ALGORITHMIQUE ET INTEGRATION DES OUTILS

Yves MARTIN
Irem de La Réunion

Les premiers mois de pratique algorithmique en seconde font ressortir l'utilisation principale de trois outils : la calculatrice, Algobox et Scratch. La calculatrice présente deux avantages : c'est d'abord un outil générique. Ensuite la programmation, sur certaines d'entre elles, est proche du pseudocode. L'utilisation massive d'Algobox pour une initiation à l'algorithmique en seconde vient de sa prise en main immédiate, même si, clairement, on atteint vite des limites pour les classes ultérieures. Scratch, avec sa structure, ses blocs détachables, reste au contraire toujours utilisable après une phase d'initiation en seconde. La question se pose quand même de savoir s'il est bien respectueux de la vivacité de nos élèves de première et Terminale de continuer à leur proposer un outil fait à l'origine pour les élèves du collège. Quoiqu'il en soit, l'utilisation d'Algobox et même de Scratch, s'ils remplissent bien le cahier des charges d'initiation à l'algorithmique et la programmation, ne préparent pas vraiment une bonne partie de nos élèves à ce qu'ils feront après le lycée.

Dans le cadre de cette réflexion sur une pratique efficace de la programmation, des collègues proposent d'autres voies. Dans le numéro 16 de MathémaTICE, on peut lire l'argument de Jean Pierre Branchard [1] pour utiliser le JavaScript, logiciel web 2.0 souvent présent dans l'interface des outils numériques de nos élèves, ou même, un peu plus en amont, et avec des arguments pertinents, Guillaume Connan pose

1 http://revue.sesamath.net/spip.php?article225

la question de l'utilisation «aussi au lycée», des langages fonctionnels [2], les seuls véritablement « proches des processus mathématiques ».

Plus modestement, mais dans ce même souci de l'efficacité d'apprentissage sur le long terme, et éventuellement même depuis la seconde, nous proposons dans cet article un panorama des possibilités d'utilisation de l'éditeur de javascript du logiciel de géométrie dynamique (GD) CaRMetal. C'est un éditeur javascript performant, et qui, en plus, est parfaitement intégré à un logiciel de géométrie dynamique. Outre le gain de temps d'apprentissage d'interface que l'on imagine, cette situation permet des approches pédagogiques d'un nouveau genre, en phase avec les programmes sur l'algorithmique et géométrie repérée par exemple. Nous allons voir aussi que des activités mathématiques classiques, dans cet environnement, deviennent d'une efficacité surprenante.

La naissance de CaRMetal : une belle histoire du libre

S'il est peu connu, ce logiciel n'est pas totalement inconnu puisqu'il est l'un des logiciels officiellement recommandés pour l'agrégation interne de mathématiques. Il y a bien des façons de présenter un logiciel, on pourrait rentrer par ses fonctionnalités, par ses pertinences propres, ou encore ses différences d'avec ce qui est généralement utilisé. Pour une fois, je vais commencer par son histoire, parce que c'est une belle histoire du (vrai) logiciel libre.

Il était une fois, dans le pays de Cinderella, un autre logiciel de géométrie dynamique, avec un cœur mathématique et informatique tellement extraordinaire qu'il avait conquis durablement le cœur et la raison d'un professeur de mathématique français. Le logiciel s'appelait Compas and Rulers, CaR [3], écrit René Grothmann. Et il faisait des choses tellement peu ordinaires que le professeur a essayé, pendant plusieurs années, de le faire connaître à ses collègues. Mais les séances de formation se terminaient toujours de la même façon : " il est puissant ce logiciel, mais son interface est telle que je n'oserais jamais le proposer à mes élèves ". Et c'est vrai, CaR n'était pas du tout ergonomique, ce qui suffisait pour l'écarter du lot des logiciels utilisables, même à cette lointaine époque où Geogebra n'existait pas encore.

Un jour, Eric Hakenholz, notre professeur de mathématique, par passion pour ce logiciel, et pour le faire connaître en France, proposa à l'auteur de CaR de reprendre l'interface. Pendant deux ans, il s'en tint à améliorer l'interface, gommant les aspects modaux, améliorant la lisibilité des nuances touchant le concept de macro constructions, rendant ensuite lisible toute la partie conditionnelle intrinsèque au logiciel, tout ceci en n'intervenant qu'au minimum sur le cœur du logiciel. Mais au fur et à mesure des évolutions de son interface, et de celles de CaR, les mises à jour de l'interface devenaient de plus en plus complexes à intégrer aux évolutions du logiciel. Les demandes extérieures (multifenêtrage, etc.) ont fait qu'un jour, puisque le code était libre, le futur auteur des Carscripts commença à développer sa propre variante. Il proposa le

2 Article dans le numéro 15 :
http://revue.sesamath.net/spip.php?article216

3 http://mathsrv.ku-eichstaett.de/MGF/homes/grothmann/java/zirkel/doc_en/

nom de CaRMetal pour se référer au projet initial : modifier puis enrichir l'interface d'un logiciel par ailleurs remarquablement construit. Avec le temps, CaRMetal a proposé de nouvelles fonctionnalités, inédites en géométrie dynamique, et s'est éloigné de CaR, même dans son comportement interne, jusqu'à la version actuelle qui propose une intégration bien particulière du javascript. Et l'histoire se poursuit, certains programmeurs proposent déjà de contribuer ponctuellement à l'avancée du logiciel. Le monde du libre permet cela très simplement par la mise en place de projets collaboratifs au sein desquels plusieurs développeurs peuvent travailler en parallèle sur certaines parties de code. C'est pour inciter et favoriser ce travail en commun qu'un projet a été mis en place sur la plate-forme sourceforge [4].

Deux premières fonctionnalités de CaRMetal

Deux particularités du logiciel ont fait qu'il a été retenu dans notre académie pour la formation initiale des enseignants (secondaire et primaire) :

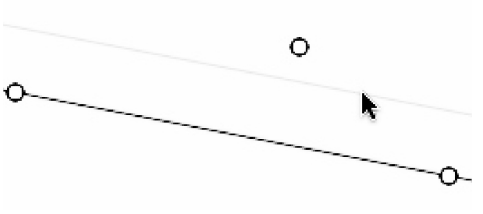
a) *l'absence de modalité :*

Les différents outils et inspecteurs d'objets sont hors du champ de la fenêtre principale et surtout disponibles de manière non modale : non seulement les élèves sont bien devant leur figure et pas le quart de la séance (au moins) devant une boite à dialogue en train de faire un réglage, mais en plus tout est disponible et réagit, bien sûr, en temps réel.

4 http://sourceforge.net/projects/carmetal/

b) *l'anticipation de tous les objets :*

Dans tous les logiciels de GD, les objets créés le sont en anticipation, au fil de la souris (principe dit du « fil à la patte » – penser à la création d'un cercle). C'est à cause de cela que l'on distingue parfois les *items de création* (totalement en engagement direct) des *items de construction* pour lesquels les objets initiaux doivent être préalablement tous désignés pour que l'objet puisse être construit. CaR (et donc CaRMetal qui en a hérité) a une autre approche. Tous les objets sont préconstruits dès qu'il ne reste plus qu'un point à donner. Par exemple, si un élève veut construire la parallèle à une droite passant par un point, en montrant la droite, une parallèle se crée au bout de la souris jusqu'à ce que l'élève clique sur un point (ou en crée un en cliquant à l'écran).



