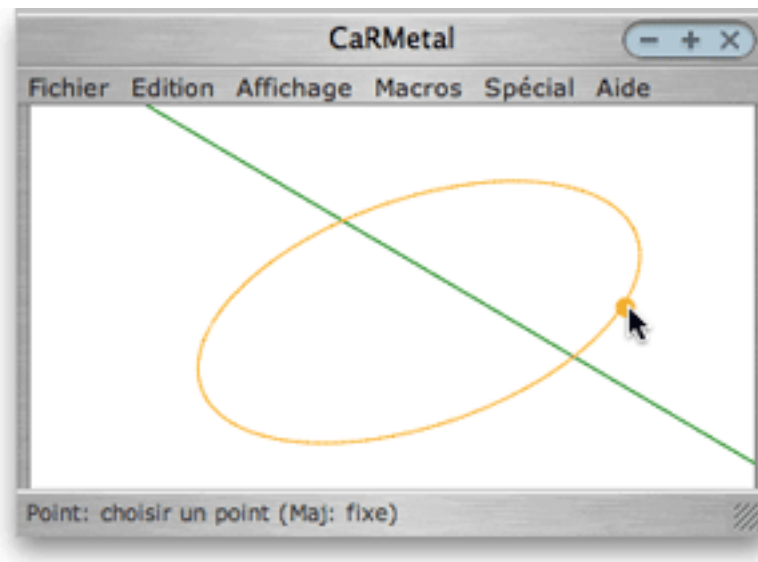


FIG. 6 – Point sur une conique avec CaRMetal : La conique passe en jaune fluo, et un point apparaît sur cet objet

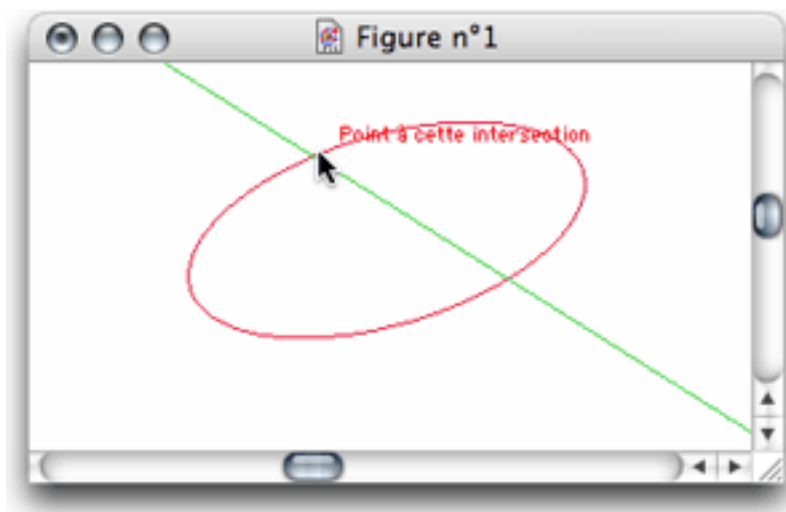


FIG. 7 – Intersection d'objets avec Cabri : Le message « Point à cette intersection » apparaît.

Aujourd'hui, les logiciels « anciens » que l'on pensait réfractaires, par leur nature, au passage au graphisme en manipulation directe ont franchi le pas, ou ont tout simplement disparu. Malgré cela, encore aujourd'hui, une objection commune des tenants de la ligne de commande géométrique est de dire que « oui, mais pour l'enseignement des mathématiques c'est différent » affirmant en cela une sorte d'exception culturelle qui nous séparerait du monde informatique dans lequel nous évoluons...

En dehors du calcul des prédicats, formalisation du langage mathématique que les programmes nous demandent de ne pas aborder dans le secondaire, le vecteur de l'écrit mathématique est la langue naturelle. Un même enseignant appréhendera par exemple de la même manière, dans un programme de construction, les situations suivantes :

1. l'élève 1 écrit : "Soit  $M$  un point de la droite  $(AB)$ "
2. l'élève 2 écrit : "Soit  $M$  un point aligné avec  $A$  et  $B$ "
3. l'élève 3 écrit : "Donnons-nous un point  $M$  sur la droite  $(AB)$ "
4. etc...

