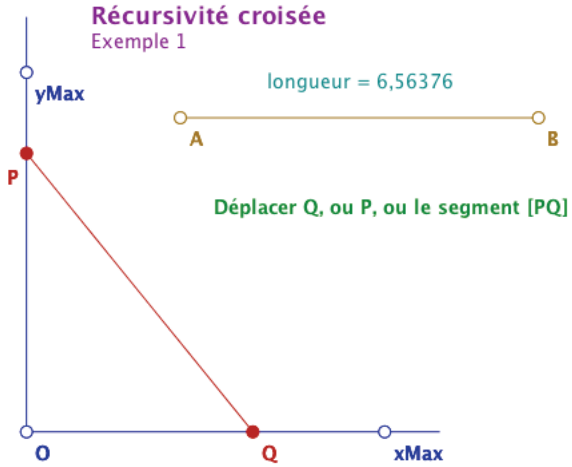
CaRMetal : une géométrie dynamique enrichie

Yves Martin

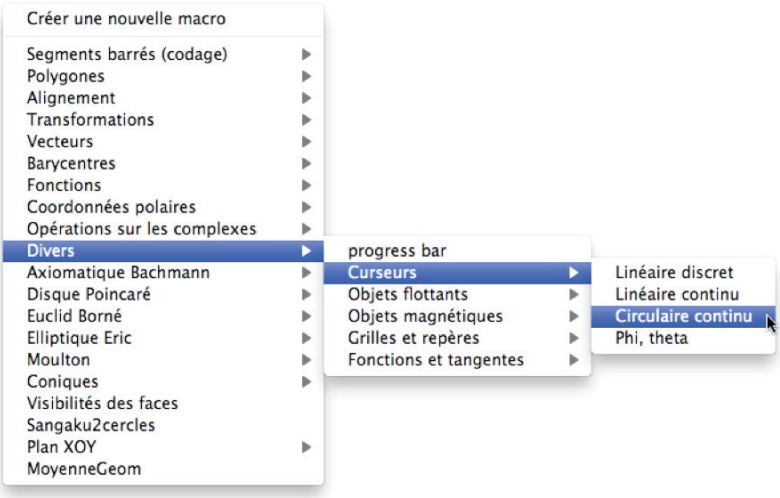


VI. Un autre traitement du temps

Dans la partie précédente nous avons vu comment la récursivité permettait de s'affranchir du déterminisme et créait ainsi, avec l'aimantation, un *déterminisme enrichi*. Mais celui-ci, tel qu'il a été proposé, ne permet pas de résoudre la question de l'enroulement continu sur un cercle qui est bien entendu un autre contexte d'affranchissement du déterminisme : on veut pouvoir compter le nombre de tours et pas seulement mesurer un angle sur une amplitude d'un seul tour.

Cette question – sans solution dans un environnement entièrement déterministe de géométrie dynamique – a été résolue par Éric Hakenholz, dès 2005, donc avant qu'il envisage même de commencer CaRMetal, dans ses explorations subtiles et fouillées de CaR. Il en a fait un article[38] (avec applet) sur son site CaRZine alors dédié à CaR. C'était la première fois qu'un logiciel de géométrie dynamique permettait de réaliser, en manipulation directe, l'uniformisation du cercle, et probablement ce résultat a-t-il participé à convaincre le futur auteur de CaRMetal à « donner un petit coup de pouce à ce logiciel », comme c'était son projet initial. Depuis, il a transformé cette possibilité en macro standard de CaRMetal, présente dès la première version, sous le nom de *Curseur/Circulaire continu*.

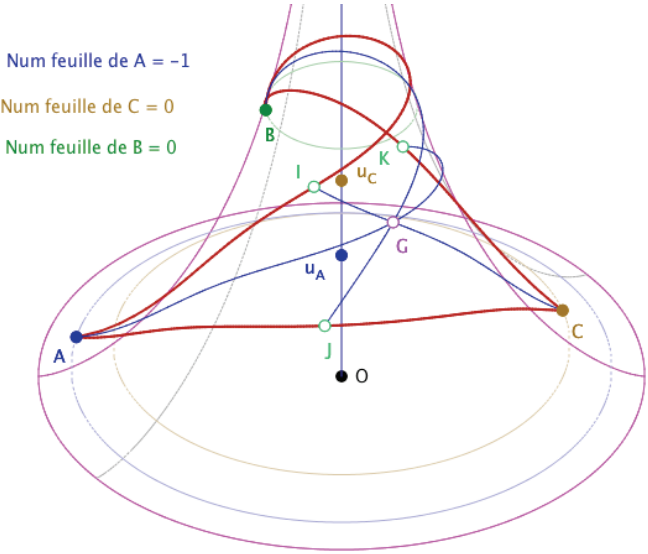
[38] http://db-maths.nuxit.net/CARzine/articles/art91/



Nous ne reviendrons pas plus avant sur cette technique[39]. Disons simplement qu'Éric Hakenholz utilise un subtil mélange de la fonction *d* de CaRMetal, déjà utilisée pour réinitialiser la récursivité, et d'une fonction *sum* qui, comme son nom l'indique, somme les valeurs d'une variable numérique.

Cette possibilité d'enroulement sur le cercle se transpose dans de nombreuses situations et, en particulier, pour ce qui nous concerne directement, dans les micromondes des géométries sur les surfaces de Beltrami. On peut donc reprendre un travail plus fin que celui mené sur la feuille principale pour construire un nouveau micromonde qui incorpore l'enroulement sur les surfaces pseudosphériques. Dans le cas de la pseudosphère, s'il faut refaire toutes les macros, elles peuvent toujours rester intrinsèques à la surface. On obtient alors des figures comme celle-ci avec un triangle ayant ses trois sommets sur trois feuilles différentes de la surface :

[39] Reprise, plus en détail, sur cette page :
http://www.reunion.iufm.fr/recherche/irem/spip.php?article50



Les médianes d'un triangle hyperboliques sont concourantes

VII. Les CarScripts

Depuis sa version 3.0 d'août 2009, le logiciel s'est doté d'un langage de scripts pour pouvoir travailler la programmation principalement dans un environnement graphique *dynamique*. CaRMetal n'est pas le seul logiciel de géométrie dynamique avec un langage de script, mais c'est le premier libre et multiplateforme[40].

Cette évolution n'est pas sans lien avec celle des programmes du lycée dont l'introduction de l'algorithmique et la pratique de la programmation. Du côté de l'élève il s'agit de proposer un environnement commun pour la géométrie et la programmation – économie de temps d'apprentissage – avec un langage non pas construit uniquement pour une utilisation scolaire, avec les artifices qui vont avec[41], mais bien un langage contemporain, largement utilisé dans le monde Web 2.0, pour lequel les navigateurs rivalisent en temps d'exécution : le JavaScript (JS).

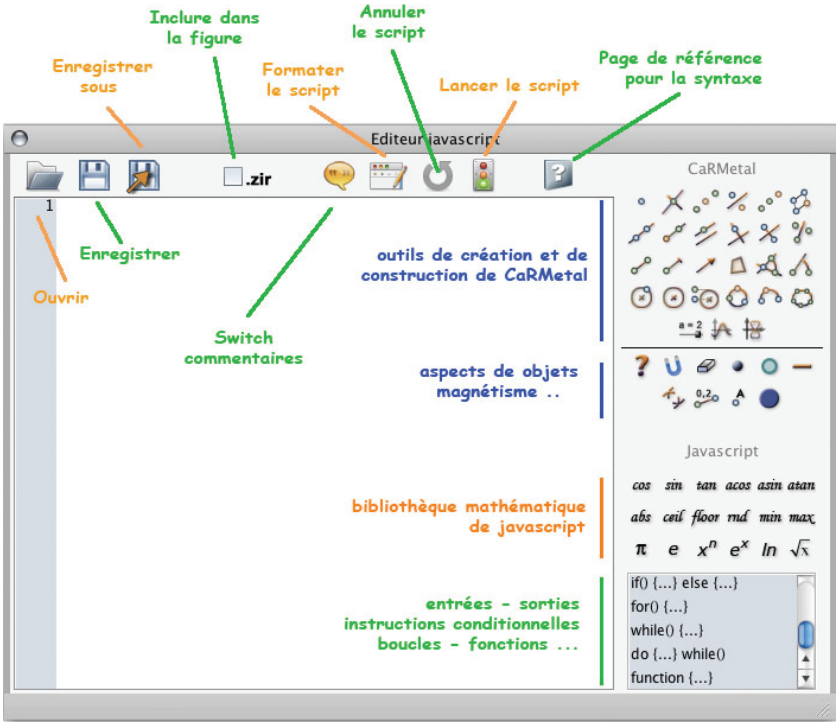
[40] Cinderella possède aussi son langage de scripts, mais il n'est ni libre ni gratuit.
[41] On se souvient de l'expérience du LSE.

Nous nous proposons dans cette partie, tout en présentant succinctement les possibilités de cet outil, d'analyser les nouveaux jeux de cadre que peut permettre une rencontre entre la géométrie dynamique et la programmation.

1. Introduction aux CarScripts

Pour l'auteur, le choix de JS relève d'un cahier des charges très précis : langage pérenne, intégrable dans un logiciel libre, utilisable en ligne, et qui plus est faiblement typé pour pouvoir ajouter les fonctionnalités géométriques de CaRMetal.

L'éditeur JavaScript de CaRMetal intègre toutes les fonctionnalités géométriques du logiciel sous forme d'icônes, ainsi que la bibliothèque Maths du JS et les principales instructions de programmation. L'utilisation du JS est ainsi simplifiée : il n'y a pas de syntaxe à apprendre, on ne fait que modifier des lignes pré-écrites. Les lignes son numérotées (renvoi du numéro de la ligne dans les erreurs) et il y a une coloration syntaxique.



Pour les premiers usages en classe, l'utilisation peut se faire en « pur JS » avec une sortie en console texte. Cette sortie est simple d'accès et pratique pour tout ce qui peut toucher les statistiques ou l'algorithmique en arithmétique par exemple. On se centrera dans la suite sur la sortie dans une fenêtre de CaRMetal car c'est là qu'est la richesse du langage puisque son intégration au logiciel est *totalement dynamique* comme on va le voir au cours des pages suivantes.

Un script de CaRMetal – on dira désormais un CarScript – peut s'appliquer à une figure vide ou à une figure préparée pour son exécution. Il y a alors deux points de vue différents. Du côté de la programmation, un CarScript est avant tout un programme qui admet comme paramètre d'entrée une figure de CaRMetal. Du côté de la géométrie, un CarScript est un outil qui va finaliser une figure que la géométrie dynamique usuelle ne pourrait pas faire – ou pas aussi rapidement. Dans un cadre scolaire, selon l'activité proposée, les choix didactiques, ses variables, l'utilisateur va être plongé, pour son objet d'étude, plutôt dans l'un des deux points de vue, l'autre servant d'outil pour la situation.

2. Géométrie repérée dynamique

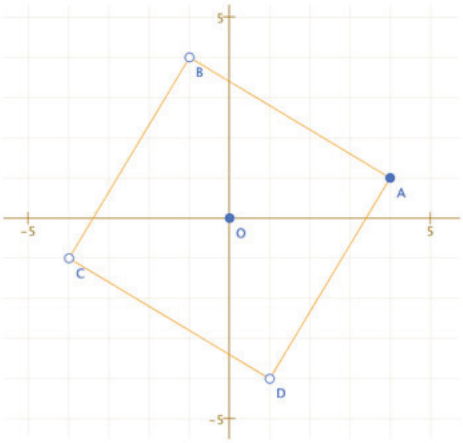
Nous allons nous attarder sur ce premier thème, très élémentaire, mais significatif de l'évolution possible des pratiques scolaires dans le cadre des nouveaux programmes du lycée. En effet, avec les nouveaux outils qu'il propose, CaRMetal autorise un nouveau regard sur la géométrie repérée car il permet de faire – très simplement – des programmes autour de ce que nous appellerons de la *géométrie repérée dynamique*, c'est-à-dire de la géométrie repérée, tournée du côté de l'algèbre. Sans aucune technicité, nous utilisons d'emblée cette spécificité dynamique de l'interaction JS-logiciel. Voyons comment sur l'exemple de la construction algébrique d'un carré, en plusieurs étapes.

Tout d'abord, commençons par un carré particulier, centré à l'origine O du repère, et de sommet A. Pour l'essentiel, il suffit donc de savoir placer un segment orthogonal à [OA] et de même longueur. Selon la classe, c'est un résultat bien connu sur l'orthogonalité, soit des droites soit des vecteurs. On peut aussi se placer en début de lycée, quand la propriété peut ne pas être connue. Une démarche (attitude mathématique du socle commun) consiste alors à utiliser une lecture graphique dans un premier temps et à engager une interprétation algébrique de cette lecture. Cette interprétation algébrique peut même être un objectif d'induction du numérique vers l'algébrique, partici-

pant à se représenter le champ algébrique comme conceptualisation du champ numérique. Le script suivant peut alors se construire par étapes :

Script inclus dans la figure :Carré A1.

```
1  // Construction algébrique d'un carré
2  // Etape 1 centre implicite à l'origine
3
4  Point("O",0,0);Point("A",4,1);
5  SetThickness("O,A","thick");
6  Point("B","-y(A)","x(A)");
7  Point("C","-x(A)","-y(A)");
8  Point("D","y(A)","-x(A)");
9  Segment("A","B");Segment("B","C");
10 Segment("C","D");Segment("D","A");
11 SetShowName("O,A,B,C,D",true);
12
```



Pour écrire les coordonnées du point B (ligne 5) sans connaître le résultat, on peut envisager la démarche *d'investigation algébrique* suivante, désormais possible grâce à cet outil :

Attitude d'investigation en géométrie repérée
- entrer la ligne 4 comme seul contenu du script,
- l'exécuter,
- lire les coordonnées (–1, 4) du point à construire,
- et conjecturer l'interprétation algébrique de la ligne 5.
- Puis annuler le script précédent, ajouter la ligne 5, exécuter le script,
- ET valider par une manipulation directe du point A.

C'est la dernière ligne de cette séquence qui donne à la démarche proposée son statut d'investigation. En effet les coordonnées des points, données dans un script, *ne fixent pas les points* dans la figure, mais *les initialisent* seulement. En faisant cette manipulation, l'élève peut déplacer A sur d'autres coordonnées entières pour non seulement valider l'orthogonalité de manière perceptive mais aussi valider que sa compréhension algébrique a bien été prise en compte dans le script comme on peut le penser. On notera que le point O ne sert pas, mais souvent les élèves placent le centre d'une figure avant de commencer. On vérifie par manipulation que le carré n'est pas véritablement de centre O mais bien implicitement de centre l'origine du repère. L'objectif suivant est alors de faire un authentique travail algébrique – avec une fiche pour une trace dans l'environnement papier-crayon – non plus seulement pour explorer avec les scripts mais bien *pour anticiper et produire* les coordonnées correctes.