Cette différence est, à elle seule, suffisamment significative en termes cognitifs pour que des formateurs de professeurs des écoles choisissent ce logiciel pour les formations initiales et continues premier degré de notre académie.

Peu à peu, dans des stages départementaux, les maitres formateurs ont appris la géométrie dynamique avec CaRMetal car ce comportement accompagne l'apprentissage. En effet, cette anticipation permet la manipulation des objets non encore finali-

sés dans leurs constructions. Le comportement du logiciel étant conforme aux propriétés géométriques en jeu, l'anticipation participe à installer, par la manipulation directe, de très bonnes représentation des propriétés des objets engagés, voire même à cristalliser rapidement la conceptualisation de ces propriétés.

Alors que l'engagement direct, par exemple du logiciel historique Cabri-géomètre, reconnaissait l'utilisateur comme sujet connaissant, cette nouvelle fonctionnalité le reconnaît en plus comme sujet apprenant, et accompagne ainsi son apprentissage.

A un niveau plus conceptuel, au lycée par exemple, cette anticipation, qui fonctionne avec tous les outils y compris les macros constructions, permet des investigations tout à fait différentes car on peut tester des conjectures sans véritablement finaliser la figure liée à cette conjecture. L'organisation de l'investigation peut être envisagée de manière plus fine, tout en finalisant quand même les figures pour les élèves qui en ont besoin.

c) *analyse de l'absence de cette fonctionnalité dans les autres logiciels*

Il est intéressant de voir pourquoi les autres logiciels sont passés à côté de cette fonctionnalité d'accompagnement cognitif, soit de l'apprentissage, soit de l'investigation. On s'intéressera à Cabri-géomètre car ce sont les travaux et réflexions didactiques sur ce logiciel qui ont ouvert les portes à l'ergonomie que l'on connaît des logiciels de GD.

Les plus anciens se souviennent des premières versions du logiciel. Pour tracer une parallèle il fallait montrer dans un ordre précis, par exemple, la droite et le point. L'item attendait l'un puis l'autre, et comme il n'y avait pas — dans les premières versions — cet engagement direct qui, maintenant par exemple surligne l'objet qui peut être saisi, et que les items avait un fonctionnement fortement modal, plusieurs personnes ont remarqué que cela serait plus simple, pour les élèves de collège en particulier si, quand les objets sont de types différents, on pouvait montrer les objets dans un ordre quelconque. Malgré la difficulté de la tâche, ce fut implémenté, et salué par chacun de nous comme un progrès de l'engagement direct sur les items de construction. Bien sûr Cabri géomètre étant l'étalon de référence en géométrie dynamique, les logiciels suivant se sont généralement contentés de reproduire ce fonctionnement.

Dans CaRMetal il n'y a pas l'indépendance des objets de construction d'un item, mais il y a cette règle simple que tout est construit autour de l'anticipation, donc on sait que le dernier objet, quand cela a du sens, est un point. Par ailleurs, ce manque d'indépendance est largement compensé par la réaction immédiate du surlignage de l'objet quand il peut être sélectionné (comme avec Geogebra). Il ressort de cette analyse que c'est le manque de début d'engagement direct (le surlignage), et la forte modalité interne des items de construction des premières versions qui ont fait prendre aux réflexions didactiques — puis aux mise en œuvre d'interface — des chemins qui, s'ils furent significatifs à l'époque de la mise en œuvre de la GD, n'était en définitive pas si optimaux que cela.

Quelques autres fonctionnalités significatives

De nombreuses autres possibilités du logiciel seraient aussi intéressantes à analy-

ser, mais ce n'est pas l'objet de cet article. Les aspects conditionnels, la gestion des macro constructions [5], une aimantation extraordinaire ou une certaine disponibilité de la 3D mériteraient [6], chacun, un article complet.

Signalons plus précisément, avec des références de développement, deux points techniques importants :

— la possibilité d'enrouler complètement une droite sur un cercle [7] – et donc de compter les tours sur un cercle. Cela permet de construire les fonctions trigonométriques sur **R**.

— la récursivité des objets : que ce soient des points ou des expressions, ils peuvent être récursifs [8] et même mutuellement récursifs en manipulation directe, ce qui ouvre un champ de possibilités énorme, comme en simulation statistique [9], et même la prise en compte de l'histoire de la manipulation de l'utilisateur au sein même d'une figure.

Comme on le voit, CaRMetal est un logiciel de géométrie dynamique de nouvelle génération. Mais ceci est sans tenir compte de la nouvelle possibilité implémentée par Eric Hakenholz depuis la version 3.0, l'intégration de JavaScript (JS) dans un éditeur performant, adapté à une utilisation scolaire.

5 Celles implicitement fonctionnelles s'appliquent automatiquement non pas à un point comme elles sont définies mais aussi aux objets reconnus par le logiciel lors de la manipulation directe.
6 Pour la 3D, voir les diaporamas dynamiques du site du logiciel
7 Uniformisation du cercle :
http://www.reunion.iufm.fr/recherche/irem/spip.php?article50
8 Références circulaires :
http://www.reunion.iufm.fr/recherche/irem/spip.php?article40
9 Les aiguilles de Buffon :
http://www.reunion.iufm.fr/recherche/irem/spip.php?article98

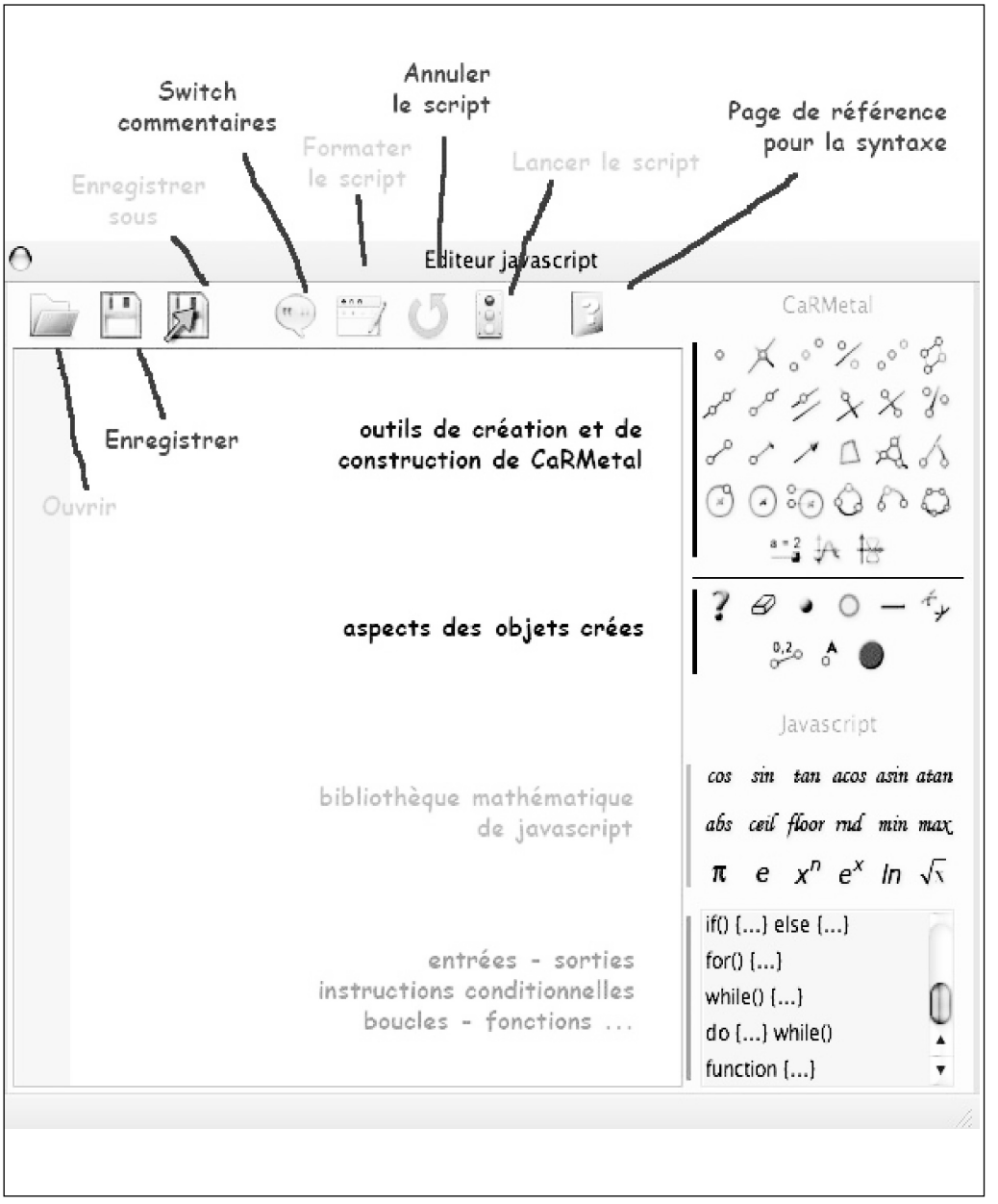
Le choix du JavaScript comme langage standard disponible pour le web