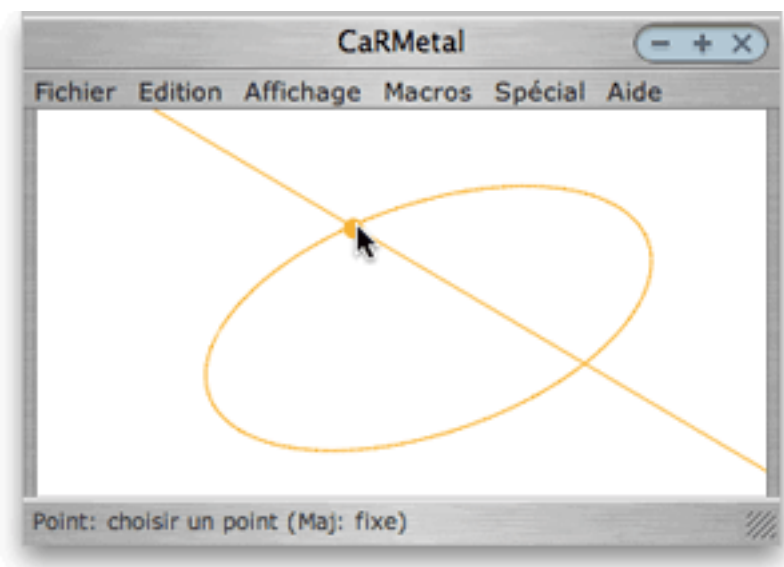


FIG. 8 – Intersection d’objets avec CaRMetal : Les deux objets passent en jaune fluo, et un point apparaît à l’intersection.

Cet écrit mathématique, comme son support naturel, utilise une surjection « forte » entre syntaxe et sémantique.

A contrario, les langages de commande imposent la bijection, et tous les logiciels de géométrie « à ligne de commande » qui ont jalonné la courte histoire de l’informatique se sont toujours comportés de ce point de vue comme un instituteur du XIX<sup>e</sup> (toute ressemblance avec des personnage existant ou ayant existé etc...) qui sanctionnerait sévèrement toute syntaxe qui ne correspondrait pas mot pour mot à « la formulation du professeur ».

Par ailleurs, les logiciels de géométrie « conversationnels » apportent quelquefois des réponses aux actions textuelles qui peuvent porter sur des problèmes syntaxiques propre au langage, et qui n’ont bien souvent que peu de sens pour l’élève. En tous cas ce type de réponse n’a que peu de rapport avec l’action de « faire de la géométrie », raison pour laquelle nous plaçons nos élèves devant un logiciel de géométrie dynamique.

Lorsqu’elles accompagnent la manipulation directe, l’interprétation et la réponse du logiciel se manifestent à un tout autre niveau : celui de la matérialisation graphique des objets, qui est beaucoup plus accessible à l’élève, a fortiori celui du XXI<sup>e</sup> siècle. Par ailleurs, dans une phase d’approche et d’appropriation, la géométrie peut très bien s’enseigner sur des figures et à travers des figures, et les enseignants sont là aussi pour que le langage reste présent - omniprésent - dans une phase de conceptualisation.

La géométrie dynamique, dans son principe d’origine - amélioré au fil du temps - de manipulation directe, constitue de son côté un formidable outil d’apprentissage et d’investigation de la géométrie. Lorsqu’on confronte nos élèves à ce type de logiciel, on sait exactement ce que l’on teste.

A côté de cela, certains logiciels de géométrie dynamique en manipulation directe sont complétés par des systèmes de scripts, qui permettent d’élaborer des figures qui ne pourraient pas voir le jour sans l’aide d’un langage de programmation. Dr Geo, par exemple, permet aussi de construire des figures par l’intermédiaire de scripts.

**Exemple 4** (Script récursif). *Réalisé avec Dr Geo*<sup>6</sup>

```
(new-figure "Spiral")
(define (square p1 p2 p3 p4 n)
```

---

<sup>6</sup><http://www.ofset.org/drgeo>

```

(let* ((s1 (Segment "" extremities p1 p2))
(s2 (Segment "" extremities p2 p3))
(s3 (Segment "" extremities p3 p4))
(s4 (Segment "" extremities p4 p1))
(A (Point "" on-curve s1 1/10))
(B (Point "" on-curve s2 1/10))
(C (Point "" on-curve s3 1/10))
(D (Point "" on-curve s4 1/10)))
(send A masked)
(send B masked)
(send C masked)
(send D masked) (if (> n 0)
(square A B C D (- n 1))))
(lets Point "M" free 5 5)
(lets Point "N" free -5 5)
(lets Point "O" free -5 -5)
(lets Point "P" free 5 -5)
(square M N O P 30)

```

et le résultat est ici (voir FIG 9).