Géométrie repérée dynamique : travail algébrique validé par la manipulation directe

On a choisi depuis plusieurs années d'utiliser massivement le tableur comme outil d'intermédiation entre l'arithmétique et l'algèbre : le tableur a été retenu car il permet de travailler sur les variables sans pour autant faire de l'algèbre. D'un point de vue didactique, c'est de l'algèbre « sur des objets[42] », ces objets étant les cellules. On peut rester dans une certaine opérationnalité du travail sur feuille de calcul, et peu progresser vers l'algèbre, en particulier si ce n'est pas dans le programme, ou au contraire, utiliser les cellules comme un travail « sur les objets ». Une cellule d'un tableur, avec son nom différent de son contenu, ayant été considéré par les concepteurs de programmes comme une première représentation acceptable de la variable algébrique.

La démarche que l'on propose ici, d'un travail algébrique sur la géométrie repérée avec validation par la géométrie dynamique à travers les scripts se situe clairement plus du côté de l'algèbre que le tableur. Mais elle garde aussi cette dimension de « calcul sur les objets » – réification de l'algèbre – car une fois le script exécuté, les élèves peuvent manipuler les points qu'ils ont eux-mêmes construits. La dimension algébrique est alors acceptée comme *embarquée dans ce projet* de travail sur les objets. Et par cette manipulation de

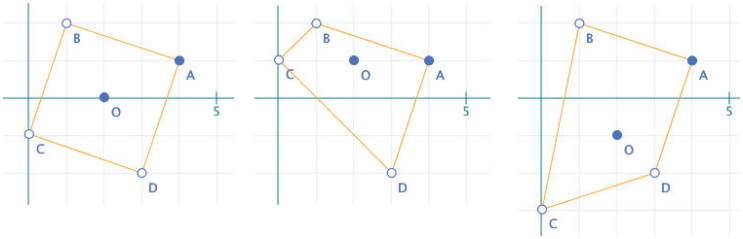
[42] Au sens du « calcul sur les objets » d'auteurs cognitivistes comme Brissiaud ou Ouzoulias dans leurs ouvrages pour l'école primaire : le passage par des objets (ici déjà conceptualisés) est un *accélérateur de conceptualisation* de ce que l'on veut enseigner. Le succès des méthodes Tchou et Picbille en est une illustration.

la figure, avec les objets produits par leurs propres scripts les élèves acquièrent aussi[43] des connaissances algébriques. Un travail régulier sur ce type de scripts dans un environnement dynamique, outre qu'il est pleinement en accord avec les nouveaux programmes, s'inscrit dans une logique générale d'apprentissage qui a amené à inscrire le tableur dans de nombreux programmes, tout en restant dans une véritable démarche algébrique. D'un point de vue didactique, il y a des hypothèses de travail à préciser et tout un champ de recherche à étudier en situation.

La situation n'est d'ailleurs pas simple pour les élèves. Ayant pris conscience de la non utilisation du point O dans le script précédent, on n'aboutit pas au carré algébrique en une seule étape, loin s'en faut. Parfois les élèves travaillent implicitement sur une seule coordonnée, sans même s'en apercevoir avant d'invalider leur travail par la manipulation des points de base.

```
1  // Construction algébrique d'un carré
2  // Etape 2 - première tentative d'intégration de O
3  // ici correct sur les abscisses
4
5  Point("O",0,0);Point("A",4,1);
6  SetThickness("O,A","thick");
7  Point("C","2*x(O)-x(A)","2*y(O)-y(A)");
8  Point("B","x(O)-y(A)","-x(O)+x(A)");
9  Point("D","x(O)+y(A)","+x(O)-x(A)");
10 Segment("A","B");Segment("B","C");
11 Segment("C","D");Segment("D","A");
12 SetShowName("O,A,B,C,D",true);
```

Un script qui produit un carré… seulement si O *est sur l'axe des abscisses*

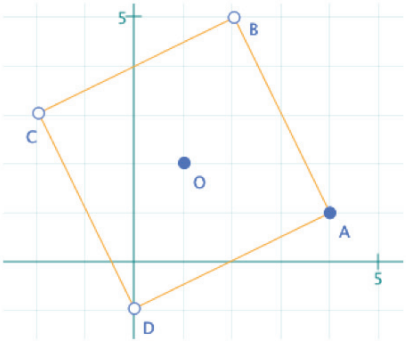


Par manipulation sur O*, et en plaçant* O *sur des valeurs entières on peut analyser ses erreurs et avoir une aide à l'organisation des calculs…*

[43] Il faut relativiser. Parfois au contraire, devant les interactions langagières entre eux face à leurs productions à l'écran, mais aussi devant leur résistance à voir leur propre capacité à algébriser, il vient à l'idée que ce type de situation est un autre contexte qui valide cette affirmation de/et que Gérard Vergnaud aime rappeler régulièrement à propos des schèmes : « la connaissance opérationnelle est supérieure à la connaissance prédicative ».

Script inclus dans la figure :Carré Algébrique

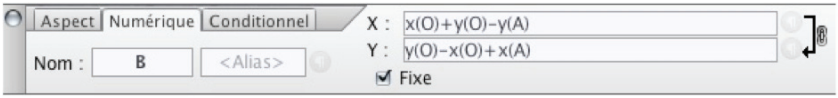
```
1  // Construction algébrique d'un carré
2  // Etape 3 -Version complète
3
4  Point("O",0,0);Point("A",4,1);
5  SetThickness("O,A","thick");
6  Point("C","2*x(O)-x(A)","2*y(O)-y(A)");
7  Point("B","x(O)+y(O)-y(A)","y(O)-x(O)+x(A)");
8  Point("D","x(O)+y(A)-y(O)","y(O)+x(O)-x(A)");
9  Segment("A","B");Segment("B","C");
10 Segment("C","D");Segment("D","A");
11 SetShowName("O,A,B,C,D",true);
12
```



... avant d'arriver à une version correcte

On retiendra de ce paragraphe l'enjeu que peut avoir, pour une entrée efficace dans l'algèbre, la manipulation d'objets, produits par les élèves, issus de leurs propres raisonnements algébriques, et les possibilités – selon les élèves, les classes et le projet d'apprentissage – d'une phase que l'on a dite *d'investigation algébrique* pour aider à l'induction vers l'algèbre.

Sur un plan plus technique, on aura observé dans les scripts précédents combien l'écriture mathématique dans les CarScripts est proche de l'écriture mathématique au tableau : pour le moment, c'est la même , elle est seulement envoyée aux points de la figure par une expression entre guillemets – par une chaine de caractères. On aura bien compris qu'alors on n'envoie pas une initialisation comme quand on donne une valeur numérique, mais bien un calcul littéral. Voici par exemple les coordonnées du point B dans le logiciel :



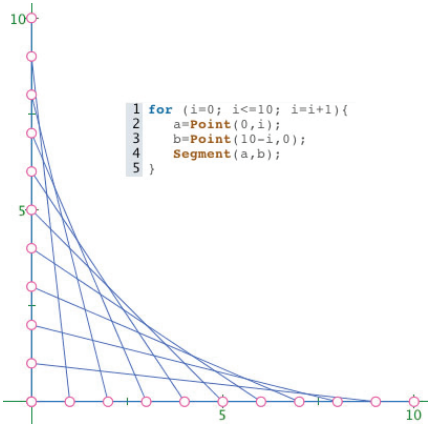
Cela signifie aussi que, dans des classes où l'algorithmique n'est pas au programme, la même démarche, mais simplement *dans l'inspecteur d'objet*, est aussi une pratique de géométrie repérée dynamique, elle aussi plus proche de l'algèbre que la pratique des tableurs. L'inspecteur d'objet, comme les autres outils, a un fonctionnement a-modal y compris dans l'écriture algébrique : chaque écriture (même partielle) qui a un sens algébrique est automatiquement interprétée et la figure actualisée, ce qui facilite l'investigation.

3. Insertion de variables JavaScript dans les figures – Exemples en analyse et statistique.

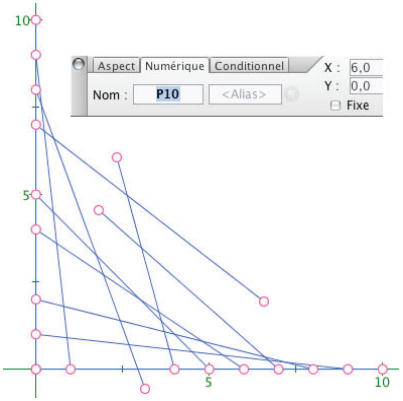
Les scripts précédents sont élémentaires au sens où ils sont écrits essentiellement du côté de la géométrie. En fait, ce sont plutôt des exercices de géométrie analytique que de programmation, auto-validés par l'utilisation des scripts et la manipulation directe sur la figure produite. Il y a donc, déjà, l'utilisation d'une interaction entre le JS et le logiciel puisqu'il y a transmission d'informations algébriques, mais ce n'est pas uniquement ce type d'interaction que l'on peut attendre d'une intégration réussie. Qu'en est-il du passage des variables JS dans les figures de CaRMetal ? Toujours dans un contexte scolaire, poursuivons notre exploration de la géométrie repérée dynamique en nous intéressant à la réalisation d'un *tableau de fils dynamique*.

Un préalable statique

Cette activité suppose avoir fait au préalable, un *tableau de fils statique*, par exemple sur les coordonnées entières dans le cadre d'un apprentissage de la boucle *for*.

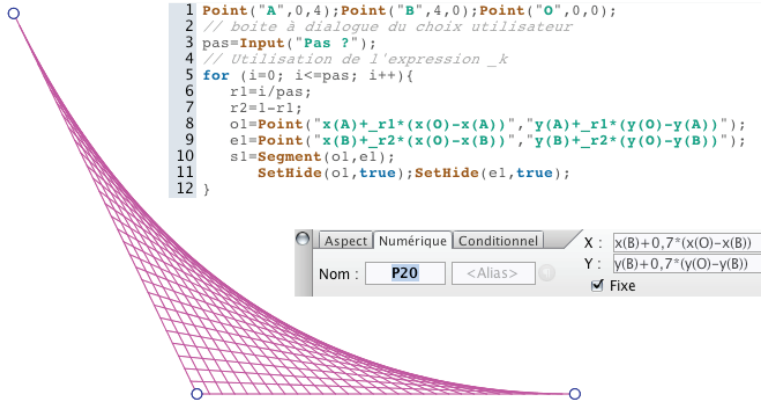


Mais puisque tous les points sont initialisés – certes à la bonne position – on n'a fait qu'un dessin – ce qui est déjà un objectif tout à fait honorable en seconde – ce n'est pas une figure « tableau de fil » au sens où il y aurait une cohérence globale interne : tous les points sont indépendants



Fils dynamiques par boucle d'itération

Faire un tableau dynamique à partir de trois points A, O, B nécessite de partager les segments [OA] et [OB], de parcourir ce partage par une boucle et de construire des points associés. Il va donc falloir envoyer une variable JavaScript à CaRMetal. La syntaxe est celle de l'adressage indirect des logiciels de programmations actuels : pour envoyer le contenu de la variable *k*, on utilise *_k* dans les coordonnées des points, comme dans ce script :

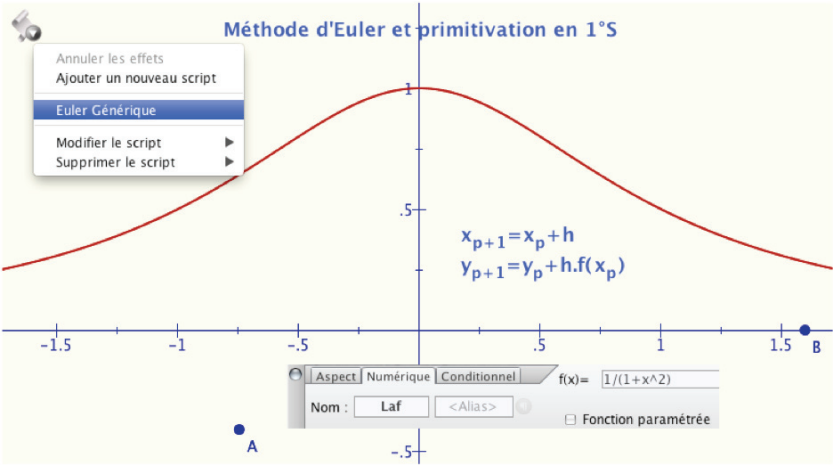


```
Point("A",0,4);Point("B",4,0);Point("O",0,0);
// boite à dialogue du choix utilisateur
pas=Input("Pas ?");
// Utilisation de l'expression _k
for (i=0; i<=pas; i++){
   r1=i/pas;
   r2=1-r1;
   o1=Point("x(A)+_r1*(x(O)-x(A))","y(A)+_r1*(y(O)-y(A))");
   e1=Point("x(B)+_r2*(x(O)-x(B))","y(B)+_r2*(y(O)-y(B))");
   s1=Segment(o1,e1);
      SetHide(o1,true);SetHide(e1,true);
}
```

En lisant la ligne 1 du script, on remarque que les segments [OA] et [OB] sont initialement orthogonaux. Sur l'illustration, le point A a été déplacé après le tracé du tableau de fils qui, parce qu'il est *dynamique*, devient en fait un tableau *affine*. À part l'underscore *_k* le reste de l'écriture est conforme à ce que l'on pratique en classe. Dans l'illustration on a placé les coordonnées de l'un des points : elles sont fonction de O et B, et du *contenu* de la variable *k*. Ce passage *par contenu* – plutôt que par valeur comme c'est le cas ici – va être plus explicite dès l'activité suivante, quand la variable ne sera pas uniquement une valeur numérique mais une structure un peu plus complexe.

4. Une figure CaRMetal comme entrée d'un script : scripts génériques

Jusqu'ici nous avons vu des scripts qui produisaient des objets dans une figure vide. Nous allons voir la réalisation des scripts génériques, qui s'appliquent à toutes les situations d'un même type, celles-ci étant considérées comme des entrées du script. Nous allons aborder cela sur un thème du lycée qui va nous permettre en même temps d'aller un peu plus loin sur l'échange entre les variables JS et CaRMetal. La situation retenue est la méthode d'Euler dans le contexte de la classe de 1re S comme primitivation approchée de fonction dont « on » (les élèves) ne connaît pas de primitive. Nous avons choisi de prendre la fonction proposée dans les documents d'accompagnement des programmes.



Le script est déjà contenu dans la figure. Du côté de la figure, il va s'appliquer à deux points A et B et une fonction *Laf*. Du côté du programme,

celui-ci prend en compte ses valeurs d'entrée, A et B et *Laf* : la figure est une donnée du script pour son exécution.

On notera l'écriture du pas avec A et B, pour que le résultat soit dynamique[44].

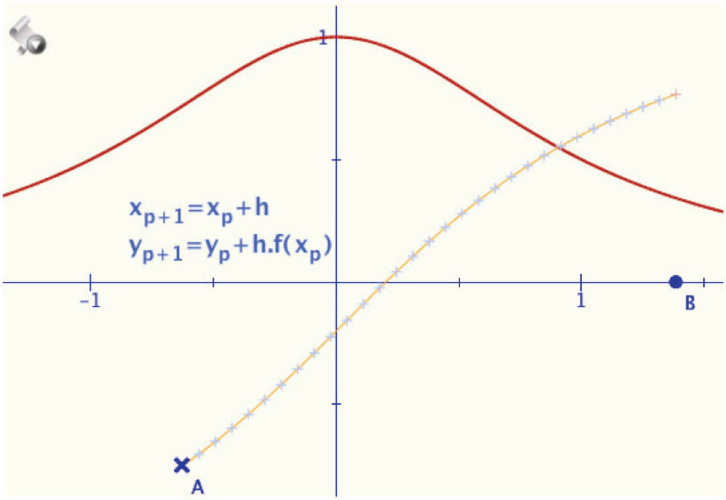
À propos de la syntaxe simplifiée : cette fois on passe par contenu non plus des variables numériques, mais des points et plus précisément leurs coordonnées. Il aurait fallu écrire *x(_m)* mais l'auteur a introduit dans ce cas-là, conformément à l'usage dans les langages de programmation ou les CAS, la simplification syntaxique *x_a* et *y_a* pour *x(_a)* et *y(_a)*, écritures qui restent fonctionnelles. C'est pour cela que l'on garde l'expression *passage par contenu* plutôt qu'un traditionnel « par valeur ».