Pour un logiciel qui a une exportation en applet et une utilisation importante sur la toile, le choix de JS est assez évident. C'est actuellement un langage incontournable du web 2.0 à tel point que sa vitesse d'exécution devient un enjeu pour les navigateurs. Pour nos élèves, loin des boutons d'Algobox, et du fluo « kids » de Scratch, c'est d'abord un *vrai* langage, moderne, celui qui interface nombre de leurs outils numériques.

Avec JS nous ne sommes pas obligé de déclarer les variables. C'est aussi la pratique courante des langages sur calculatrice, et des logiciels de mathématiques professionnels comme Maple ou Mathematica. Pourquoi apprendre à nos élèves, en cours de mathématiques, le contraire de notre propre pratique mathématique ? Et même si on veut y voir une logique informatique — le seul argument — d'un point de vue mathématique il ne tient pas vraiment car les logiciels les plus "mathématiques" sont les logiciels fonctionnels (du LISP si cher à Chaïtin au plus récent Haskell) et ils sont justement très peu typés, et le moins typé possible.

L'éditeur javacript de CaRMetal

Il a été fait pour une utilisation rapide et efficace : coloration syntaxique, icones du logiciel et boutons de fonctions mathématiques et instructions JS prédéfinies. Dans tous les cas l'item cliqué apparaît comme déjà opérationnel. On ne fait que modifier une syntaxe préalablement correctement rédigée.


Switch commentaires
Annuler le script
Page de référence pour la syntaxe
Formater le script
Lancer le script
Enregistrer sous
Editeur javascript
CaRMetal
Enregistrer
outils de création et de construction de CaRMetal
Ouvrir
aspects des objets crées
Javascript
bibliothèque mathématique de javascript
cos sin tan acos asin atan
abs ceil floor rnd min max
π e x^n e^x ln √
entrées - sorties
instructions conditionnelles
boucles - fonctions ...
if() [...] else [...]
for() [...]
while() [...]
do [...] while()
function [...]

On remarquera les deux parties de la palette d'outils : une partie orientée géométrie, et une partie orientée programmation en JS.

C'est parce que CaRMetal propose une subtile intégration de ces deux mondes que nous allons être assez rapidement devant quelque chose de nouveau, un peu comme les plus anciens d'entre nous ont pu l'être devant leurs premiers écrans Visicalc.

Utilisation en JavaScript standard

Bien entendu l'éditeur dispose de sa propre sortie sous forme de console texte pour les premiers apprentissages. C'est dans cet environnement qu'on fait généralement les premiers TP sur les entrées et les sorties. Alain Busser, de l'Irem de La Réunion propose les scripts de ses séances en seconde sous forme de narration de recherche [10]. D'autres collègues, sur le forum des Carscripts proposent des simulations statistiques [11] soit en sortie texte, soit en sortie dans la fenêtre graphique. Même si cet aspect est incontournable dans le cadre d'un travail en classe, dans cet article nous allons plutôt nous centrer sur les spécificités de l'intégration du JS dans un logiciel de géométrie dynamique.

La suite de l'article présente à grand trait les possibilités de l'outil, sur les trois années du lycée, et n'essaie pas de construire une progression d'apprentissage. On se reportera aux liens en fin d'article pour aller plus dans le détail des fonctionnalités et de l'apprentissage.

Aspects fonctionnel et procédural des items géométriques

Voici un premier script [12] qui construit un carré ABCD implicitement de centre l'origine du repère : on utilise seulement les propriétés des diagonales (milieux et orthogonalité) :

```
1 Point("A",3,4);
2 Point("C","-x(A)","-y(A)");
3 Point("B","-y(A)","x(A)");
4 Point("D","-x(B)","-y(B)");
5 Segment("A","B");Segment("B","C");
6 Segment("C","D");Segment("D","A");
7 SetShowName("A,B,C,D",true);
```

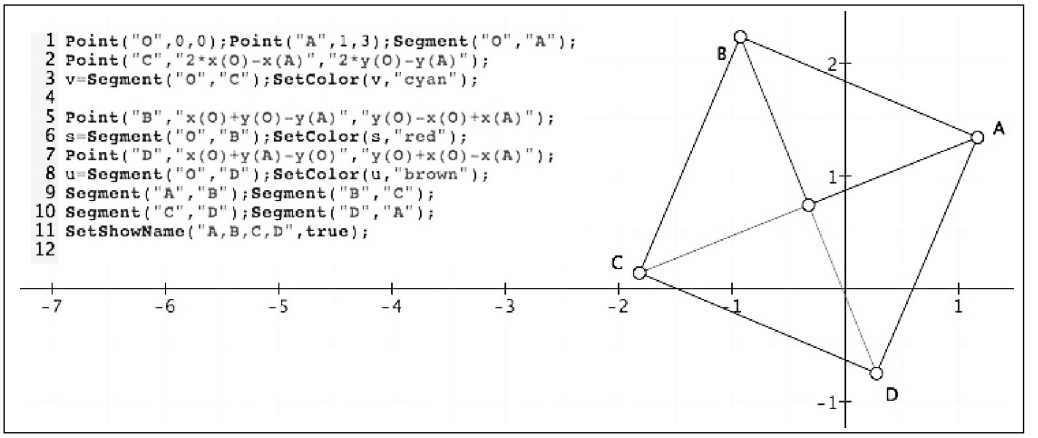
Les coordonnées numériques attribuées à A ne sont qu'une initialisation. On peut ensuite agir sur A et le carré est dynamique. Partant d'un script comme celui-ci, on peut envisager un travail de *géométrie repérée dynamique* et chercher à construire un carré dont on peut déplacer le centre O et un sommet A.