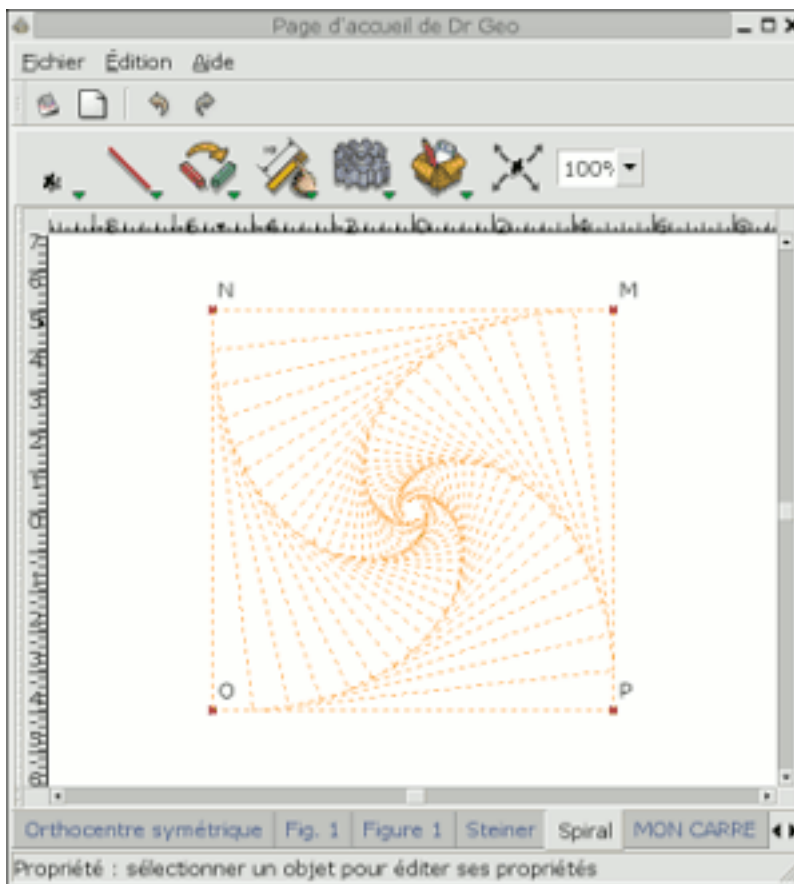


FIG. 9 – Script récursif - Réalisé avec Dr Geo

On voit tout de suite à quel type de public on s'adresse lorsqu'on présente ce type de fonctionnalité. Dans l'interface de Dr Geo, l'implémentation des scripts ne se fait pas remarquer et la priorité reste à la construction géométrique.

*« les script ont une approche numérique mais aussi et surtout nous pouvons les utiliser dans un esprit de bidouillage (« hacking » en anglais). » - Hilaire Fernandes, créateur de Dr Geo*

S'il s'agit d'un complément destiné entre autre à faire plaisir aux bidouilleurs, le programmeur que je suis est ravi. Le non-sens didactique serait par contre que l'on impose toute ces choses à des élèves lorsqu'il s'agit de construire un cercle circonscrit...

## 2 CaRMetal, un logiciel de géométrie dynamique

René Grothmann, professeur de mathématiques à l'université d'Eichstätt (Allemagne), a programmé le logiciel C.a.R. (Compass and Ruler) autour d'algorithmes puissants et fiables qui permettent à son logiciel d'élaborer des constructions géométriques très complexes. Du point de vue de l'interface, René a aussi fait un énorme travail concernant l'interaction propre à la figure (réponses anticipées systématiques, changement de curseurs, etc...). Toutes les réactions du logiciel décrites dans le paragraphe 1.2. sont par exemples dues exclusivement au travail de René.