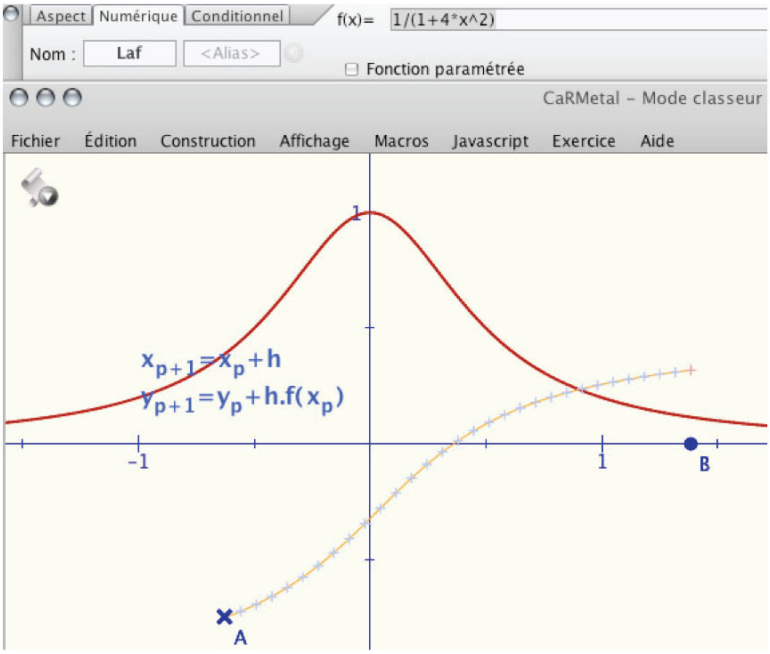
Dans l'illustration de la page suivante, en haut, l'application du script, en bas le changement de fonction, avec mise à jour immédiate[45] du résultat du script, puisqu'il est entièrement construit sur ses données initiales.

---

[44] Ceci n'est pas propre au script : si on voulait faire la construction géométriquement, par macro, pour que la figure soit dynamique, il faudrait aussi prendre le pas en fonction de A et B. Détails dans cette vidéo flash de Monique Gironce : http://db-maths.nuxit.net/CaRMetal/tutoriels/ScriptEuler/ScriptEuler.htm

[45] Rappelons l'importance didactique de l'absence de modalité dans les onglets de l'inspecteur d'objet : sur les fonctions par exemple, les élèves voient leurs modifications en temps réel, car dès que l'expression écrite a un sens algébrique, elle est aussitôt prise en compte. L'investigation algébrique peut se faire aussi dans cet onglet, sans les scripts.

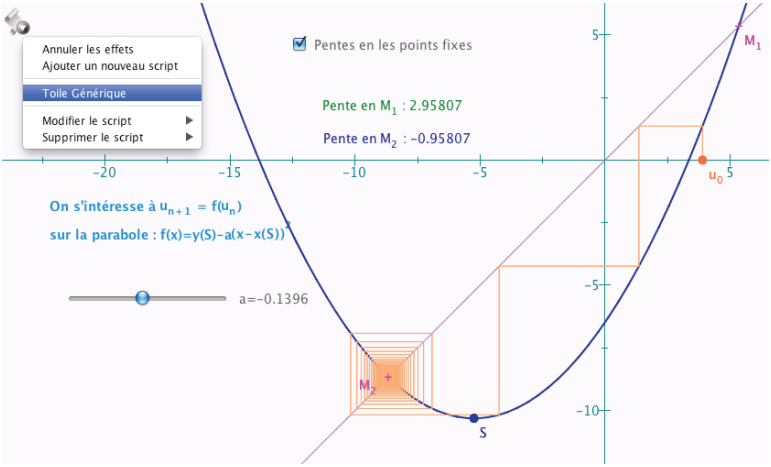

1
$x_{p+1}=x_p+h$
$y_{p+1}=y_p+h.f(x_p)$
-1
1
B
A


Aspect
Numérique
Conditionnel
f(x)=
1/(1+4*x^2)
Nom :
Laf
<Alias>
Fonction paramétrée
CaRMetal – Mode classeur
Fichier
Édition
Construction
Affichage
Macros
Javascript
Exercice
Aide
1
$x_{p+1}=x_p+h$
$y_{p+1}=y_p+h.f(x_p)$
-1
1
B
A

Script de recherche du point fixe $u_{n+1} = f(u_n)$

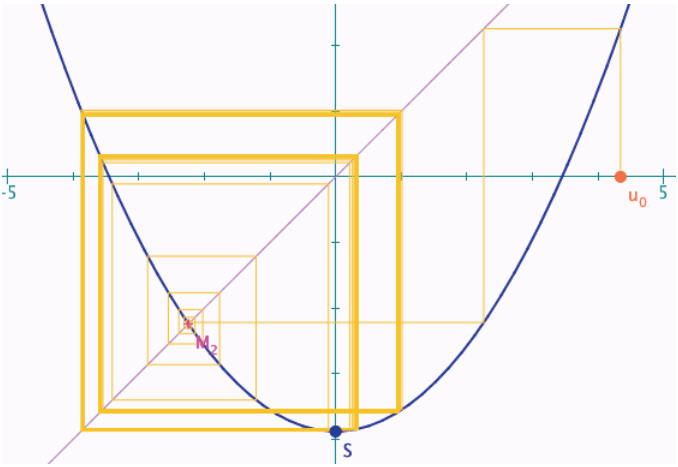
Toujours dans un contexte scolaire, voici un exemple dynamique, avec une fonction du second degré, avec paramètre et le sommet de la parabole sur l'axe des ordonnées, pour une écriture plus simple de la fonction $y = y(S) - ax^2$ que l'on continue d'appeler *Laf* dans le logiciel. On peut alors appliquer ce simple[46] script, lui aussi générique (à partir de u_0 et *Laf*) :

```
// fonctionne pour toute courbe nommée Laf avec u0
// Le script fait 100 "spirales"

u="u0";
a=Point("x_u","Laf(x_u)");
SetHide(a,true);
sl=Segment(u,a);
for (i=0; i<200; i=i+1){
    b=Point("y_a","y_a");SetHide(b,true);
    s=Segment(a,b);
    c=Point("x_b","Laf(x_b)");SetHide(c,true);
    v=Segment(b,c);
    a=Point("x_c","y_c");SetHide(a,true);
}
```

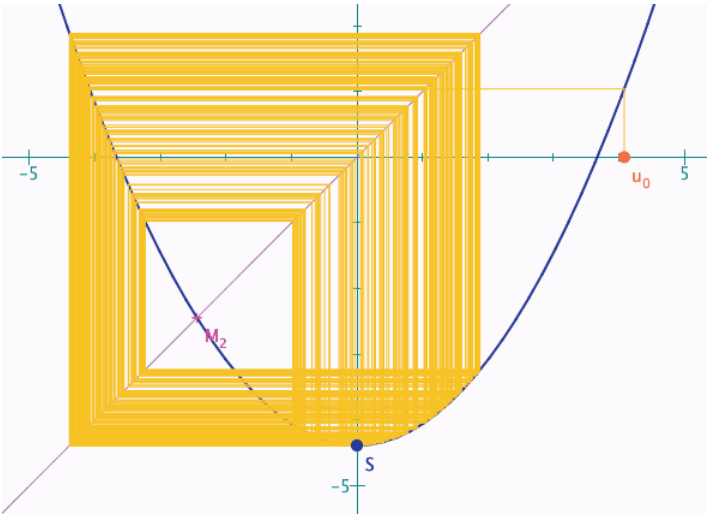


[46] En formation continue, un enseignant a eu cette remarque qu'il était « bien connu » que cet algorithme « de la toile d'araignée » n'a pas besoin de trois variables, mais que deux suffisent. C'est rassurant pour ce qui est de l'engagement des jeunes enseignants dans les nouveaux programmes. Pour des raisons didactiques évidentes, on peut maintenir trois variables même si ce n'est pas optimisé.

Selon la classe (lycée, BTS, IUT), cet outil *dynamique* permet d'aborder, d'une manière plus dynamique qu'avec un tableur même muni d'un curseur, soit simplement le phénomène de point répulsif, soit encore le concept de valeur d'adhérence (ci-dessous 4) tout ceci simplement *en déplaçant à la souris le sommet* S.

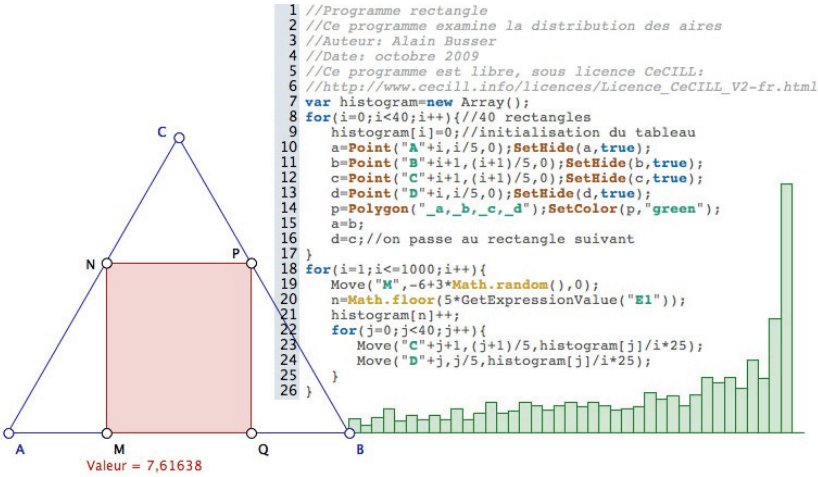


Et toujours en déplaçant S *de quelques pixels, on rencontre un phénomène nouveau, le chaos…*



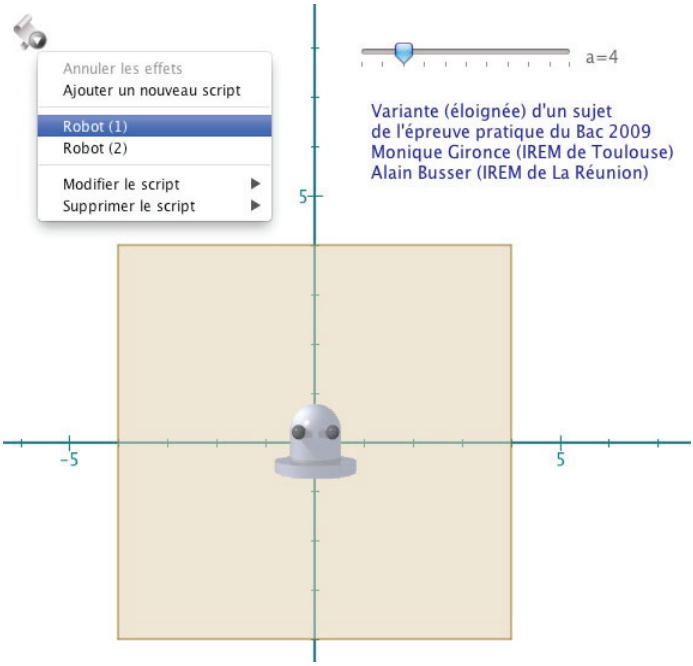
5. Utilisation en statistique

C'est un domaine à la fois simple d'accès, pour lequel l'usage des scripts est pertinent, et qui peut être plus dynamique que ce qui est proposé d'habitude – le forum de CaRMetal sur les CarScripts contient surtout des propositions en statistique. On peut bien sûr faire de simples programmes en « JS pur » avec une sortie texte, on est alors quasiment devant une calculatrice. La sortie graphique est quand même plus intéressante. Voici quelques exemples de contributions générées par les échanges entre l'IREM de Toulouse et celui de la Réunion. Tout d'abord les histogrammes sur les aires de rectangles inscrits dans un triangle.



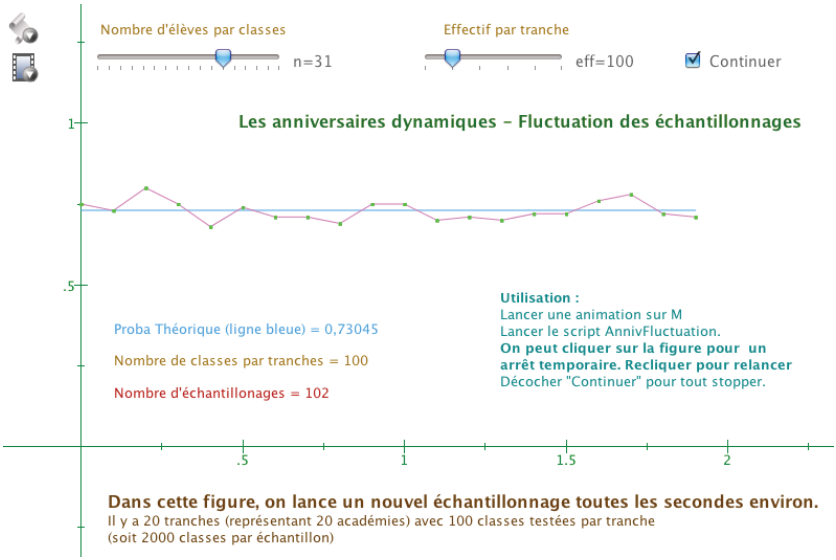

```
//Programme rectangle
//Ce programme examine la distribution des aires
//Auteur: Alain Busser
//Date: octobre 2009
//Ce programme est libre, sous licence CeCILL:
//http://www.cecill.info/licences/Licence_CeCILL_V2-fr.html
var histogram=new Array();
for(i=0;i<40;i++){//40 rectangles
   histogram[i]=0;//initialisation 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);
   }
}
```

Le thème du parcours aléatoire est fort présent dans l'ingénierie scolaire. Ici il s'agit simplement de modéliser l'exercice, à savoir étudier le nombre de lancers d'un dé à quatre faces pour sortir du carré, et d'étudier statistiquement la situation. Le premier script fait lentement une partie, visible à l'œil, une par lancer du script. Le second fait 20 parties d'affiliée pour des premiers résultats statistiques.



D'autres thèmes classiques peuvent être revisités d'une manière plus dynamique que ce qui est proposé habituellement aux élèves. C'est le cas par exemple du problème de la coïncidence des anniversaires dans une classe. Sur un plan technique, nous sommes désormais dans la préparation de figures plus sophistiquées pour les enseignants.