A la page suivante, on voit un carré avec son centre initialisé à l'origine du repère, puis déplacé en manipulation directe. L'initialisation à l'origine du repère permet de mettre en évidence les différentes représentations erronées des élèves quand on va déplacer O. La constructions des segments d'extrémité O est là pour vérifier visuellement l'orthogonalité des diagonales. En effet, avec le centre à l'origine, plusieurs calculs erronés peuvent faire illusion, c'est intéressant de les mettre en évidence en remarquant que pour O à l'origine, les formules utilisées peuvent

10 http://www.reunion.iufm.fr/recherche/irem/spip.php?rubrique81
11 http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=300

12 En réalité il y a une coloration syntaxique, ce qui rend le texte plus lisible qu'ici.

```
1 Point("O",0,0);Point("A",1,3);Segment("O","A");
2 Point("C","2*x(O)-x(A)","2*y(O)-y(A)");
3 v=Segment("O","C");SetColor(v,"cyan");
4
5 Point("B","x(O)+y(O)-y(A)","y(O)-x(O)+x(A)");
6 s=Segment("O","B");SetColor(s,"red");
7 Point("D","x(O)+y(A)-y(O)","y(O)+x(O)-x(A)");
8 u=Segment("O","D");SetColor(u,"brown");
9 Segment("A","B");Segment("B","C");
10 Segment("C","D");Segment("D","A");
11 SetShowName("A,B,C,D",true);
12
```



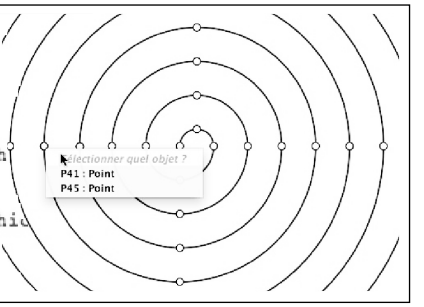
être valides mais plus dès que l'on déplace O. C'est une façon d'utiliser la manipulation directe associée à la géométrie repérée en fin de processus, pour la validation. D'où cette expression de *géométrie repérée dynamique*, comme nouvelle possibilité d'apprentissage et nouvel outil de vérification.

Les items JavaScript de géométrie peuvent être procéduraux ou fonctionnels. On remarquera que les segments d'extrémité O sont utilisés sous forme fonctionnelle alors que les points sont sous forme procédurale. Le nommage des points n'est pas lié à cette distinction, mais si on veut utiliser à nouveau un point dans un script sans le nommer sur la figure, on peut utiliser l'aspect fonctionnel. Cette souplesse peut être utile dans des itérations.

Par exemple ci-dessous on a dessiné une spirale uniquement de manière fonctionnelle, ce qui est particulièrement efficace. On voit à droite que les points sont nommés par défaut par le logiciel.

On remarquera que c'est bien un *dessin* car, donnant des coordonnées numériques, les points sont initialisés à la position attendue mais non reliés entre eux dynamique-

```
1 // hg hd pour horizontal gauche et droit
2 // vh et vg pour vertical haut et bas
3 for (i=0; i<10; i=i+1){
4    hg1=Point(-2*i-1,0);
5    vh=Point(0,2*i+1);
6    hd=Point(2*i+1,0);
7    arch=Arc3pts(hg1,vh,hd);SetThickness(arch,"th
8    vb=Point(-1,-2*i-2);
9    hg2=Point(-2*i-3,0);
10   arcb=Arc3pts(hd,vb,hg2);SetThickness(arcb,"thic
11 }
12
```



ment. Il faudrait procéder autrement pour cela (géométriquement par exemple).

Sur cet exemple, on commence à percevoir une première subtilité entre le JavaScript et la figure : sauf à faire référence explicitement aux points de la figure, comme dans l'exemple du carré, les scripts construisent des objets libres qu'ils initialisent seulement.

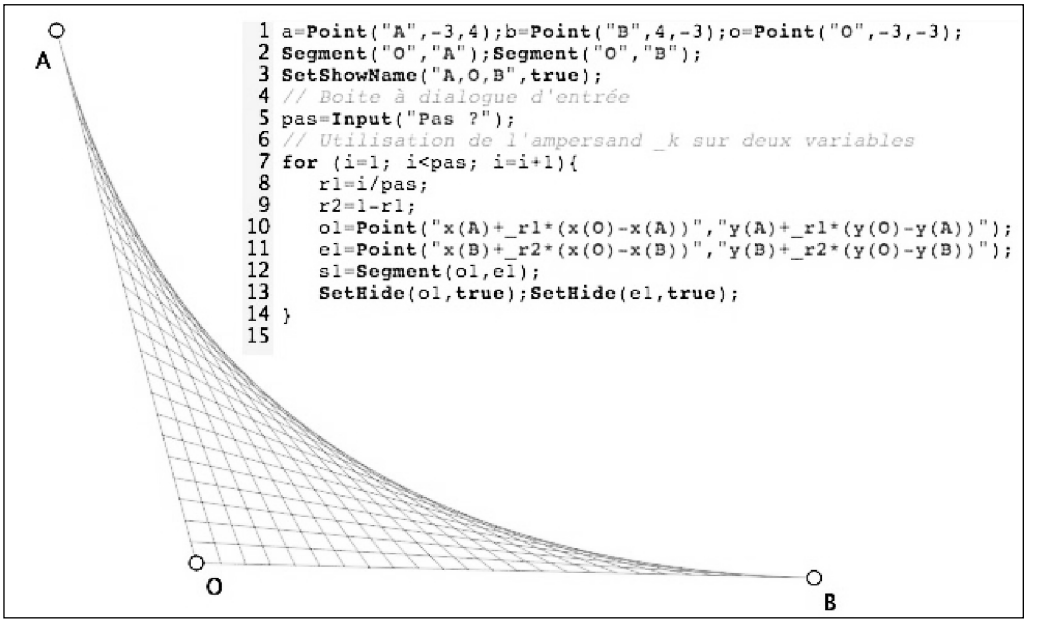
Calculs dynamiques

Bien entendu le JavaScript se comporte, avec ses variables, comme on s'y attend, dans la sortie console ordinaire que ce soit pour de l'arithmétique ou des statistiques. Mais on peut aussi travailler *dynamiquement*, par script, sur la fluctuation des échantillonnages, la méthode d'Euler, ou les suites récurrentes. Pour cela il n'y a qu'une règle à connaître, celle du passage d'une variable javascript à une figure dynamique. Nous avons vu comment on a fait référence aux coordonnées des points pour le carré dynamique. On peut retenir que, **dans un script, les coordonnées dynamiques sont passées entre guillemets à CaRMetal**.

Reste à voir comment, au sein de ces coordonnées dynamiques, on insère une variable JavaScript. **La seule règle à connaître est que l'on passe *le contenu* d'une variable k par la syntaxe _k (underscore k)**.

Sur la figure ci-dessous, initialement les segments [OA] et [OB] sont orthogonaux, un pas est demandé. Puis le tableau de fils est tracé. Et les points ont été déplacés.