L'auteur de cet article (qui va dire « je » à partir de maintenant !) a utilisé Cabri - de manière plutôt intensive - pendant quinze ans, avant de tomber sous le charme du logiciel C.a.R., il y a trois ans. J'ai découvert petit à petit qu'il y avait dans C.a.R. beaucoup de ce que j'attends d'un logiciel de géométrie dynamique, et même bien plus. En particulier, C.a.R. est doté d'une gestion très fine et pertinente de l'outil macro-construction (voir partie 2.2).

Si l'interaction propre à la figure m'a semblé particulièrement soignée dans C.a.R., j'ai cependant regretté que certaines fonctionnalités importantes et neuves soient cachées. Pour ne prendre qu'un exemple, l'aspect conditionnel d'un objet s'obtient dans C.a.R. par une combinaison ctrl-alt-clic qui mène à l'ouverture d'une boîte à dialogue... A côté de cela, beaucoup de situations dans ce logiciel demeurent modales (voir paragraphe A.2.) et la création d'une figure peut mener à des manipulations que l'on ne peut pas qualifier de directes.

Comme ce logiciel est écrit en java et distribué sous licence GNU GPL (General Public Licence), j'en ai téléchargé les sources en février 2006 et je me suis lancé dans la reprogrammation complète de son interface graphique. Il ne s'agissait pas dans mon idée d'un simple réhabillage cosmétique de l'application - ce qui en soit n'a que peu d'intérêt - mais bel et bien d'un changement radical dans la manière d'accéder aux fonctionnalités. Mon but était clairement de faire en sorte que l'utilisation du logiciel soit plus conforme à l'idée que je me fais de la manipulation directe.

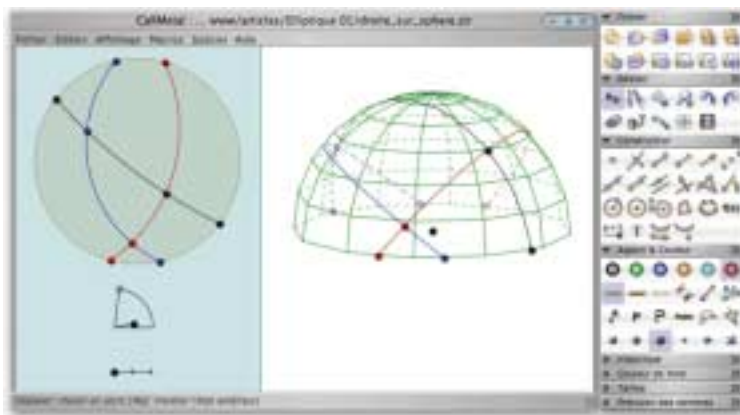


FIG. 10 – Présentation CaRMetal

Ce nouveau logiciel de géométrie dynamique, dérivé de C.a.R., s'appelle CaRMetal(voir FIG.10), est multplateforme, multilingue et est disponible à l'adresse suivante : <http://db-maths.nuxit.net/CaRMetal/>.

## 2.1 La manipulation directe dans CaRMetal

### 2.1.1 Présentation de la palette principale

Plus une application évolue, plus son spectre de fonctionnalités s'élargit, et plus se fait sentir le besoin d'organiser les commandes par thèmes. On pense alors bien évidemment aux menus déroulants. Il reste qu'une fonctionnalité accessible par un article de menu rajoute un degré de complexité - un intermédiaire - qui réduit le sentiment d'aisance pendant la création du document, et qui augmente ainsi quelquefois considérablement le temps d'appropriation du logiciel et de conception du document. Nous avons tous, à un moment ou à un autre, passé du temps dans telle ou telle application à chercher un item de menu dont on sait qu'il existe, mais qui résiste à notre prospection. Le choix a été fait dès le début de la création de CaRMetal de limiter le plus possible la mise en menu des actions, en préférant implémenter un autre type d'interfaçage : les palettes flottantes (voir FIG.11).

La palette principale de CaRMetal contient pratiquement tous les outils du logiciels, regroupés par thèmes dans des zones qui peuvent s'étendre, se rétracter (simple clic sur la barre de titre de la zone), se détacher ou se rattacher (simple clic sur la partie droite du titre de zone). Cette organisation ressemble formellement à celle d'une barre de menus déroulants à un seul niveau de hiérarchie, mais le niveau d'engagement direct n'est toutefois pas le même, et ceci pour plusieurs raisons :

- tous les « menus » de la palette sont déroulés, dévoilant ainsi en permanence l'intégralité de leur contenu. La durée de la phase de recherche d'une commande est alors considérablement réduite.
- La sélection d'un outil ne se fait plus qu'en un seul clic, sans passer par l'étape d'un déroulement de menu.
- Une aide contextuelle est délivrée au survol de chaque icône.
- une palette peut contenir beaucoup d'autres types d'objets interactifs que des icônes. CaRMetal fait par exemple usage de curseurs et de sélecteurs de couleur.