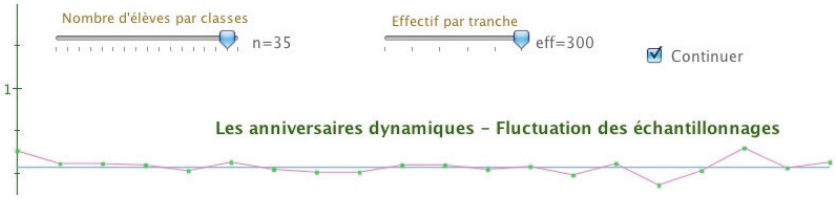
Dans l'illustration suivante, dans chacune des 20 tranches d'échantillons on effectue des tirages pour un effectif de classe donné, allant de 50 à 300 par pas de 50. Dans le même temps – et toujours en temps réel – on peut choisir par classe les effectifs allant de 20 à 36 élèves. C'est bien entendu ce point-là qui est nouveau par rapport aux autres pratiques : les scripts – et la manipulation aisée des booléens par le logiciel – permettent de mettre sur chaque point (comme une pile booléenne) les résultats pour tous les effectifs de classe, sous la forme d'une combinaison linéaire de booléens écrite par le script et mise à jour à chaque tirage.



Ci-dessous, l'ordonnée d'un point pour des échantillons de 50, pris en cours d'exécution du script [47] :

*(n==20)*23/50+(n==21)*24/50+(n==22)*26/50+(n==23)*27/50+(n==24)*30/50+(n==25)*33/50+(n==26)*34/50+(n==27)*35/50+(n==28)*36/50+(n==29)*36/50+(n==30)*37/50+(n==31)*38/50+(n==32)*39/50+(n==33)*39/50+(n==34)*39/50+(n==35)*42/50+(n==36)*43/50*

Le balayage du nombre d'élèves par classe au curseur ne fait que donner à cette expression la valeur correspondante. Le point est alors affiché en temps réel, toutes les données étant déjà dans ses coordonnées. On peut tout de suite observer l'effet des effectifs sur la fluctuation des échantillonnages et le positionnement autour de la valeur théorique.

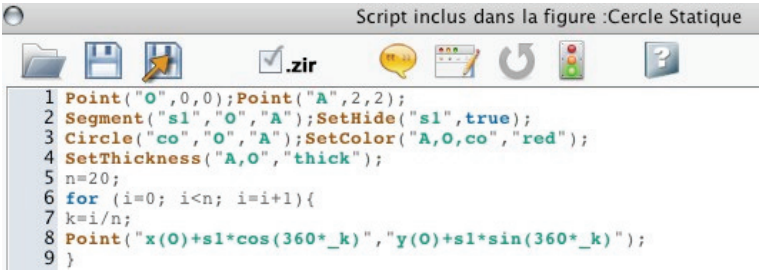


[47] Le script est détaillé sur cette page :
http://www.reunion.iufm.fr/recherche/irem/spip.php?article326

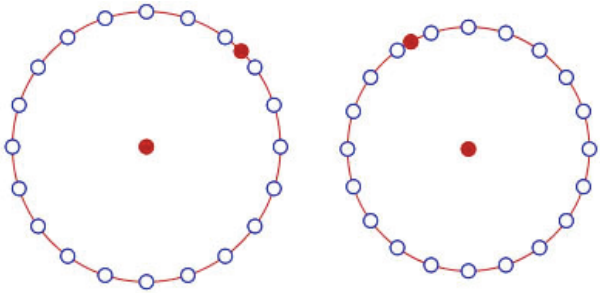
6. Statique/Absolu : un subtil mélange de statut

Nous revenons dans ce paragraphe sur la géométrie repérée dynamique. Nous allons ajouter un peu de géométrie à la démarche analytique mais le propos essentiel restera centré sur les représentations liées à la géométrie repérée. Le matériel de base va être le cercle trigonométrique, et l'enjeu sera la perception du statique et du dynamique qu'il y a dans les scripts que nous allons écrire. On pourrait penser que c'est une problématique – et une expertise – de géométrie dynamique, mais en réalité, comme nous l'avons vu au début de cette partie, la géométrie dynamique va servir, du côté de l'élève, d'outil d'auto-validation pour interroger et faire progresser les représentations de la programmation, voire éventuellement de l'algorithmique. Plusieurs ingénieries peuvent être construites avec les scripts proposés ici, selon le niveau de la classe, la pratique de la programmation. Nous ne proposons que des pistes de réflexion pour que chacun construise ses propres séquences : la pédagogie est elle aussi un micromonde dans lequel la didactique – comme ici – propose ses outils avec lesquels il faut construire.

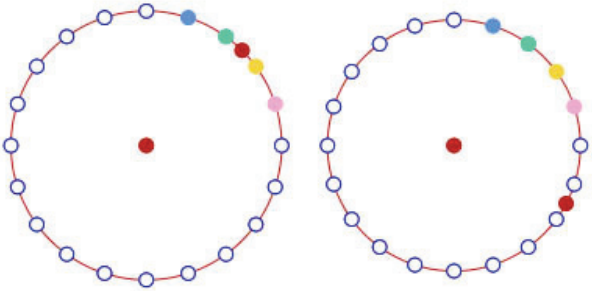
Commençons par ce script, *éventuellement que l'on propose à l'analyse en classe*, puisque c'est une des compétences mises en valeur par les programmes, sur les trois niveaux du lycée. Et ici, c'est bien dans l'analyse que le discours va être intéressant, que les représentations sur les implicites des repères – et les repères implicites – vont devoir s'exprimer.



Clairement ce script construit un cercle de centre O passant par A (points gras) et sur ce cercle place 20 points. Divers commentaires techniques peuvent être possibles, en particulier on notera que l'on a construit le segment [OA], en lui donnant un nom dans CaRMetal, *s1* ce qui permet d'utiliser *s1* comme rayon du cercle dans la ligne 8 et éviter un calcul de distance. On peut aussi parler de la référence explicite au point O, et donc de la dépendance du cercle à ce point, ce que l'on voit ci-dessous.



Pendant le déplacement de O, les plus curieux des élèves peuvent se poser des questions, sur le mouvement des autres points. Ils paraissent, à réduction prés du cercle, se déplacer en translation. Qu'en est-il en fait ? Avant de déplacer le point A, et pour voir le mouvement des points, on peut en colorier quelques-uns, ce qui donne :

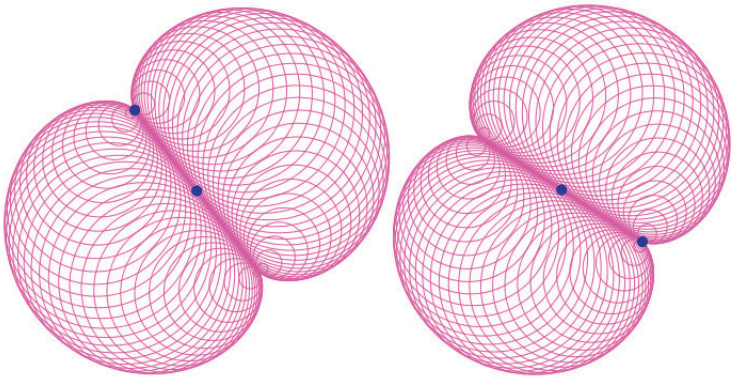


Manifestement les points, s'ils se déplacent à l'écran quand on manipule O ou A, restent figés sur le cercle de centre O, ils ne se déplacent pas sur ce cercle. *Ces points, construits géométriquement, conservent un comportement statique sur le cercle*. Selon la classe, selon si c'est une évidence ou non, un questionnement sur ce sujet peut être lancé. La raison en est évidemment élémentaire, et on peut confirmer sur les coordonnées des points que les angles utilisés sont effectivement des constantes et donc *absolus* :

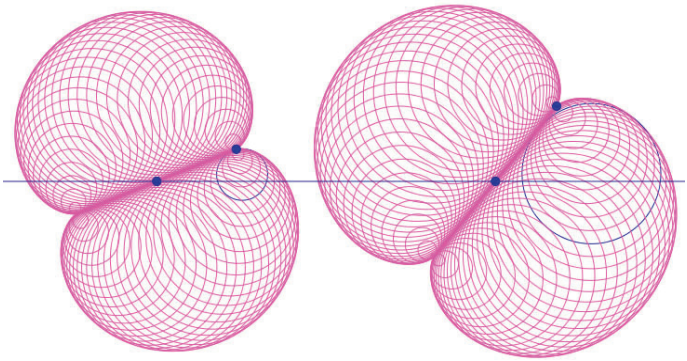
Pour avoir encadré plusieurs formations sur les CarScripts, un invariant émerge à cet endroit : ***la représentation que l'on a du dynamique est questionnée***. Elle l'est déjà sur l'articulation entre le statique et l'absolu. En effet, jusqu'ici le statique – la constante – était identifié au dessin que la manipulation directe cassait (les tableaux de fils statiques) alors qu'ici l'absolu, s'il contient une dimension statique, s'inscrit aussi dans une logique dynamique. Elle l'est d'autant que, dans de nombreuses situations, l'usage des angles absolus peut suffire. Voyons en un exemple.

Un absolu pas si statique qu'on pourrait le penser

En fait le script précédent est une partie d'un travail plus général. Le support de ce questionnement était la construction d'une néphroïde comme enveloppe de cercles. On se donne un cercle de diamètre [AB] et à partir d'un point M du cercle (les points du script précédent) on construit le cercle de centre M tangent à [AB]. On obtient alors une néphroïde. Si on s'arrange pour la construire à partir de deux points, le centre O du cercle et un point A qui définit ce cercle, elle est manipulable facilement. Et elle fait totalement illusion – au sens où l'on oublie l'aspect absolu des angles – car en manipulant A, même avec ces angles absolus, le cercle de centre un point donné bouge avec A puisque le diamètre [AB] bouge aussi.



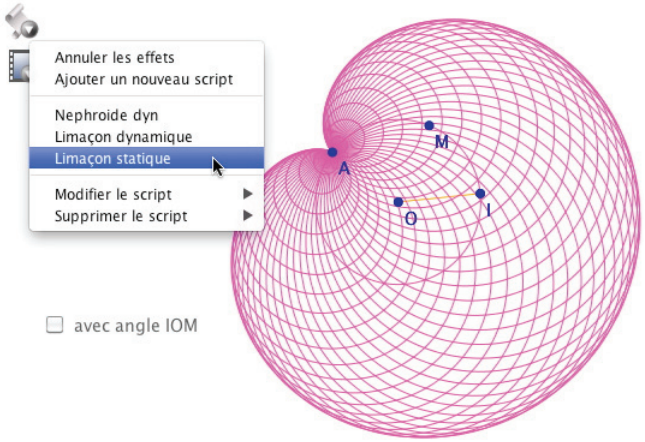
Mais si, dans un contexte géométrique donné, on devait répondre à une contrainte dynamique spécifique, comme utiliser un cercle particulier en fonction de A – le quatrième cercle avant le point A dans l'illustration ci-dessous – clairement la figure ne convient pas, c'est-à-dire *l'algorithme ne convient pas*, les angles absolus ne suffisent plus.



À gauche le quatrième cercle avant A *est mis en bleu, à droite on voit que c'est le cercle de centre d'angle absolu nul dans le script.*

```
// Nephroide
Point("O",0,0);Point("A",2,1);SetThickness("A,O","thick");
Segment("s1","O","A");Line("d","O","A");
Circle("co","O","A");SetHide("s1,d,co",true);
n=80;
for (i=0; i<n; i++){
    k=i/n;
    pi=Point("x(O)+s1*cos(360*_k)","y(O)+s1*sin(360*_k)");
    hi=Perpendicular("d",pi);SetHide(hi,true);
    mi=Intersection("d",hi);SetHide(mi,true);
    Circle(pi,mi);SetHide(pi,true);
}
```

Cardioïde version formation (dont utilisation des booléens)

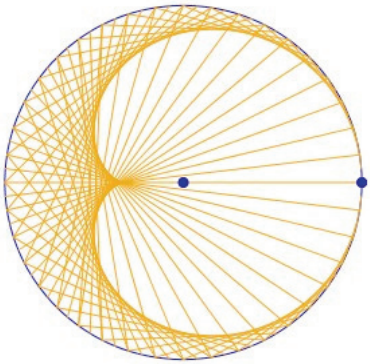


Sur cette figure, une animation de M laisse la cardioïde statique si la case n'est pas cochée, mais la cardioïde tourne avec M si la case est cochée. Le script doit inclure, dans l'angle, une constante (l'angle absolu) et une variable (le complément dynamique) en fonction de la case à cocher. C'est une occasion pour utiliser la multiplication par un booléen.

```
// on est sur un cercle centre O passant par I, M et A sont deux points
// de ce cercle. a estla variable associée à une boite à cocher et
// a1 est l'angle (sur 360°) IOM

Segment("s1","O","I");SetHide("s1",true);
n=50;
for (i=0; i<n; i++){
   k=i/n
   ci=Point("x(O)+s1*cos(a1*(a==1)+360*_k)","y(O)+s1*sin(a1*(a==1)+360*_k)");
   SetHide(ci,true);
   Circle(ci,"A");
}
```

Une version scolaire du questionnement absolu/statique

L'exemple précédent a été proposé pour mettre en valeur cette relation entre statique/absolu/dynamique, mais dans registre flou pour que cela n'apparaisse qu'après la construction, en fonction de l'usage que l'on fera de la figure. En classe, pour provoquer ce questionnement, on choisirait une situation sans ambiguïté. Se plaçant dans un contexte où l'on n'a pas abordé la question du script introductif sur le cercle pour découvrir la problématique, on peut travailler avec cet exemple : il s'agit de réaliser une cardioïde par enveloppe de segments ; on place sur le cercle n points avec un angle au centre constant entre deux points consécutifs et, en parcourant deux fois le cercle, on joint les points d'indice k et $2k$.



Alors, avec ce script...

```
Point("O",0,0);Point("A",3,0);
Segment("s1","O","A");SetHide("s1",true);
Circle("co","O","A");SetColor("co","blue");
SetThickness("A,O","thick");
n=80;
for (i=0; i<n; i++){
   k=i/n;
   pi=Point("x(O)+s1*cos(360*_k)","y(O)+s1*sin(360*_k)");
   pf=Point("x(O)+s1*cos(360*2*_k)","y(O)+s1*sin(360*2*_k)");
   SetHide(pi,true); SetHide(pf,true);
   Segment(pi,pf);
}
```

... il est clair qu'en déplaçant A, la cardioïde ne va pas suivre. Et pourtant... c'est souvent une surprise, et c'est elle qui fait surgir la question de l'angle absolu et fait ressortir des représentations de cette situation, soit sur les angles, soit tout simplement sur *les nombres réels* car après tout, utilise-t-on des angles ou des réels ici ?

Ce questionnement révèle l'un des sens de l'interrogation préalable repéré sur « notre représentation du dynamique ». C'est aussi, en fait, un questionnement sur le statut des variables. La « cardioïde de formation » a été proposée pour discuter de cette question et éclaircir la situation (avec de plus un visuel très convaincant). Les variables de CaRMetal ont toujours un comportement dynamique, celle du JS, passées algébriquement dans des objets sont dynamiques, mais passées par contenu, deviennent des nombres, et ont un comportement statique.