La seule spécificité des scripts de CaRMetal est que *le passage* dans une figure *du*




```
1 a=Point("A",-3,4);b=Point("B",4,-3);o=Point("O",-3,-3);
2 Segment("O","A");Segment("O","B");
3 SetShowName("A,O,B",true);
4 // Boite à dialogue d'entrée
5 pas=Input("Pas ?");
6 // Utilisation de l'ampersand _k sur deux variables
7 for (i=1; i<pas; i=i+1){
8    r1=i/pas;
9    r2=1-r1;
10   o1=Point("x(A)+_r1*(x(O)-x(A))","y(A)+_r1*(y(O)-y(A))");
11   e1=Point("x(B)+_r2*(x(O)-x(B))","y(B)+_r2*(y(O)-y(B))");
12   s1=Segment(o1,e1);
13   SetHide(o1,true);SetHide(e1,true);
14 }
15
```

contenu d'une variable JavaScript (ci dessus r1) se fait avec la syntaxe _r1 (uniquement parce que CaRMetal a ses propres variables). Le reste est une syntaxe très proche de l'écriture mathématique usuelle. Remarquer que les points ne sont pas nommés, d'où l'usage de l'aspect fonctionnel - o1=Point() - pour cacher les points créés en utilisant le nom de la variable associée.

Applications en analyse

Nous avons insisté deux fois sur le passage de la variable *par contenu*, alors que l'on n'a vu qu'un passage par valeur. Mais parce que ci dessus la variable est numérique. Si une variable est un point a, on passera ses coordonnées naturellement par **x(_a)** et **y(_a)**. Même si cette écriture reste standard, pour plus d'efficacité, l'auteur du logiciel autorise le raccourci **x_a** et **y_a** (lire « x de a et y de a »).

Voyons cela sur une première application de la méthode d'Euler en 1°S pour une approximation d'une primitive. Le script suivant est générique au sens où il suffit de disposer d'un point A et d'une fonction *Laf* pour qu'il s'applique. Ces deux noms sont pris par défaut, comme le seraient des données implicites dans une macro construction par exemple.

Dans l'illustration de la page ci-contre on voit le nom de la fonction, le script et son application. Une fois le script effectué non seulement on peut déplacer A mais *on peut aussi modifier la fonction Laf*, on a donc un script tout à fait dynamique. On observe donc les notations x_m et y_m pour les coordonnées du point courant m et l'utilisation de $\_k$ pour le pas.

Remarques : a) La notation i++ n'est pas standard, par défaut le logiciel écrit i=i+1 pour l'itération.

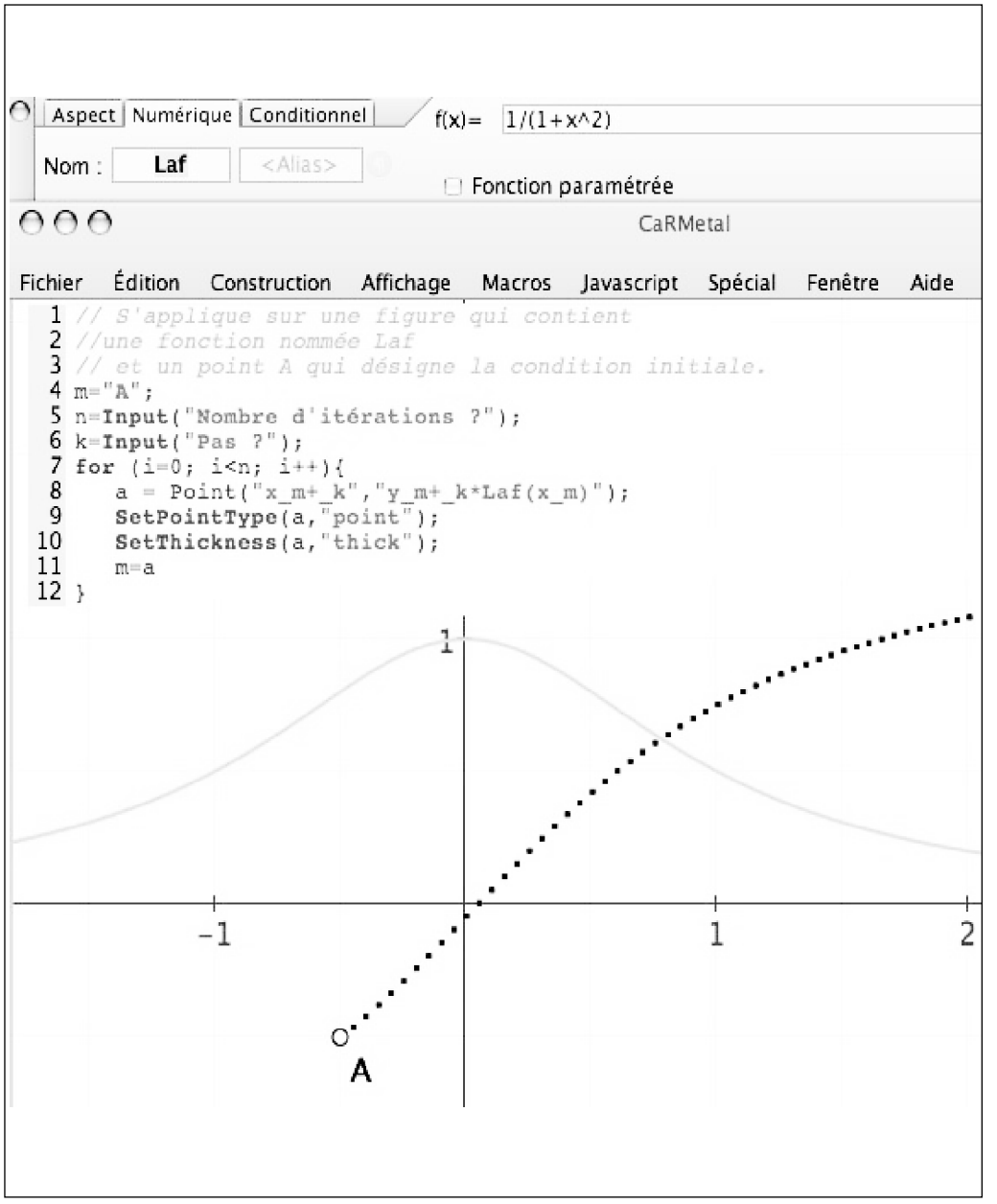
b) Bien entendu l'illustration suivante est un montage, le script n'est pas dans la fenêtre de sortie