Cette palette est dotée d'un système d'auto-retractation : la palette ne dépasse jamais le bas de l'écran. Si déplier une zone de palette provoque le dépassement du bas d'écran, une autre zone va se refermer automatiquement (par ordre de moindre importance) de façon à ce que la palette ne sorte pas de l'espace visible.

Dans le menu Affichage, on peut aussi changer les dimensions de cette palette principale : trois tailles sont disponibles (grande, moyenne, petite).

### 2.1.2 Les inspecteurs de CaRMetal

Certains logiciels sont accompagnés d'une profusion de boîtes de dialogues attendant de l'utilisateur qu'il entre un texte, un nombre, ou qu'il choisisse telle ou telle option afin de modifier les attributs d'un objet sélectionné. Bien souvent, dans ce cas, le dialogue est bloquant et les modifications ne sont apportées au document que lorsque l'utilisateur valide par « ok ».

Pour éviter cette situation de blocage et supprimer ainsi un intermédiaire inutile, le choix a été fait dans CaRMetal de faire appel à des inspecteurs. La copie d'écran ci-dessus (voir FIG.12) montre l'un d'entre eux : l'inspecteur des propriétés (partie gauche de la fenêtre). La construction peut se faire en gardant ouvert ce panneau, et si on souhaite modifier les attributs d'un objet, il suffit d'agir directement sur les outils de cet inspecteur : les modifications prennent effet immédiatement, sans intermédiaire et sans que l'utilisateur ait à valider quoi que ce soit.

Cela reste vrai pour les entrées clavier : l'exemple ci-dessous montre ce qui se passe lorsqu'on entre une formule dans la rubrique « coordonnées » du point B. L'utilisateur tape « x(A » - sans la parenthèse fermée - dans la rubrique X. Le logiciel reconnaît qu'on veut forcer les coordonnées et coche de lui-même la case « Fixe » dès l'entrée de la lettre « x » (voir FIG.13).

La formule « x(A » n'est pas reconnue, la figure n'est pas modifiée.

Dès qu'on ferme la parenthèse, la formule « x(A) » étant valide, la figure est modifiée (voir FIG.14).



FIG. 11 – Palettes flottantes

## 2.2 La gestion des macro-constructions

Les macro-constructions (macros) sont aux logiciels de géométrie dynamique ce que la procédure -ou méthode - est à la programmation. Créer une macro revient à utiliser les fonctionnalités du logiciel, anciennes macros comprises, pour construire un nouvel outil. L'utilisateur façonne ainsi petit à petit son propre monde à partir des outils géométriques disponibles, ce qui correspond d'un point de vue cognitif à ce que fait l'apprenant en mathématiques lorsqu'il se fabrique au fil du temps des connaissances de plus en plus évoluées à partir des anciennes.

Dans le logiciel d'origine C.a.R., René Grothmann a beaucoup travaillé - et soigné - la question des macros. Outre toutes les caractéristiques propres à l'outil macro construction que l'on retrouve dans beaucoup

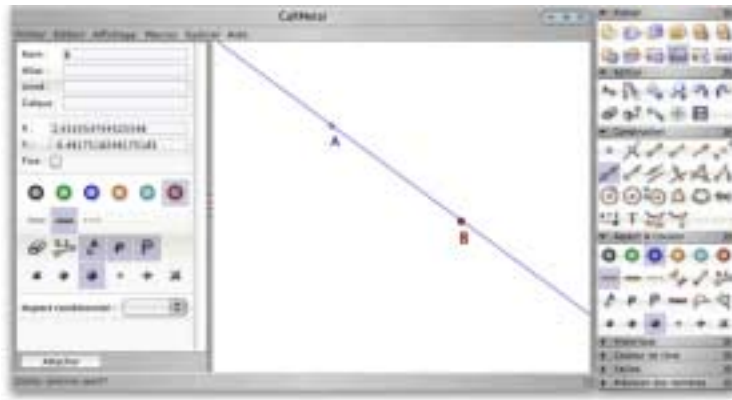


FIG. 12 – Inspecteurs CaRMetal



FIG. 13 – Changement de coordonnées « à la volée »