En définitive, à ce stade, on a envie de dire tout simplement[48] : le *statique* est du registre numérique, le *dynamique* du registre algébrique. Pour éclaircir cela, on peut proposer aux élèves la contrainte suivante : faire que la cardioïde construite par des segments suive le point A, plus précisément qu'elle soit de *sommet* le point A. Il n'y a pas grand-chose à modifier bien entendu mais cela peut être complexe pour les élèves car, pour faire cela, il est plus simple de se placer dans un autre contexte (que l'on peut préciser), celui où les objets sont déjà sur la figure.

On se donne donc un cercle de centre O passant par I, un point M sur objet du cercle. On note *ang*, l'angle (OI, OM)[49]. De fait, on s'est donné une origine des angles. Alors ce script :

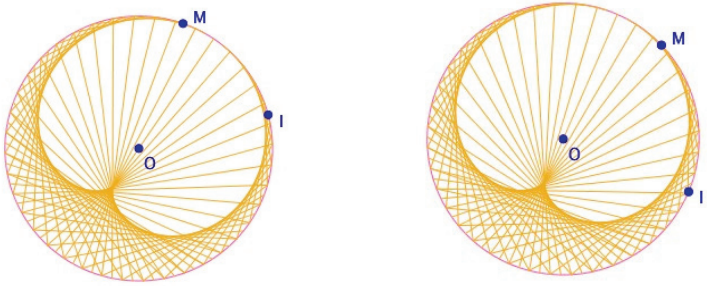
---

[48] Sans oublier le comportement affine : un point au tiers d'un segment n'est pas du registre numérique, car la manipulation directe respectant les relations affines, elle lui donne aussi, dans son contexte de point sur objet, un statut dynamique.

[49] Dans l'inspecteur d'objets, il faut donner à la mesure de *ang* une amplitude jusqu'à 360°.

```
n=80;
for (i=0; i<n; i++){
    k=i/n;
    pi=Point("x(O)+sl*cos(ang+360*_k)","y(O)+sl*sin(ang+360*_k)");
    pf=Point("x(O)+sl*cos(ang+360*2*_k)","y(O)+sl*sin(ang+360*2*_k)");
    SetHide(pi,true); SetHide(pf,true);
    Segment(pi,pf);
}
```

donne bien ce qui est attendu. En particulier une animation sur M produit bien l'effet souhaité d'une rotation de la cardioïde (à gauche sur l'illustration suivante).



Une erreur de manipulation

Malencontreusement – mais plus certainement malicieusement – un élève[50] a fait une animation sur le point I (illustration de droite ci-dessus), il demande – avec un vrai plaisir de déstabilisation potentielle – qu'il « ne comprend pas ce qu'il se passe, M n'est plus là où il faut, et pourtant, on a bien mis l'angle IOM cette fois ». Encore une belle illustration que, définitivement, l'absolu des angles est toujours présent, caché mais présent...

Bilan : premières pratiques de trigonométrie dynamique ?

Dans un registre différent de la (simple) géométrie repérée dynamique, plus délicat en terme de concepts mathématiques engagés (le cercle trigonométrique et la mesure des angles), nous commençons à dégager cette hypothèse : *l'interaction entre la géométrie dynamique et la programmation, par la manipulation directe sur les objets produits, permet d'enrichir les deux cadres de travail*. Par ailleurs, à travers des problèmes concrets (on travaille sur des objets, produits par les élèves), on manipule, sans grande technicité,

[50] Un stagiaire en fait, et nous étions bien dans la seconde option.

les outils usuels de la géométrie analytique élémentaire et de la trigonométrie en posant des questions qui relèvent de la culture des élèves (quel type de manipulation directe sur les images suis-je en train de construire ?) tout en revisitant, de manière originale, les concepts mathématiques engagés.

VIII. Aller (un peu) plus loin avec les scripts

1. Utilisation pour réaliser des figures 3D

Les scripts sont aussi une opportunité pour faire rapidement des figures complexes dans l'espace. C'est à Jérôme Caré, régulièrement présent sur les forums de CaRMetal, que l'on doit l'idée d'utiliser directement les points du repère 3D sachant qu'un point M(x, y, z) dans le repère (O, X, Y, Z) de la feuille d'espace est représenté dans le plan de la figure par le point :

$$M_e(x_O + x_M(x_X - x_O) + y_M(x_Y - x_O) + z_M(x_Z - x_O),\\ y_O + x_M(y_X - y_O) + y_M(y_Y - y_O) + z_M(y_Z - y_O))$$

On utilise alors simplement une fonction[51] générique pour les scripts comme celle-ci :

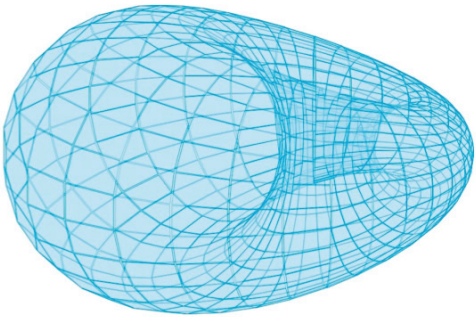
```
function TracePoint(Nom,ux,uy,uz){
   vx=ux;vy=uy;vz=uz;
up=Point(Nom,"
x(O)+(x(X)-x(O))*_vx+(x(Y)-x(O))*_vy+(x(Z)-x(O))*_vz","
y(O)+(y(X)-y(O))*_vx+(y(Y)-y(O))*_vy+(y(Z)-y(O))*_vz");
}
```

On arrive alors facilement à produire des figures 3D, *statiques* au sens de la partie précédente, c'est-à-dire qu'on ne manipule que par le trièdre, comme cette cyclide de Dupin[52] :

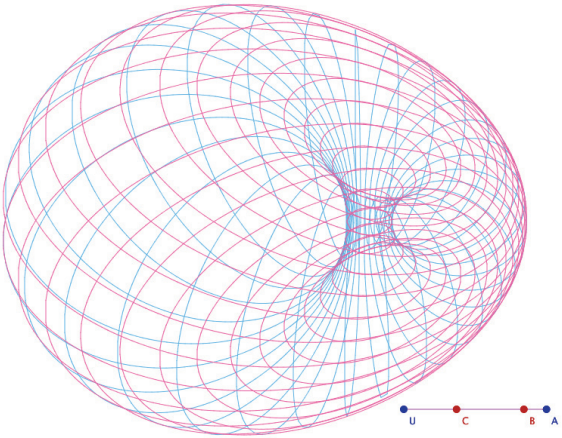
---

[51] C'est parce que les variables sont passées en adressage indirect à CaRMetal qu'il faut les recopier dans la fonction. On remarquera que dans le script de Cesaro, il n'y a pas de recopie car les variables sont utilisées de manière standard, comme simples variables JavaScript. Cette présentation de la fonction *TracePoint* est à caractère didactique pour mettre en évidence – en formation des enseignants – la place des coordonnées 3D du point dans le calcul des coordonnées 2D. Une fois cette présentation faite, une autre option plus efficace consiste à tronquer la chaîne de caractères pour utiliser directement la variable JS *ux*, *uy*, *uz* sans avoir à les recopier localement, ce qui donne : "x(O) + (x(X) – x(O))*" + ux + "+ x(X) – x(O))…".

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



La figure précédente est construite avec plus de 960 quadrilatères. On peut y ajouter des paramètres et rendre le script bien plus rapide en choisissant de tracer des ellipses à la place des polygones. Le partage en 31 parties, comme dans le cas suivant, n'utilise que 62 coniques. Les points B et C définissent les rayons des sphères intervenant dans la cyclide.



Mais ce n'est pas véritablement de la manipulation directe sur la figure comme on l'entend usuellement. Pour cela, comme le logiciel n'est pas un logiciel de 3D[53], on ne peut travailler que dans le cadre de relations entre les

[53] On voit bien qu'une gestion des coordonnées 3D permettrait une manipulation directe 3D réellement significative. Cela va être possible par script : dans la partie suivante nous allons récupérer les véritables coordonnées 3D des points à partir de leurs coordonnées 2D à l'écran et donc construire une injection de $\mathbf{R}^3$ sur $\mathbf{R}^2$!

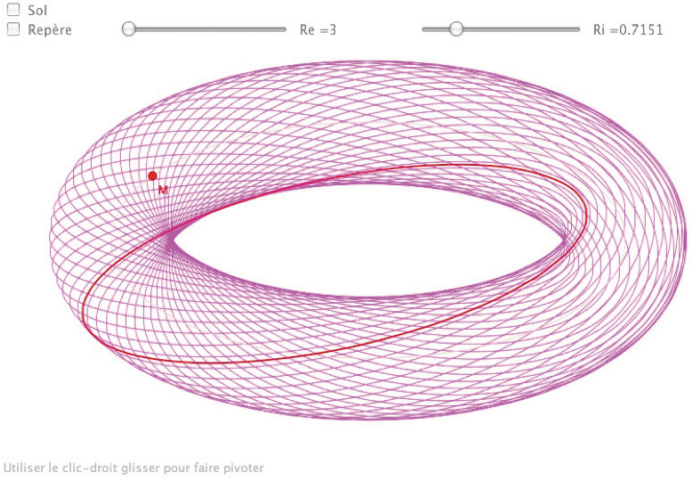
coordonnées, typiquement des relations linéaires, dans un plan. Pour donner un exemple, voyons comment manipuler un cercle à partir d'un point du sol. Pour cela il faut récupérer les coordonnées 2D dans le plan (XOY) du point M. Dans ce cas précis, il serait très simple d'utiliser un parallélisme aux axes et d'utiliser des points intermédiaires pour déterminer ces coordonnées.

Mais pour montrer que la démarche peut être plus générale, exprimons les coordonnées planes (x_p, y_p) d'un point M du plan du sol dont on a les simples coordonnées (x_M, y_M) dans le plan de la feuille. Il s'agit de résoudre un système d'ordre 2, et ces coordonnées sont, dans une notation utilisant les fonctions de CaRMetal x et y :

$$x_p = \frac{((y(M)-y(O))(x(Y)-x(O))-(x(M)-x(O))(y(Y)-y(O))}{(x(Y)-x(O))(y(X)-y(O))-(x(X)-x(O))(y(Y)-y(O))}$$

$$y_p = \frac{(-(y(M)-y(O))(x(X)-x(O))+(x(M)-x(O))(y(X)-y(O))}{(y(X)-x(O))(y(X)-y(O))-(x(X)-x(O))(y(Y)-y(O))}$$

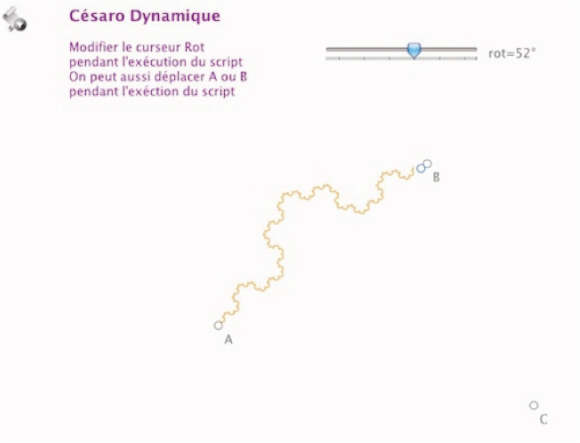
On peut alors entreprendre des figures de l'espace, entièrement construites par script, qui ont un degré de manipulation directe *sur la figure*. Par exemple, voici un tore[54] construit par 40 cercles de Villarceau, de rayon paramétrable, avec l'un d'eux accessible à la manipulation directe par le point M.



[54] Script et figure manipulable en ligne à l'adresse de la note 52.

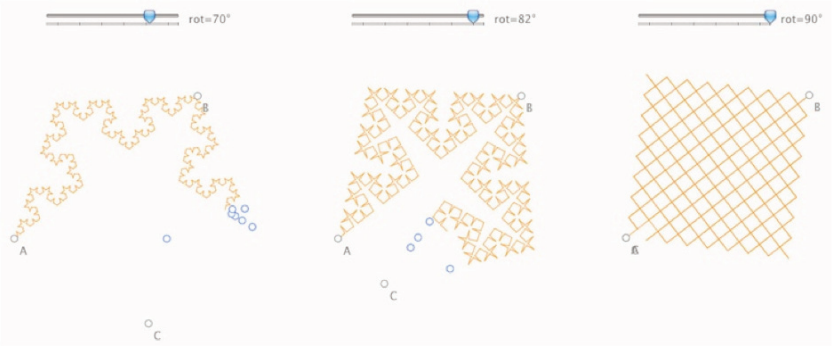
Poursuivons par deux scripts qui illustrent de manière plus profonde l'intégration réalisée au sein de ce logiciel entre le JavaScript et les outils internes à CaRMetal.

2. La récursivité dynamique (sur la courbe de Cesaro)

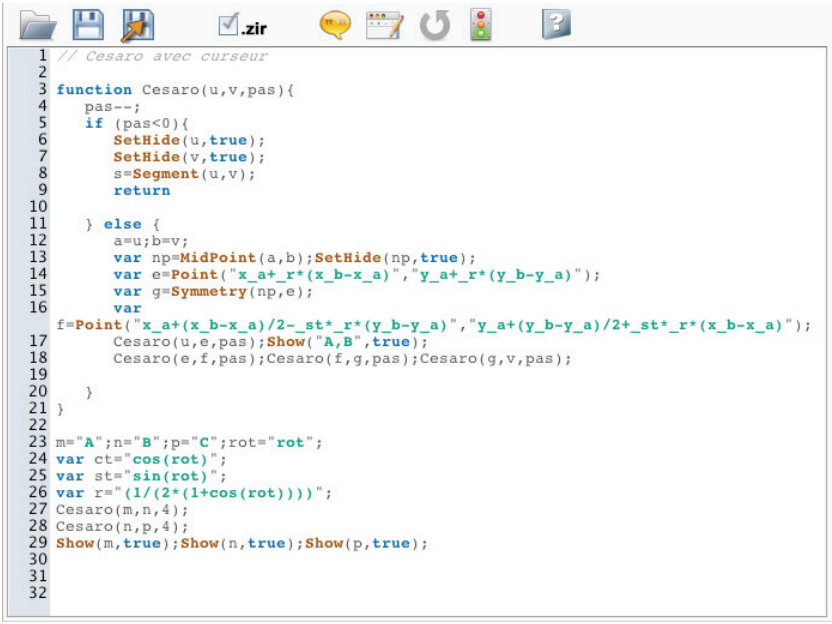


La courbe de Cesaro généralise celle de Koch avec un angle variable

La construction est récursive et la manipulation directe sur l'angle de la courbe de Cesaro est disponible *pendant l'exécution du script*. L'effet est alors saisissant puisque tout est modifié en temps réel avec le nouvel angle. En voici des illustrations prises en cours d'exécution.