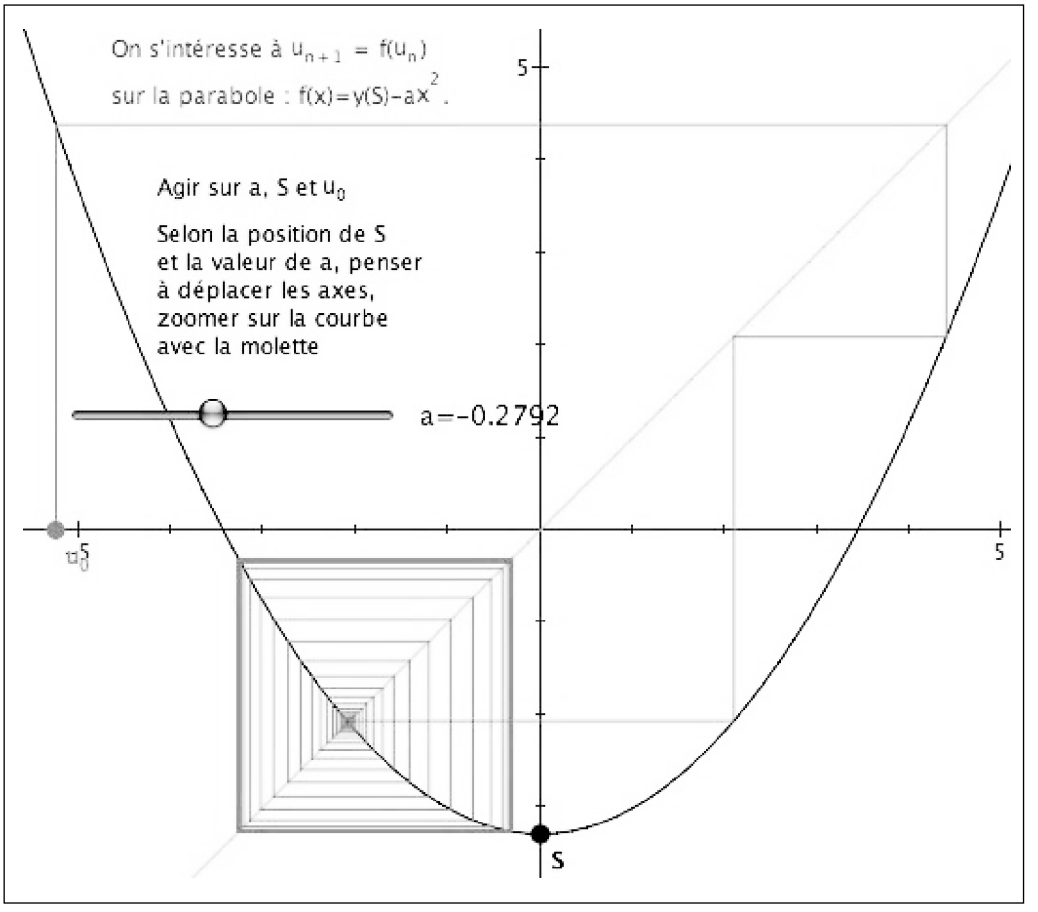
Voici un script sur la recherche d'un point fixe d'une fonction :

```
4 u="u0";
5 a=Point("x_u","Laf(x_u)");
6 SetHide(a,true);
7 Segment(u,a);
8 for (i=0; i<100; i++){
9    b=Point("y_a","y_a");SetHide(b,true);
10   Segment(a,b);
11   c=Point("x_b","Laf(x_b)");SetHide(c,true);
12   Segment(b,c);
13   a=Point("x_c","y_c");SetHide(a,true);
14 }
```

En formation — et a fortiori en classe — il est intéressant de faire ce script en plusieurs étapes. On réalise d'abord la phase d'initialisation (lignes 4 à 7) qui, à partir de *u0* sur l'axe des abscisses, place le point a sur la courbe et construit le premier segment. Ensuite on construit «manuellement» une première itération (les lignes 9 à 12), aussi bien pour des raisons techniques de syntaxe que pour des raisons de conceptualisation. Cette phase générique construite, sa transformation en itération (en particulier la ligne 13) pourra poser problème en classe — ou non — selon ce qui a été fait avant sur le thème de l'itération.

Voir, page suivante, l'application de ce script : La représentation graphique de la fonction *Laf* est ici une parabole modifiable par son sommet S et le paramètre a. Sur l'illustration, l'itération arrive sur un des deux points fixes, et celui-ci est manifestement répulsif (pour cette valeur de a et position de S). On appréciera l'économie d'écriture pour un script produisant une figure aussi dynamique, d'autant que *Laf* peut être n'importe quelle fonction.

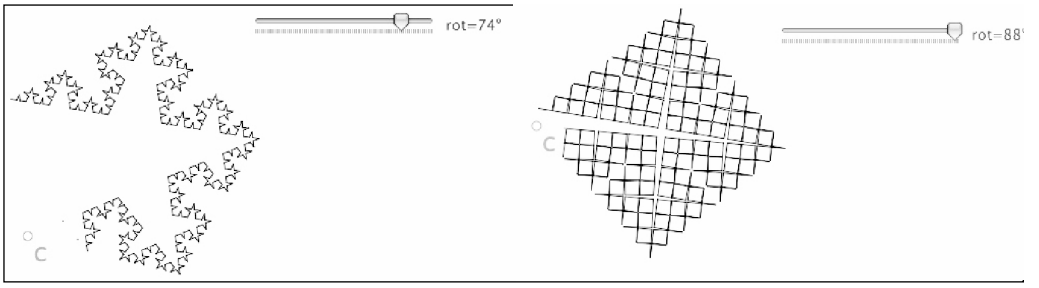

Aspect
Numérique
Conditionnel
f(x)=
1/(1+x^2)
Nom :
Laf
<Alias>
Fonction paramétrée
CaRMetal
Fichier
Édition
Construction
Affichage
Macros
Javascript
Spécial
Fenêtre
Aide
1 // S'applique sur une figure qui contient
2 //une fonction nommée Laf
3 // et un point A qui désigne la condition initiale.
4 m="A";
5 n=Input("Nombre d'itérations ?");
6 k=Input("Pas ?");
7 for (i=0; i<n; i++){
8 a = Point("x_m+_k","y_m+_k*Laf(x_m)");
9 SetPointType(a,"point");
10 SetThickness(a,"thick");
11 m=a
12 }
1
-1
1
2
A



Un exemple de récursivité dynamique

En s'éloignant du programme du lycée le temps d'un paragraphe, nous pouvons illustrer de manière encore plus profonde, l'extraordinaire intrication entre le JavaScript de CaR-Metal et la géométrie dynamique. Bien entendu, comme langage, JS est récursif. Peut-on illustrer une interaction dynamique avec une procédure récursive ? Oui, pendant l'exécution de la procédure si celle-ci peut dépendre globalement d'un paramètre extérieur. Un des exemples les plus simples de cette situation est la courbe de Césaro [13], une extension de la courbe de Koch avec un angle variable. Ci dessous l'exécution d'un script *Cesaro* à l'ordre 4 avec modification de l'angle pendant l'exécution du script : on voit que *la modification*

13 http://www.mathcurve.com/fractals/koch/koch.shtml



du paramètre est accessible non pas seulement après le script, mais *pendant son exécution même*.

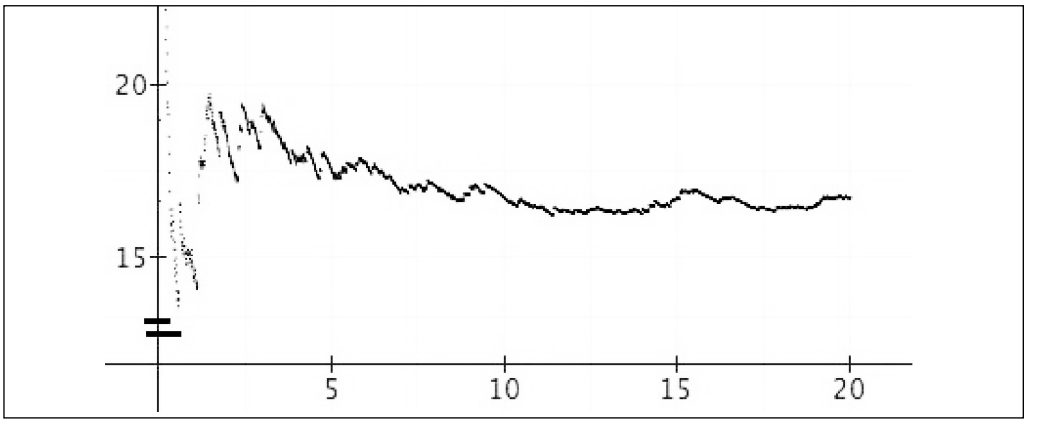
Utilisation en statistique

Dans ces exemples nous avons surtout insisté sur la possibilité d'une construction dynamique par script car c'est bien la nouveauté qu'introduit CaRMetal, mais, dans certains cas il n'est pas utile d'avoir cette sophistication-là et l'on revient à une programmation en JavaScript tout à fait classique, même en sortie graphique, qui produit un résultat ordinaire. C'est par exemple suffisant quand on veut travailler sur les statistiques.