FIG. 14 – Figure modifiée

de logiciels, C.a.R. propose la possibilité (voir FIG.15) :

- de créer des macros à dialogues qui, lors de leur application, attendent des nombres ou des formules.
- de ne pas déclarer d'objets finaux pendant la création. La macro enregistre alors toute la partie de construction qui dépend des initiaux que l'on a désigné, en conservant l'aspect de tous les objets rencontrés.
- de créer des macros en déclarant certains initiaux comme implicites.

La liste des originalités est loin d'être épuisée, puisque les macros de C.a.R. acceptent toutes les finesses du logiciel. Prenons deux exemples, pour situer :

- On peut créer dans C.a.R. des objets auto-référents, dont les coordonnées sont exprimées en fonction d'elles-mêmes. Cela permet par exemple, en faisant un test, de créer une macro « Point sur disque » qui contraint un point M à rester intérieur à un cercle.

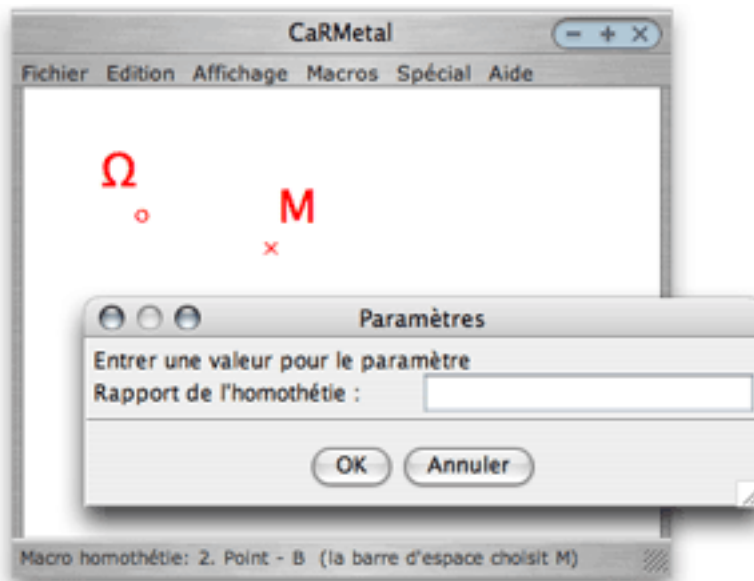


FIG. 15 – Figure modifiée

- La référence circulaire est acceptée dans C.a.R., ce qui signifie par exemple que les coordonnées d'un point A peuvent se référer à celles d'un point C, et dans le même temps, les coordonnées de C se référer à celles de A. On peut ainsi construire une macro « symétrie centrale » telle que l'on puisse déplacer indifféremment le point M ou son image M' (avec bien sûr un résultat dynamique cohérent).

Pour reprendre la citation d'Alan Kay (§ A.3.), René s'est largement occupé pour les macros de « rendre possible les choses complexes ». Tout cet aspect-là se retrouve dans CaRMetal, et, voulant aussi « rendre simple les choses simple », j'ai implémenté un gestionnaire en manipulation directe dédié à cet outil puissant.

J'ai utilisé les macros, pour mes constructions de figures, de manière assez intensives dans Cabri, puis dans C.a.R.. Se faisant, je me suis souvent trouvé dans une situation où j'avais besoin d'une macro spécifique, que je savais avoir déjà créé par le passé, et que je devais aller rechercher sur mon disque dur. Cette manipulation est lourde et ennuyeuse même dans le cas où on connaît l'emplacement exact du fichier, ce qui pour ma part n'est pas tout à fait le cas général...

Pour régler ce problème d'ergonomie, CaRMetal fait la distinction entre deux types de macros. Les macros de bibliothèques sont présentes dans le logiciel dès le lancement, et les macros de fichiers sont celles qui sont attachées à une figure spécifique. Lorsqu'il crée une macro, l'utilisateur décide de la maintenir dans le logiciel (macro de bibliothèque) ou de ne l'attacher qu'au fichier en cours (macro de fichiers).