Manipulation directe sur l'angle de la courbe de Cesaro
pendant l'exécution du script



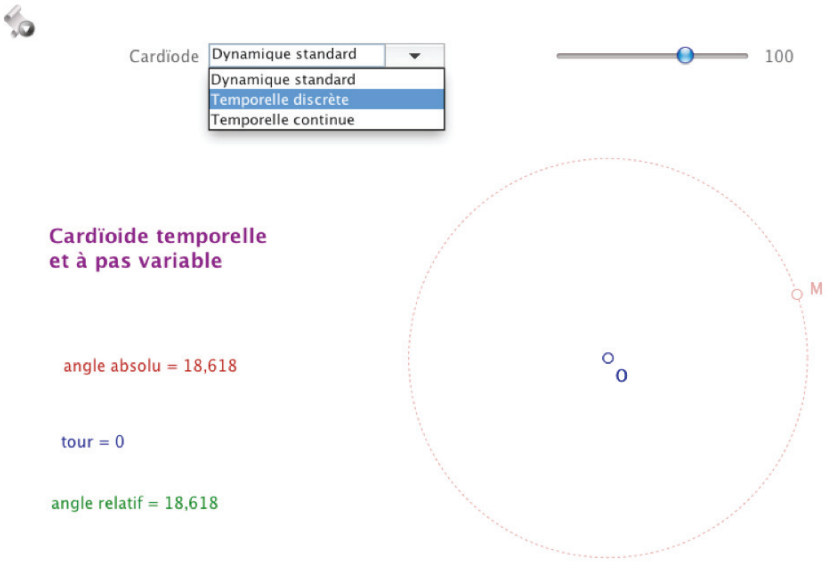
```
// Cesaro avec curseur

function Cesaro(u,v,pas){
   pas--;
   if (pas<0){
      SetHide(u,true);
      SetHide(v,true);
      s=Segment(u,v);
      return

   } else {
      a=u;b=v;
      var np=MidPoint(a,b);SetHide(np,true);
      var e=Point("x_a+_r*(x_b-x_a)","y_a+_r*(y_b-y_a)");
      var g=Symmetry(np,e);
      var
f=Point("x_a+(x_b-x_a)/2-_st*_r*(y_b-y_a)","y_a+(y_b-y_a)/2+_st*_r*(x_b-x_a)");
      Cesaro(u,e,pas);Show("A,B",true);
      Cesaro(e,f,pas);Cesaro(f,g,pas);Cesaro(g,v,pas);

   }
}

m="A";n="B";p="C";rot="rot";
var ct="cos(rot)";
var st="sin(rot)";
var r="(1/(2*(1+cos(rot))))";
Cesaro(m,n,4);
Cesaro(n,p,4);
Show(m,true);Show(n,true);Show(p,true);
```

C'est un script de figure, il utilise les points A, B et C et l'angle *rot* du curseur. Dans la figure C est le transformé de A dans la rotation de centre B et d'angle $\pi - 2*rot$.

3. Animation, script et temporalité

Retour à la question de la temporalité dans les figures : qu'en est-il des scripts et de la gestion du temps ? La nouvelle animation de la version 3.5 est-elle capable de respecter cette temporalité ?

Pour observer cela, nous poursuivons sur la cardioïde par segments déjà rendue dynamique dans la partie précédente. Nous allons y ajouter, d'une part, un pas variable en temps réel (*i. e.* il y aura de 60 à 120 segments), mais surtout une rotation de la cardioïde qui pourra prendre trois aspects : standard, temporel discret, temporel continu.



La figure avant le script, est déjà construite dans un environnement temporel

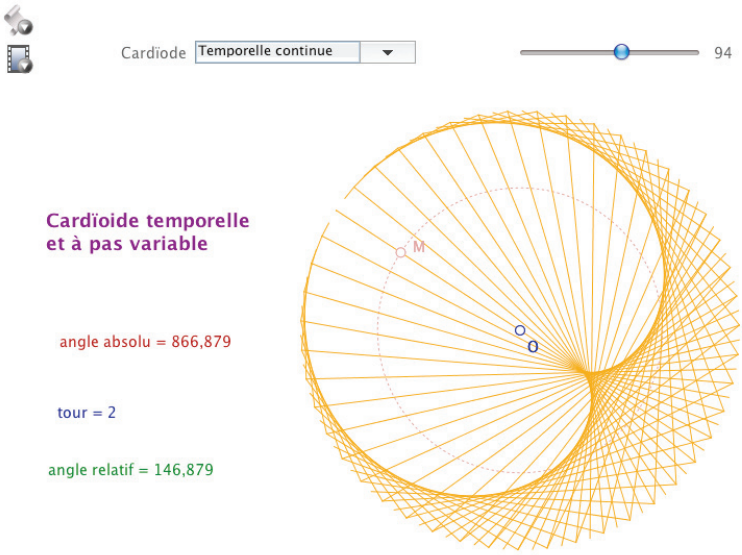
On a utilisé la macro-construction *Curseur circulaire* pour construire M. C'est un point qui s'enroule sur le cercle avec son compte-tours. La macro renvoie l'angle absolu et le nombre de tours. Les menus pop-up et le curseur sont inactifs, ce sont des données qui seront utilisées par le script suivant :

Script inclus dans la figure :CardioTempetVar

```
1 n=120;
2 for (i=0; i<n; i++){
  pi=Point("x(O)+(sA+(chx>1)*tour+(chx==3)*anrel/360)*cos(angl+360*_i/nseg)","
  y(O)+(sA+(chx>1)*tour+(chx==3)*anrel/360)*sin(angl+360*_i/nseg)");
  pf=Point("x(O)+(sA+(chx>1)*tour+(chx==3)*anrel/360)*cos(angl+360*2*_i/nseg)"
  ,"y(O)+(sA+(chx>1)*tour+(chx==3)*anrel/360)*sin(angl+360*2*_i/nseg)");
3    SetHide(pi,true); SetHide(pf,true);
4    sg=Segment(pi,pf);
5    Conditional(sg,"hidden","_i>nseg");
6 }
7
```

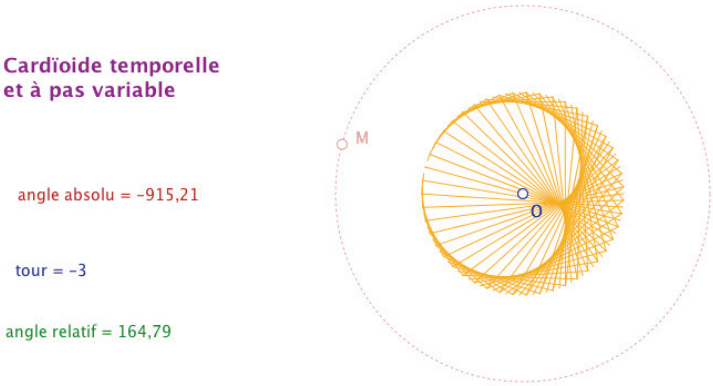
Le script construit le segment [pi pf]. L'écriture paraît complexe mais c'est seulement parce que le rayon du cercle est un peu long à écrire et qu'il est utilisé quatre fois. *sA* est le rayon du cercle initial, *nseg* est le nombre de segments, donné par le curseur. On remarquera que la boucle va jusqu'à 120, tous les segments sont construits, même si *nseg* est plus petit. Leur affichage

est traité dans la ligne 5, la dernière ligne de la boucle, le segment est caché si son indice est trop élevé. Ainsi les points ne sont calculés qu'une fois, seul leur affichage dépend du curseur. La variable *chx* est bien sûr le choix du popup. Le rayon du cercle par défaut vaut *sA*. On lui ajoute une expression si *chx > 1* (donc si on fait un choix de cardioïde temporelle). Dans ce cas on lui ajoute le nombre de tours, c'est le sens de *(chx > 1)*tour*. Si de plus on a fait le choix *temporel continu*, il faut ajouter une variation continue du rayon : c'est l'expression *(chx==3)*anrel/360*. On remarquera aussi l'économie considérable du traitement par les booléens qui évitent le recours à des conditionnels imbriqués qui seraient difficiles à mettre en œuvre dans ce cas.



La cardioïde après deux tours, en mode animation…

Bien entendu, pendant l'animation, on peut faire sauter la cardioïde d'un mode à l'autre en changeant de choix dans le pop up et faire varier le nombre de segments. Une dernière expérience illustrera, une nouvelle fois, la profondeur de l'intégration des outils entre eux, et donc la sensibilité de l'auteur à la recherche de cohérence du logiciel utilisé comme EIAH. Si avant de lancer l'animation sur M, on fait faire trois tours en arrière à M, en mode temporel, et qu'on lance l'animation, celle-ci démarre bien dans l'état temporel de la cardioïde. Il en résulte que *l'animation de CaRMetal respecte la temporalité* et le déterminisme enrichi induit.



… puis en inversant le sens de l'animation (ici à 60 segments)

IX. Aller (nettement) plus loin avec les scripts.

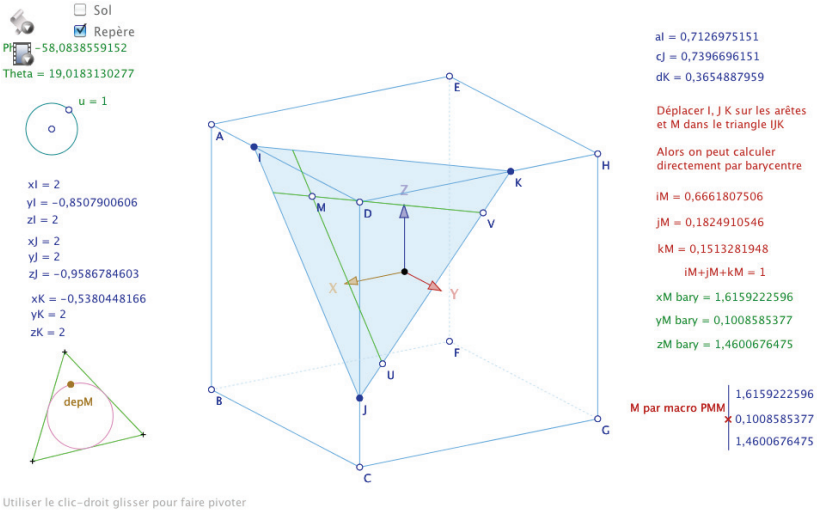
Pour terminer cette présentation des CarScripts et du logiciel, nous abordons des travaux récents (juillet 2010) de deux collègues, Jérôme Caré et Pierre-Marc Mazat. Ces travaux ouvrent de nouvelles perspectives pour l'utilisation du logiciel en classe. Et comme tout travail novateur, ils posent aussi de nouvelles questions. Les deux thèmes présentés ici ont cette particularité d'avoir été initiés par Jérôme Caré, et retravaillés par Pierre-Marc Mazat pour leurs aspects ergonomiques.

1. Retrouver les coordonnées 3D d'un point à l'écran

Le logiciel n'étant pas un logiciel 3D, les points ne sont pas stockés avec leurs coordonnées 3D. On peut naturellement penser qu'il n'y a aucune raison de rechercher les coordonnées 3D d'un point d'une figure : ce serait, selon le point de vue, construire une injection de $\mathbf{R}^3$ sur $\mathbf{R}^2$ (les points 3D depuis leurs représentations 2D) ou une surjection de $\mathbf{R}^2$ vers $\mathbf{R}^3$ (les coordonnées 2D décrivant tout l'espace 3D). Nous sommes nombreux à avoir déjà argumenté que ce n'est même pas la peine d'y réfléchir.

Mais cela n'était pas l'avis de Jérôme Caré, car les arguments des mathématiques statiques ne s'appliquent peut-être pas en géométrie *dynamique* quand ce dynamisme est à l'œuvre. Il a remarqué (en octobre 2009) que la

rotation du trièdre de CaRMetal permettait d'obtenir une bonne approximation des coordonnées des points placés dans le trièdre. Cela avait aboutit à une première macro *xyr.mcr* diffusée avec un fichier explicatif. Toutefois, cette macro, qui donnait théoriquement des valeurs exactes, était peu pratique à l'utilisation et surtout ne donnait des résultats qu'avec une précision de l'ordre de 10^{-2}, ce qui n'était pas suffisant pour une utilisation régulière. Un travail sur ce thème, avec une autre approche, fut relancé en juin 2010 par Pierre-Marc Mazat et, après de nombreux échanges de courriels pour affiner les démarches de chacun, une contribution fut publiée[55] par l'initiateur de cette *injection dynamique* de $\mathbf{R}^3$ sur $\mathbf{R}^2$ et enrichie par une procédure plus complète – mais qui demande un aménagement des figures – par le second auteur. C'est cette seconde procédure que nous allons rapidement présenter.



Dans cette figure test, M *est un point de l'ellipse de Steiner du triangle* IJK[56] *obtenu par transformation affine du point depM du cercle inscrit d'un triangle équilatéral (procédé géométrique classique). En vert : le calcul des coordonnées de* M *par barycentre, et en bleu : les valeurs produites par la macro point3D utilisant une vibration du trièdre. On peut déplacer* I, J, K, *depM, le calcul de* M *est aussitôt mis à jour, il coïncide bien avec les calculs.*

[55] http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=358
Le post de J. C. date du 30 juin, l'amélioration de P.-M. M., plusieurs fois affinée, du 31 juillet. Entre-temps les deux auteurs ont travaillé sur la partie suivante…

[56] Mais cela aurait pu être tout point du triangle par la macro « point sur plan ».

Dans le projet initial, il s'agissait de provoquer une rotation du trièdre (O, X, Y, Z) pour avoir les coordonnées d'un point, démarche suffisante pour le traitement des problèmes de géométrie dans l'espace au lycée, qui sont essentiellement numériques. Pierre-Marc Mazat a voulu aller plus loin, et a cherché à automatiser cette démarche pour arriver à la rendre utilisable en manipulation directe. Si une rotation du trièdre donne les coordonnées au moment de la rotation, pour avoir en permanence les coordonnées en manipulation directe, il faut faire tourner le trièdre en permanence… sans que cela soit perceptible. Le principe est donc de faire vibrer le trièdre en phi et en thêta, d'une faible quantité (0,01°) pour que cela reste imperceptible à l'œil, et alternativement positivement et négativement pour que le trièdre ne tourne pas.