Voici (page ci-contre) un script, proposé sur le forum des Carscripts par Pierre Marc Mazat [14] de Blois autour de l'exercice du document d'accompagnement des programmes sur l'algorithmique :

> *Un joueur lance deux dés, et fait la somme des points obtenus. S'il obtient 8 il gagne 10 €, sinon il perd 1 €.*

Il propose plusieurs simulations [15] et la dernière teste le nombre moyen de parties pour que le joueur gagne. Voici le résultat :



14 Sur son site personnel on peut voir une interaction web2.0, réalisée en javascript, entre une page html et un script produit par CaRMetal : http://pm.mazat.free.fr/exercices/curiosites/sierpinski.php

15 http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=300

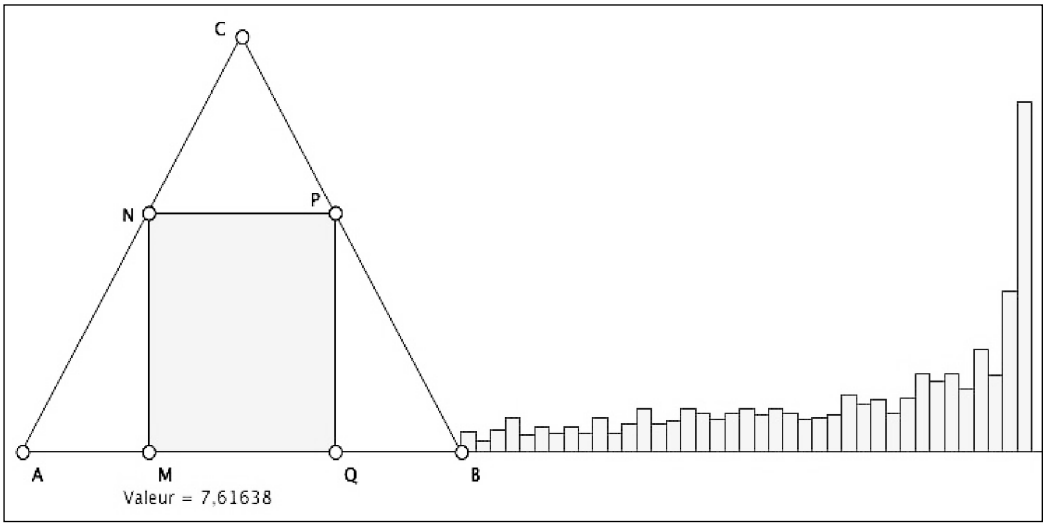
```
//effectue k simulations d'un lancer de deux dés
//pour chaque simulation, comptabilise le nombre de lancer nécessaires pour obtenir 5€
//affichage en temps réel du nombre moyen de lancer nécessaires pour obtenir 5€
//1 unité représente 100 simulations
var k, i, de_1, de_2, compteur_lancer, gain_cumule, nb_lancer_moyen;

k = eval( Input("Nombre de simulations ?\n1 unité représente 100 simulations") ) + 1;
//k=2001;
compteur_lancer = 0;

for (i=1; i<k; i++){
   gain_cumule=0;
   while ( gain_cumule<5 ){
      de_1 = Math.floor( 6*Math.random() + 1 );
      de_2 = Math.floor( 6*Math.random() + 1 );
      if (de_1+de_2 == 8){
         gain_cumule = gain_cumule + 10;
      } else {
         gain_cumule = gain_cumule - 1;
      }
      compteur_lancer++;
   }
   a=Point(i/100,compteur_lancer/i);
   SetPointType(a,"point");
}
```

On voit que, comme dans le cas de la spirale précédente, les variables sont directement utilisées dans les points, car passées simplement comme valeurs numériques d'initialisation.

Dans le même registre, on peut construire des histogrammes en temps réel. Dans le script de droite, Alain Busser (*même adresse que le précédent*) nous propose une méthode générale pour réaliser cela... L'intérêt est dans l'exécution effective du script. Dans la figure du dessous, le point M tiré par le script (et affiché par *Move*) permet de construire un rectangle, l'aire est dans l'expression *E1*, l'histogramme s'affiche au fur et à mesure des tirages par les *Move* des points C et D indicés par le tirage.

Toutefois, on peut aussi aller beaucoup plus loin sur le thème des statistiques et produire, grâce à cette étroite intrication entre la programmation et la géométrie dynamique, des *simulations dynamiques* d'un nouveau genre. Prenons l'exemple classique des coïncidences des anniversaires au sein d'une classe. On peut réaliser une figure et un script qui, appliqué à cette figure réalisera une simulation de tirages continus de 20 échantillons de 100 classes tout en autorisant l'utilisateur à modi-

```
//Programme rectangle
//Ce programme examine la distribution des aires
//Auteur: Alain Busser
//Date: octobre 2009
//Ce programme est libre, sous licence CeCILL:
//http://www.cecill.info/licences/Licence_CeCILL_V2-fr.html
var histogram=new Array();
for(i=0;i<40;i++){//40 rectangles
   histogram[i]=0;//initialisation du tableau
   a=Point("A"+i,i/5,0);SetHide(a,true);
   b=Point("B"+i+1,(i+1)/5,0);SetHide(b,true);
   c=Point("C"+i+1,(i+1)/5,0);SetHide(c,true);
   d=Point("D"+i,i/5,0);SetHide(d,true);
   p=Polygon("_a,_b,_c,_d");SetColor(p,"green");
   a=b;
   d=c;//on passe au rectangle suivant
}
for(i=1;i<=1000;i++){
   Move("M",-6+3*Math.random(),0);
   n=Math.floor(5*GetExpressionValue("E1"));
   histogram[n]++;
   for(j=0;j<40;j++){
      Move("C"+j+1,(j+1)/5,histogram[j]/i*25);
      Move("D"+j,j/5,histogram[j]/i*25);
   }
}
```