Dès lors qu'on décide de maintenir un bon nombre de macros dans le logiciel, il faut aussi qu'on puisse les organiser de manière claire et hiérarchisée. C'est une des fonctions de l'inspecteur de macros, qui permet, comme dans les gestionnaires de signets des navigateurs web, de créer des dossiers, de classer ses items, et de changer la structure par simple glisser-déplacer. A chaque modification de cette structure, le menu des macros s'actualise (figure ci-contre).

A tout moment, l'utilisateur peut changer le type de macros par le biais d'un menu contextuel : on peut par exemple en une passe « ajouter à la bibliothèque » une macro, une sélection multiple de macros, ou tout un dossier. Dans la hiérarchie de l'inspecteur, les deux types - bibliothèque et fichier - se distinguent visuellement par des icônes très différentes, et lorsqu'on modifie le type d'une macro, cet élément graphique change immédiatement (voir FIG.16).

La création des macros se fait en deux étapes - désignation des initiaux puis des finaux - à l'issue desquelles une « macro sans titre » vient se glisser dans la hiérarchie du gestionnaire, attendant que l'utilisateur entre

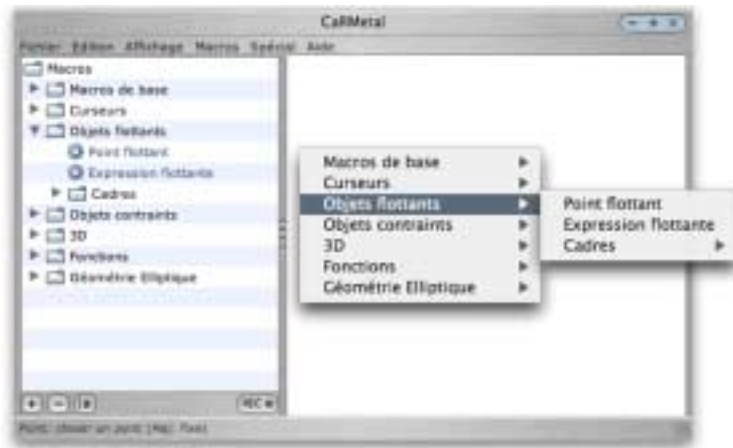


FIG. 16 – Création de macros

un nom au clavier. Tout se passe comme lorsqu'on demande de créer un « nouveau dossier » sur le bureau par menu contextuel, de manière immédiate et sans aucune boîte à dialogue.

### 3 Expérimentation

J'ai pu expérimenter ce mécanisme récemment avec deux classes de cinquième, et je peux témoigner - certes le « témoin » peut-être jugé peu objectif! - du fait qu'ils ont pu réaliser en une seule séance un jeu de macros complet sur le thème de la symétrie centrale, sans véritable soucis liés à l'informatique. Aucun d'entre eux ne connaissait CaRMetal, et je me suis contenté de passer dix minutes au vidéo-projecteur avant qu'ils se mettent au travail, afin de « dégrossir » le terrain.

Bien entendu, les « experts » peuvent modifier les propriétés d'une macro par le menu contextuel, une fois que celle-ci est construite. Une palette flottante apparaît alors, permettant par exemple d'entrer une aide, de changer le nom des constituants, de transformer la macro pour qu'elle fasse apparaître un dialogue, de fixer des initiaux, etc...

Ce dernier point illustre aussi l'esprit dans lequel j'ai essayé de programmer CaRMetal. Lorsque j'ai attaqué ce chantier en février 2006, mon objectif était de mettre en valeur, par application des principes de manipulation directe, les fonctionnalités les plus simples et les plus utilisées. Cette mise en valeur passait selon moi par un masquage de certaines fonctionnalités avancées du logiciel : celles-ci sont toujours là, à la disposition des experts, mais de façon suffisamment discrète pour qu'elles ne parasitent pas l'apprentissage du débutant.

Il n'est sûrement pas d'usage de faire plusieurs fois la même citation dans un même article... Mais pour terminer, je ne peux résister - et j'espère que le lecteur ne m'en tiendra pas trop rigueur - à l'envie de reprendre, comme un refrain dédié à une certaine conception de l'informatique, la phrase d'Alan Kay :

*« Simple things should be simple, complex things should be possible »*<sup>7</sup>

<sup>7</sup>Lire aussi : <http://revue.sesamath.net/spip.php?breve=3>