Les premiers essais dans une boucle *while (true) … end while* ne donnent pas satisfaction en terme d'ergonomie. La manipulation directe pendant l'exécution d'un script, bien qu'elle soit possible – on l'a vu avec Cesaro – n'est pas aussi immédiate que directement sur la figure. L'idée est alors d'arriver à exporter dans la figure la vibration du trièdre et de n'utiliser le script que pour placer les expressions associées dans la figure, ces expressions n'étant évaluées qu'avec la vibration du repère. Par ailleurs, l'usage du script permet d'automatiser plus finement toute une partie de la figure.

```
30 Expression("y1"+p,
   "(-((sin(E10)-sin(old(E10)))*(y("+p+")-old(y("+p+"))))/(-(sin(E10)-sin(o
   ld(E10)))^2*sin(E11)-(cos(E10)-cos(old(E10)))^2*sin(E11))-((cos(E10)-cos
   (old(E10)))*sin(E11)*(x("+p+")-old(x("+p+"))))/(-(sin(E10)-sin(old(E10))
   )^2*sin(E11)-(cos(E10)-cos(old(E10)))^2*sin(E11)))*(!x("+p+")==x(O))",
   "x(_q)+7/pixel", "y(_q)-4/pixel");
31 Expression("y"+p, 0, "x(_q)+7/pixel", "y(_q)-4/pixel");
32 SetExpressionValue("y"+p, "if(d(Rot)==0,0,if(u==2,y1"+p+",y"+p+"))");
33
34 Expression("z1"+p,
   "(((sin(old(E10))*(sin(E10)-sin(old(E10)))*(sin(E11)-sin(old(E11)))+cos(
   old(E10))*(cos(E10)-cos(old(E10)))*(sin(E11)-sin(old(E11))))*(y("+p+")-o
   ld(y("+p+"))))/((cos(E11)-cos(old(E11)))*(-(sin(E10)-sin(old(E10)))^2*si
   n(E11)-(cos(E10)-cos(old(E10)))^2*sin(E11)))+(old(y("+p+"))-old(old(y("+
   p+"))))/(cos(E11)-cos(old(E11)))+((sin(old(E10))*(cos(E10)-cos(old(E10))
   )*sin(E11)*(sin(E11)-sin(old(E11)))-cos(old(E10))*(sin(E10)-sin(old(E10)
   ))*sin(E11)*(sin(E11)-sin(old(E11))))*(x("+p+")-old(x("+p+"))))/((cos(E1
   1)-cos(old(E11)))*(-(sin(E10)-sin(old(E10)))^2*sin(E11)-(cos(E10)-cos(ol
   d(E10)))^2*sin(E11))))*(!x("+p+")==x(O))+(x("+p+")==x(O))*(y("+p+")-y(O)
   )/(y(Z)-y(O))", "x(_q)+7/pixel", "y(_q)-30/pixel");
35 Expression("z"+p, 0, "x(_q)+7/pixel", "y(_q)-30/pixel");
36 SetExpressionValue("z"+p, "if(d(Rot)==0,0,if(u==2,z1"+p+",z"+p+"))");
```

E10 et E11 sont respectivement ϕ et θ de la figure précédente, une expression comme sin(E10) – sin(old(E10)) permet d'avoir la variation du

sinus de l'angle ϕ. Les calculs[57] montrent que cela permet d'avoir la valeur exacte de chaque coordonnée à partir d'un mouvement du trièdre avec un seul $\Delta\phi$ et $\Delta\theta$. Il faut prendre des précautions pour les coordonnées sur l'axe O*z* comme on le voit à la fin de la ligne 34. La mise en place d'une vibration complexifie la procédure, et nécessite de la séquentialiser avec précision. Il en résulte que l'on a un cycle de trois changements des variables au fur et à mesure que le repère vibre, et que les valeurs calculées ne sont correctes qu'au troisième changement. L'introduction de la variable *u* – présente dans la figure – qui prend successivement les valeurs 0, 1, 2, 0, 1… permet de n'afficher les valeurs correctes que lorsque le repère a effectué une vibration complète ($u = 2$) depuis la situation initiale ($u = 0$).

Voici un tableau récapitulant les différents stades :

Valeurs de *u*	Valeurs des angles	Expressions CaRMetal associées aux angles	Valeurs des coordonnées	Expressions CaRMetal associées aux coordonnées
0	(ϕ ; θ)	(old(E10 ; old(E11))	(x ; y)	(old(x) ; old(old(y))[58]
1	(ϕ ; $\theta + \Delta\theta$)	(old(E10) ; E11)	(x ; $y + \varepsilon$)	(old(x) ; old(y))
2	($\phi + \Delta\phi$; $\theta + \Delta\theta$)	(E10 ; E11)	($x + \delta$; $y + \varepsilon + z$)	(x ; y)

Le script affiche alors l'expression calculée (de x, y en ligne 32, et z en ligne 36) seulement si *u* vaut 2 et sinon garde la valeur précédente (et donc l'expression générale est de plus récursive[59]), ce qui évite d'afficher les valeurs intermédiaires erronées qui sont cachées dans des variables x_1, y_1 et z_1 dont l'existence sert à éviter une récursivité sur des expressions aussi longues à interpréter que la ligne 34.

Le choix d'utiliser un script permet d'automatiser plusieurs tâches comme la création de l'animation nécessaire à la vibration du trièdre et de la variable *u* si elles n'existent pas dans la figure, ou encore d'effectuer quelques considérations ergonomiques : les coordonnées prennent le nom du point cliqué (d'où le "*z*" + *p* en ligne 35 par exemple), ce qui est très pratique pour une utilisation ultérieure comme ci-dessous. Pourtant, comme on le voit, le

[57] Non reproduits ici, ils seront ultérieurement publiés dans un numéro de *MathémaTICE* (n° 21 ou 22).

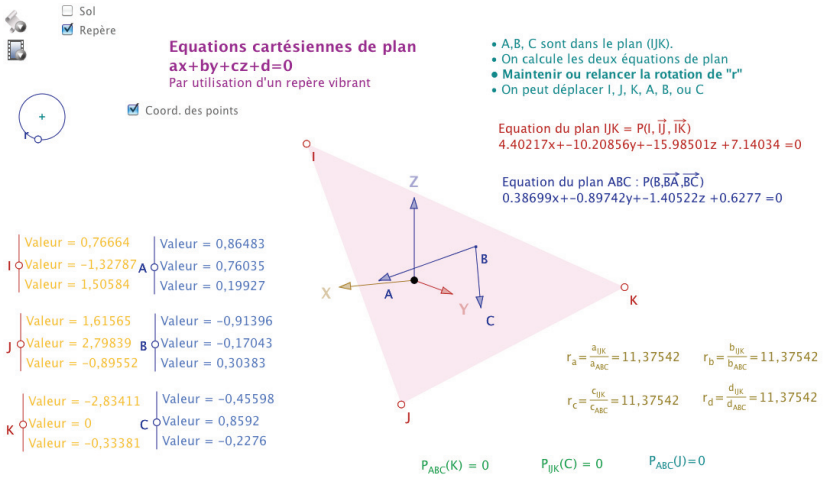
[58] Voir à la fin de la 5e ligne de la ligne 34.

[59] Ce qui l'oblige à être initialisée indépendamment – par *d(Rot)==0* – car, comme l'a déjà vu et contrairement aux points, la création des expressions ne les initialise pas pour la récursivité.

script ne sert qu'à la création et à l'initialisation des variables, ensuite tous les rafraichissements ont entièrement lieu dans la figure.

On retiendra de ce paragraphe qu'une rotation du trièdre en géométrie dynamique permet d'avoir les coordonnées réelles d'un point 3D depuis sa projection en perspective et qu'une vibration permanente du trièdre permet d'avoir les coordonnées des points en manipulation directe, et donc de faire de la géométrie 3D dans un logiciel dit de 2D+. Bien entendu, ces points doivent d'une façon ou d'une autre – par construction ou comme point sur objet (y compris de lieu ou de plan) – être liés au repère 3D de la figure.

Voici un exemple d'utilisation en cours, sur la géométrie repérée dans l'espace et les équations de plan. On dispose d'un plan I, J, K, et dans ce plan de trois points A, B, C. On calcule une équation cartésienne du plan passant par I et de vecteurs directeurs $\overrightarrow{IJ}$ et $\overrightarrow{IK}$, puis on refait de même avec le plan passant par B et de vecteurs directeurs $\overrightarrow{BA}$ et $\overrightarrow{BC}$. On peut alors comparer les coefficients des deux plans.



Exemple d'utilisation du script de repère vibrant : I, J, K, A, B *et* C *sont en manipulation directe et on dispose de leurs coordonnées 3D en temps réel.*

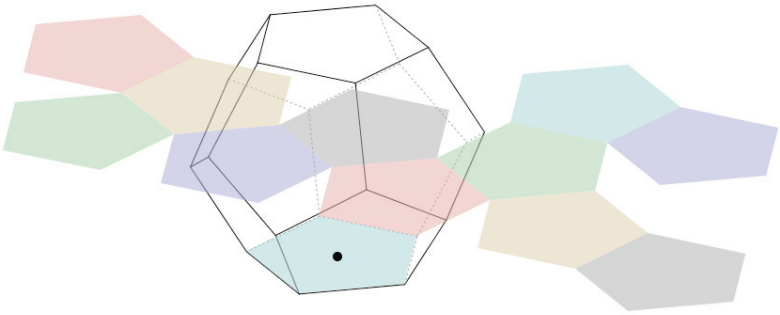
On retiendra aussi que, après 20 ans d'existence, nous avons encore à découvrir et à mettre en œuvre des propriétés assez surprenantes qu'offre la géométrie dynamique. La démarche initiale de Jérôme Caré, presque contre

une évidence mathématique, est un exemple de ce que peut être une profonde perception de la géométrie dynamique.

2. Premiers scripts objets – Patrons dynamiques de polyèdres

Lors de la création des CarScripts, l'auteur a proposé une instruction, *interactiveInput()*, qui permet de saisir des objets au cours d'un script. L'idée était de pouvoir permettre aux scripts d'être aussi de véritables extensions des macro-constructions. Et c'est de belles extensions de macros dont nous allons parler maintenant.

C'est encore à Jérôme Caré que nous devons le premier CarScript objet. Son projet initial était de produire le patron d'un cube, à partir d'un cube, l'utilisateur cliquant sur les arêtes du cube – *avec le interactiveInput()* – pour construire son patron. On pouvait ainsi construire les 11 patrons du cube, à partir d'un seul script[60]. Le lendemain, il proposait la même chose pour le dodécaèdre[61], et désormais chacun peut construire, avec un seul script, en manipulation directe, l'un des 43 380 patrons du dodécaèdre.

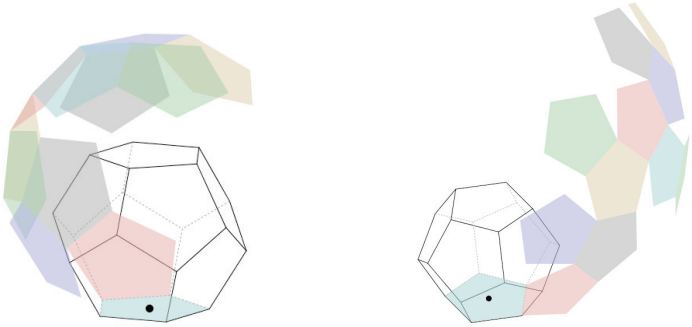


Non seulement on peut choisir les arêtes, mais l'ouverture est accessible pendant la construction – ce qui aide à choisir les arêtes.

---

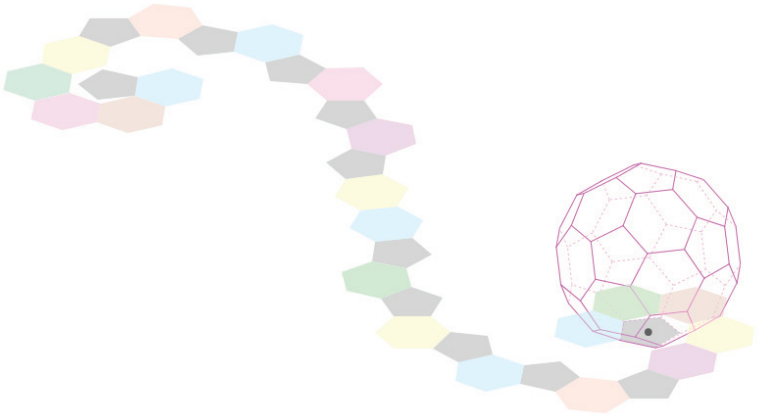
[60] Le script est téléchargeable sur cette contribution :
http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=359
dans laquelle l'auteur donne de nombreuses explications et illustrations autres que ses commentaires détaillés déjà présents dans le script.

[61] http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=360

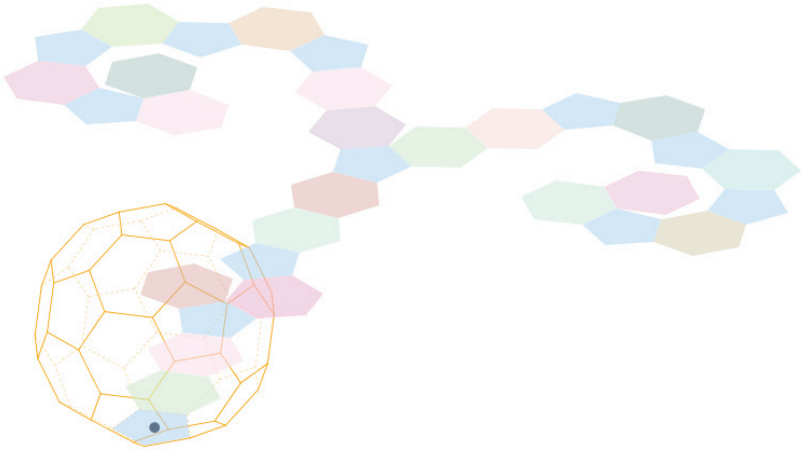


Les 43380 patrons du dodécaèdre au bout des clics de l'utilisateur

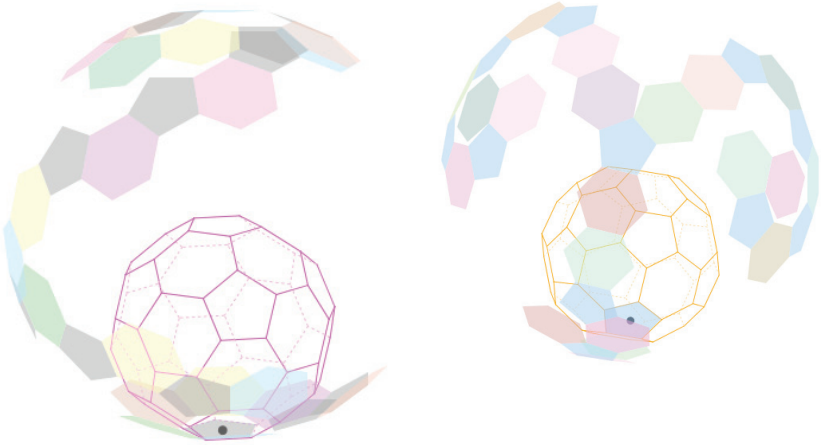
Encore une fois, Pierre-Marc Mazat a proposé une variante de ces scripts, en particulier en proposant, sur une figure plus complexe[62], la possibilité de cliquer sur les arêtes, non plus du polyèdre de départ, mais directement sur les faces du patron. Cela permet de chercher des patrons spécifiques :



[62] L'icosaèdre tronqué. Figures et vidéos explicatives disponibles à cette adresse : http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=363
On utilisera l'archive du 20 juillet, plus complète que la première. Elle permet d'enregistrer des patrons déjà réalisés et d'en produire aléatoirement. Pour voir rapidement le fonctionnement du script principal, on peut consulter la première des six vidéos proposées. Elle reprend les différentes possibilités pour construire ces patrons.



Combien de patrons de l'icosaèdre tronqué ?