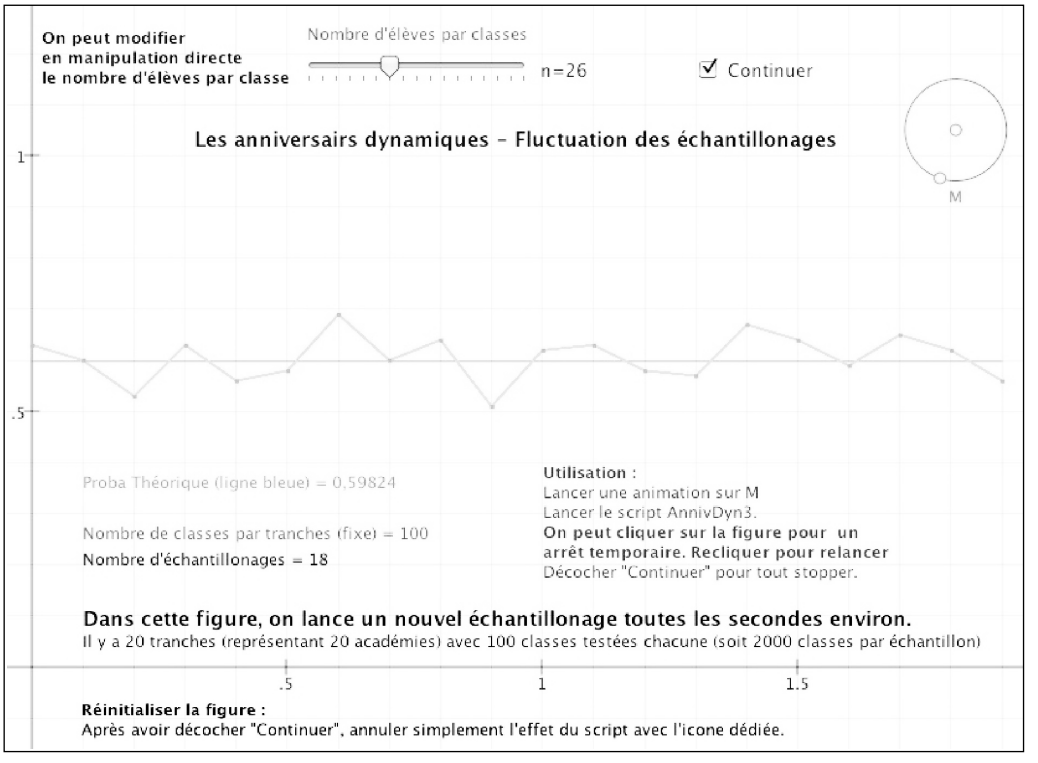
fier en temps réel le nombre d'élèves par classe. Bien sûr ce script, plus délicat, nécessite un traitement particulier.

Plus encore que dans les figures précédentes, la version statique ne rend pas compte de ce que fait cette double communication entre une figure et un script : un nouvel échantillonnage est produit toutes les secondes et l'effectif des classes (de 20 à 36 élèves) est manipulable au curseur pendant l'exécution du script. C'est assez surprenant. On trouvera le principe, les détails de ce scripts et quelques illustrations connexes sur le site de l'Irem de La Réunion [16].

Utilisation dans l'espace

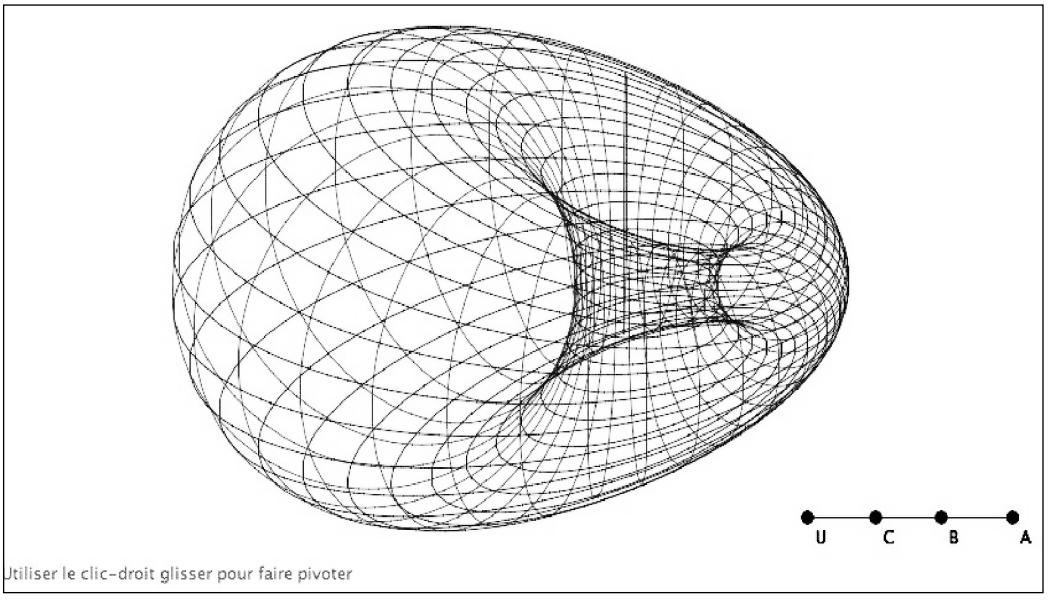
Un mot pour signaler que l'on peut aussi, par script, faire très facilement des figures dans l'espace. C'est à Jérôme Caré [17] (Saint Brieuc) que l'on doit une méthode simple pour réaliser cela. Voici, pour montrer l'efficacité de la démarche, une cyclide de Dupin, manipulable en 3D, dont les trois paramètres fondamentaux sont accessibles par curseurs, la cyclide se modifiant en temps réel.

On comprendra qu'on dessine simplement deux familles de coniques (donc de quintuplets de points) dont les paramètres sont défi-



16 http://www.reunion.iufm.fr/recherche/irem/spip.php?article300

17 http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=298 pour un script 2D de ce collègue. Son script permet de générer une grille quelconque selon un repère dynamique chois par l'utilisateur, avec un point aimanté sur cette grille.



nis par des combinaisons algébriques construites à partir des points U, C, B et A. L'intérêt est que c'est bien plus facile à réaliser par script — avec des boucles imbriquées — que manuellement, bien entendu.

Références

Le site de CaRMetal (logiciel libre GNU multiplateforme) : téléchargement, tutoriaux en flash, diaporamas dynamiques, forums, galerie des utilisateurs …

http://db-maths.nuxit.net/CaRMetal/

L'Irem de la Réunion a un groupe *algorithmique avec les CaRScripts* qui propose un manuel de référence de 56 pages (Alain Busser), des articles de prise en main pour l'enseignant, et de propositions d'utilisations dans lesquels les exemples vus ici sont repris et largement détaillés, avec plusieurs dizaines de scripts directement copiables. On trouve aussi une rubrique de narration de recherche sur les premiers TP d'algorithmique réalisés en classe de seconde, par Alain Busser.

Adresse générale de ce groupe de travail :

http://www.reunion.iufm.fr/recherche/irem/spip.php?rubrique58

Pour une première prise en main des CarScripts :

http://www.reunion.iufm.fr/recherche/irem/spip.php?article186

Détail des techniques d'utilisation dans l'espace

http://www.reunion.iufm.fr/recherche/irem/spip.php?article210

Un article sur l'itération (en 14 onglets, avec de nombreux scripts à copier)

Partie 1 : http://www.reunion.iufm.fr/recherche/irem/spip.php?article232

Partie 2 : http://www.reunion.iufm.fr/recherche/irem/spip.php?article238

D'une manière générale, l'Irem de La Réunion propose de nombreuses pages réalisées avec CaRMetal, que l'on retrouve ici :

http://www.reunion.iufm.fr/recherche/irem/spip.php?page=recherche&recherche=carmetal

Un interview de Eric Hakenholz sur Framablog (octobre 2009) :

http://www.framablog.org/index.php/post/2009/10/21/carmetal-geometrie-dynamique-eric-hakenholz

Cet article a été écrit pour la version 3.1.1. Une réécriture en profondeur du logiciel est en cours (version 3.5), pour une disponibilité fin avril ou début mai 2010. Un article présentant en détail les nouvelles fonctionnalités (dont l'intégration des scripts dans les figures et donc dans les applets) sera alors en ligne à cette adresse :

http://www.reunion.iufm.fr/recherche/irem/spip.php?article347