On comprendra que nous ne détaillions pas plus avant les scripts de ces constructions. La seconde version est bien plus lisible, même si elle est plus compacte ; plusieurs paramètres placés dans les tableaux de points sont explicitement dans la structure objet :

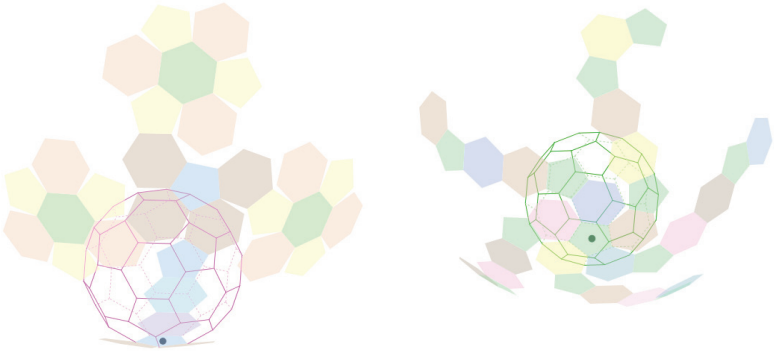
```
//Objets
Hexagone = function(art0, art1, art2, art3, art4, art5){
    this.etat = 0;
    this.arete = new Array(art0, art1, art2, art3, art4, art5);
    this.segment = new Array(6);
    this.point = new Array(6);
    this.o;
    this.z;
}
Pentagone = function(art0, art1, art2, art3, art4){
    this.etat = 0;
    this.arete = new Array(art0, art1, art2, art3, art4);
    this.segment = new Array(5);
    this.point = new Array(5);
    this.o;
    this.z;
}
```

Les objets de base utilisés dans ce script (largement commenté par l'auteur)

Comme souvent, ces premiers scripts objets[63] ont été mis en œuvre pour réaliser des figures complexes qu'on ne réaliserait pas sans eux. Mais des utilisations plus élémentaires sont tout aussi possibles et didactiquement pertinentes (à la fois pour une utilisation en classe et en formation des enseignants). C'est par exemple le cas d'un premier travail mené sur les nombres complexes[64]. Après la vibration du repère 3D pour récupérer les coordonnées des points, avec les scripts objets et la puissance de cette programmation, le travail conjoint de ces deux collègues semble ouvrir un nouveau chantier pour la géométrie dynamique, avec ses premières illustrations dont on ne se lasse pas…

[63] On trouvera un script pour les patrons de l'icosaèdre à cette adresse : http://db-maths.nuxit.net/CaRMetal/forums/viewtopic.php?t=364

[64] Disponible dans ce wikibook : http://fr.wikibooks.org/wiki/Programmation_objet_et_g%C3%A9om%C3%A9trie/CaRScripts_et_nombres_complexes
Voir aussi cette page, initiée par J. Caré, qui explore autrement les possibilités 3D : http://fr.wikibooks.org/wiki/Programmation_objet_et_g%C3%A9om%C3%A9trie/Points_3D_avec_CaRScript



X. Conclusion et perspectives

La naissance de l'équipe EREDIM au sein du LIM de l'université de la Réunion a été l'occasion de proposer un tour d'horizon des potentialités didactiques du logiciel CaRMetal pour un usage de la géométrie dynamique et/ou de la programmation. Ce tour d'horizon a été composé à partir de réflexions issues d'une pratique personnelle de la GD depuis 1989 et de nombreuses formations initiales et continues animées depuis plusieurs années avec ce logiciel, dans le premier degré et dans le second degré.

Dans cette partie nous ciblons quelques-unes des possibilités de recherches en didactique que l'on peut dégager de ces premières réflexions. Certaines seront menées au sein de l'EREDIM dans le cadre de masters « enseignement » ou de masters « recherche ». Nous invitons les collègues didacticiens à s'emparer eux aussi de ces thèmes pour les faire vivre à leur façon, dans leurs structures, au sein de leurs parcours didactiques, et bien entendu à en dégager d'autres sur cette géométrie dynamique, effectivement enrichie.

Rappelons les quelques points qui émergent de notre texte. Tout d'abord, nous avons vu que l'anticipation systématique de tous les outils provoque de fait une nouvelle instrumentation de la géométrie dynamique. Celle-ci peut être expérimentée et explorée dans des ingénieries précises qui vont instrumentaliser quelques particularités. On pense en premier lieu à l'étude des constructions des représentations chez les jeunes élèves (cycle 3 de l'école primaire, cycle d'adaptation du collège) par appropriation des schèmes

d'usage[65] des outils. Pour ces recherches, le tout nouvel outil « Monkey », outre qu'il favorise lui aussi l'instrumentation de la GD en tant que telle – et donc le concept de figure géométrique – devrait pouvoir être instrumentalisé dans des ingénieries adaptées, plus particulièrement pour étudier l'investigation sur « la géométrie des propriétés des objets » de l'équipe ERMEL.

Une autre piste à ce niveau d'expérimentation pourrait être un travail de réflexion sur les problématiques et les hypothèses de recherches, la mise en œuvre d'ingénieries sur des configurations finies, avec la manipulation directe discrète que nous avons dégagée comme première application didactique de l'aimantation. Les problématiques peuvent être variées, sur l'apprentissage des configurations ou de leurs propriétés – par la perception spécifique du discret comme on l'a vu avec le quadrilatère orthocentrique, mais aussi l'appropriation des schèmes d'action dans un contexte discret pour mieux s'approprier le contexte naturel, etc.

Dans tous les cas une variable didactique importante est l'environnement restreint utilisé. Une réflexion doit être engagée pour que cet environnement ne transforme pas la richesse et la richesse d'investigation d'un micromonde en simple exerciciel à environnement dynamique.

Dans le cycle d'orientation du collège, sur la base d'hypothèses précises, des ingénieries peuvent être développées pour tester l'introduction des fonctions dans le passage du registre discret (les entiers les demi-entiers) au registre continu. Cela n'a pas été présenté, mais il n'est pas nécessaire d'utiliser l'aimantation pour cela, on peut utiliser la grille du logiciel qui peut être aimantée sur l'unité et la demi-unité.

Au lycée, l'a-modalité de tous les outils, mais plus particulièrement de l'inspecteur d'objet, a été soulignée pour entreprendre, avec l'onglet Numérique de l'inspecteur d'objet, d'étudier ce que nous avons appelé la géométrie repérée dynamique (GRD) avec son engagement du côté de l'algèbre tout en permettant un travail, une manipulation – et une expérimentation par les élèves – sur les objets (la GRD comme objet d'étude). On a vu que le même travail, mais avec des objectifs de recherche largement différents, peut-être envisagé avec un emploi spécifique des scripts pour utiliser la géométrie

[65] La même chose serait passionnante à travailler sur des géométries non standard comme le plan de Moulton, pour explorer comment des étudiants en mathématiques déjà chevronnés (M1 par exemple) utilisent l'anticipation des outils pour s'approprier plus rapidement le comportement des objets de cette géométrie dans ce modèle et en tirent des conclusions sur les possibilités de configuration. Il n'est pas certain que cela puisse être mis en place, mais c'est un projet de recherche personnelle potentielle.

repérée dynamique à des fins de validation dans un travail de programmation (la GRD comme outil pour les élèves).

Dans le cadre de l'utilisation des scripts, nous avons vu comment ce contexte, à travers des ingénieries adaptées, peut précipiter la réflexion sur le statut de variables (y compris chez les enseignants) par la confrontation de deux types de variables, celles du logiciel, naturellement dynamiques et celle du JavaScript qui peuvent être passées algébriquement à la figure ou par valeur numérique, et donc avec un statut de nombre. Nous avons vu – sur une pratique de formation continue, qu'il n'y avait aucune ambigüité sur le partage d'un segment, et comment l'ambigüité naissait sur le partage du cercle. Il y a là tout un champ de recherches didactiques possibles.

Toujours au lycée, des recherches peuvent être entreprises sur l'usage de la logique dans des figures, notamment dans un cadre de « logique-outil » – d'abord avec l'onglet Conditionnel de l'inspecteur d'objet, sur des ingénieries accessibles à de nombreuses classes, puis ensuite avec les booléens pour un usage réservé aux classes scientifiques. Nous avons en particulier vu comment l'usage de la logique réifiait l'unicité d'une solution quand elle se décline en plusieurs cas à traiter dans le champ des configurations. Des recherches sur la logique-outil et éventuellement sur cette réification peuvent être entreprises aussi dans les deux premières années de l'université.

Sans que nous ayons véritablement de pistes bien précises pour une recherche didactique en classe, nous envisageons à l'EREDIM de poursuivre la réflexion sur le *déterminisme enrichi*, plus particulièrement sur l'engagement direct augmenté, quand celui-ci propose à l'utilisateur un environnement global riche, recherché au cours d'une phase d'investigation, dans lequel l'utilisateur peut entrer et en sortir.

L'interaction entre le langage de script et cette implémentation de la géométrie dynamique avec toute la richesse que l'on a vue dans la dernière partie invite à entreprendre des développements propres, pour l'étude de cette intégration elle-même. Quelles applications imaginer pour la récursivité croisée au sein des scripts ? Comment intégrer la temporalité dans des scripts ? Quelles orientations didactiques aborder avec la vibration du trièdre et cette nouvelle *injection dynamique* de $\mathbf{R}^3$ sur $\mathbf{R}^2$? Quelles ingénieries spécifiques va-t-on développer, mais aussi quelles nouvelles problématiques va-t-on rencontrer ? Enfin, avec les scripts objets, que les objets soient d'essence scolaire ou non, de nombreux champs d'investigation sont envisageables même s'ils peuvent être, dans un premier temps, éloignés de la didactique.

Bibliographie

Textes historiques

BELTRAMI Eugène (1868), « Essai d'interprétation de la géométrie non euclidienne », trad. par Jules Hoüel, *Annales Scientifiques de l'École Normale Supérieure,* 6, p. 251-288.

BOLYAI Janos (1867), « La science Absolue de l'Espace indépendante de la vérité ou de la fausseté de l'axiome XI d'Euclide (que l'on ne pourrait jamais établir à priori) ; suivi de la quadrature géométrique du cercle dans le cas de la fausseté de l'Axiome XI », trad. par Jules Hoüel, *Mémoires de la Société des sciences physiques et naturelles de Bordeaux*, 5, p. 189-246.

COXETER Harold Scott MacDonald (1989), *Introduction to Geometry,* 2nd ed., Wiley.

HILBERT David (1971), *Les Fondements de la géométrie*, éd. critique et trad. par Paul Rossier, Dunod (1re éd., *Grundlagen der Geometrie*, Teubner, 1899).

MOULTON Forest Ray (1902), « A simple non-Desarguesian geometry », *Transactions of the American Mathematical Society*, 3, p. 192-195.

Géométrie dynamique

ACOSTA Martin (2008), *Démarche expérimentale, validation, et ostensifs informatisés. Implications dans la formation d'enseignants à l'utilisation de Cabri en classe de géométrie*, Thèse de doctorat de l'université de Genève.

BELLEMAIN Franck (1992), *Conception, réalisation et expérimentation d'un logiciel d'aide à l'enseignement de la géométrie : Cabri-géomètre*, Thèse de doctorat de l'IMAG, Grenoble.

GALWICK Thomas (2002), « Dynamic Notions for Dynamic Geometry », *5th International Congress on Teaching Mathematics with Technology (ICTMT 5).*

GENEVÈS Bernard (2004), *Vers des spécifications formelles : Fondements mathématiques et informatique pour la géométrie dynamique*, Thèse de doctorat de l'université Joseph Fourier Grenoble 1.

KORTENKAMP Ulli (1999), *Foundations of Dynamic Geometry*, Thèse d'habilitation de l'ETH Zürich.

KORTENKAMP Ulli & RICHTER-GEBERT Jürgen (2001), « Grundlagen dynamischer Geometrie », in *Mathematische und didaktische Aspekte Dynamischer Geometrie Software*, Hildesheim & Berlin : Franzbzecker.

LABORDE Jean Marie (1985), « Spécification informelle pour un Cahier de Brouillon Informatique pour la géométrie », in *Environnements Interactifs d'Apprentissage avec Ordinateurs*, Eyrolles, Paris.

LABORDE Jean Marie (1996), « Explorations en géométries non euclidiennes », in *Actes de l'Université d'été « Cabri-géomètre, de l'ordinateur à la calculatrice, De nouveaux outils pour l'enseignement de la géométrie »*, Grenoble

MARTIN Yves (2003), *Conception et mise en œuvre de micromondes de géométries non-euclidienne dans le cadre de la géométrie dynamique, illustrées avec Cabri-géomètre. Expérimentation en formation des maîtres*, Thèse de l'université Joseph Fourier-Grenoble 1.

PAPERT Seymour (1980), *Jaillissement de l'esprit. Ordinateurs et apprentissage*, Paris, Flammarion.

EIAH

BALACHEFF Nicolas (2002), « Contribution à la réflexion sur la recherche sur les Environnements Informatiques pour l'Apprentissage Humain », in *Les technologies en éducation : Perspectives de recherche et questions vives*, Paris, INRP, MSH & IUFM de Basse Normandie, p. 193-201.

BEAUDOUIN LAFON Michel (1997), *Interactions instrumentales : de la manipulation directe à la réalité augmentée*, Orsay, LRI-CNRS.

SHNEIDERMAN Ben (1983), « Direct manipulation : a step beyond programming languages », *IEEE Computer*, 16 (8), p. 57-69.

NANARD Jocelyne (1990), *La manipulation directe en Interface Homme Machine*, Thèse d'État de l'U.S.T., Montpellier.

TCHOUNIKINE Pierre (2002), « Pour une ingénierie des EIAH », *Revue I3 information-interaction-intelligence* (www.revue-i3.org).

Mathématique et Didactique

BRISSIAUD Rémi (2003), *Comment les enfants apprennent à calculer*, Paris, Retz.

BROUSSEAU Guy (1986), *La théorisation des phénomènes d'enseignement des mathématiques*, Thèse d'État de l'université de Bordeaux 1.

CUPPENS Roger (1996-1999), *Faire de la géométrie… avec Cabri*, 4 vol., Paris, APMEP.

POLSTER Burkard (1998), *A Geometrical Picture Book*, New York, Springer

TOURNÈS Dominique (2004), *Autour d'un mémoire de Vincenzo Riccati : histoire de la construction tractionnelle des équations différentielles*, Thèse d'habilitation de l'université Pierre et Marie Curie-Paris 6.

Webographie

Cet article a son complément en ligne, avec toutes les figures citées – et bien d'autres – manipulables en ligne et téléchargeables, ainsi que tous les scripts :

Pour télécharger les classeurs de géométrie :
http://db-maths.nuxit.net/IREMrun/art372/a372inter.html

Pour télécharger les classeurs de scripts :
http://db-maths.nuxit.net/IREMrun/art375/a375galerie.html

Le site de CaRMetal pour utiliser les classeurs directement
http://db-maths.nuxit.net/CaRMetal/
(téléchargement, tutoriaux vidéo, ressources, forums)

Sur le site de l'IREM de la Réunion :

Présentation de la version 3.5 (avril 2010)
http://www.reunion.iufm.fr/recherche/irem/spip.php?article347

Le groupe de travail Algorithmique et CarScripts :
http://www.reunion.iufm.fr/recherche/irem/spip.php?rubrique58
avec un manuel de référence de 67 pages (Alain Busser), des articles de prise en main pour l'enseignant, et de propositions d'utilisations dans lesquelles les exemples vus ici sont repris et largement détaillés, avec plusieurs dizaines de scripts. On trouvera aussi deux rubriques très fournies d'Alain Busser :

- Narration de recherche sur les TP d'algorithmique en classe de seconde.
- Sujets de l'épreuve pratique TS 2009 avec de nombreux scripts.

Le groupe de travail Abaques et nomogrammes
http://www.reunion.iufm.fr/recherche/irem/spip.php?rubrique49
utilise les CarScripts. Les scripts (Alain Busser) sont essentiellement dans les quatre sous-rubriques.