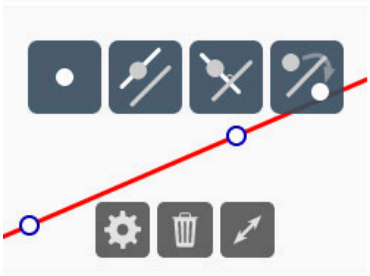
La Géométrie dynamique - option webApp et tablettes tactiles
Introduction à DGPad – octobre 2015

DGPad est une application de géométrie dynamique orientée tactile, mais disponible en WebApp sur ordinateur. Elle a l'avantage d'avoir un module 3D particulièrement facile à utiliser pour les élèves. Comme pour la géométrie, son module de tracé de courbe est aussi avec anticipation des constructions, ce qui peut être l'occasion - en lycée, d'explorer les fonctions autrement que ce qui peut se faire généralement.

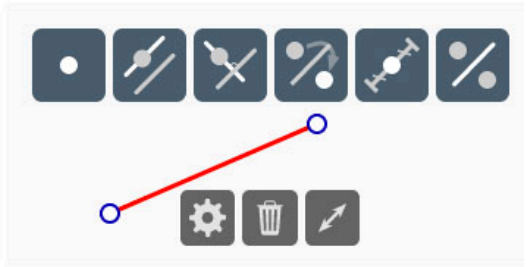
1.1. Un nouveau concept d'interface - les palettes conceptuelles

Vous avez peut-être pratiqué GGB ou Cabri, ou encore CaRMetal : tous ces outils on une interface classique pour leurs outils : on dit que les outils sont préfixés : on choisit l'outil puis les objets associés. On ne s'en aperçoit pas toujours (à force si quand même) car on a l'habitude et on va assez vite mais cette interface est particulièrement « gourmande en clic », surtout s'il y a des menus déroulants.
L'auteur de DGPad a réfléchi à une interface plus efficace nécessitant moins de contact (de « touch ») sur tablette, ou clic de souris sur ordi. Il a choisi une approche radicalement nouvelle, celle des palettes contextuelles à la situation. Informatiquement, c'est le choix d'une programmation objet centrée sur les objets géométriques et non plus sur les outils. C'est comme si on venait réveiller un objet (point, droite, segment, polygone, cercle) et qu'il nous proposait tout ce qu'on peut faire avec lui. Les palettes sont donc différentes pour un point, une droite, un segment, un cercle …

Palette contextuelle du point



… de la droite



… du segment

Conceptuellement, on peut dire qu'on passe d'une interface où les outils étaient préfixés (tous les autres logiciels de GD), à une interface où ils sont infixés. C'est une nouvelle habitude à prendre surtout pour les outils à trois entrées ou plus (cercle circonscrit, angle, polygone) : le pointeur tactile - le doigt - ne se comporte pas comme une souris : quand on lève le doigt, le système ne sait pas où il va apparaître, alors qu'on sait toujours où est le pointeur de la souris même si on ne clique pas dessus : on passe d'un pointeur continu (ordinateur) à un pointeur discontinu (tablette). L'interface doit en tenir compte pour forcer le lieu où l'on attend le pointeur pour conserver des constructions en anticipation - avec un « fil à la patte ».

1.2. Le tableau de bord - Mode standard et mode consultation

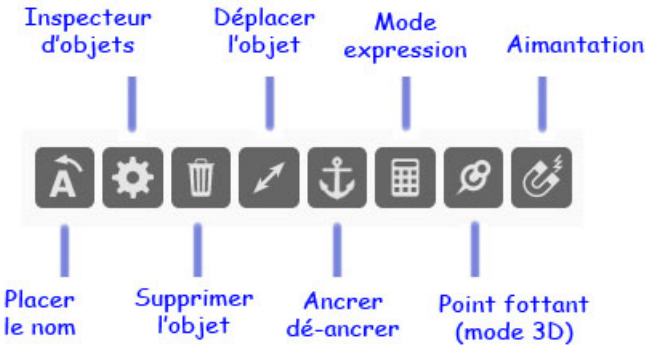
Le tableau de bord est généralement sélectionné sur un mode donné. Or la plupart sont des interrupteurs, on peut les désactiver, et quand aucun outil n'est sélectionné, on est en **mode consultation** : on peut agir sur la figure sans créer des points par inadvertance : le mode consultation fait partie intégrante de cette interface préfixée très ouverte.

Dans ce mode consultation, en particulier le repère 3D est plus rapide et il ne se déplace qu'à un doigt (tablette-trackpad) ou clic-gauche-glisser (souris) au lieu de deux doigts (respectivement clic-droit-glisser). Quand on charge une figure, par défaut elle est en mode consultation. Un «tap » (un clic sur ordi) sur un item du tableau de bord l'application passe en **mode standard** (de création d'objets).

En classe - surtout sur tablette - on peut apprendre aux élèves à passer d'un mode à l'autre rapidement pour manipuler la figure « en consultation ».

Manipulation d'une figure en mode consultation : **http://goo.gl/dtawaZ**

1.3. La palette des comportements (elle aussi contextuelle)

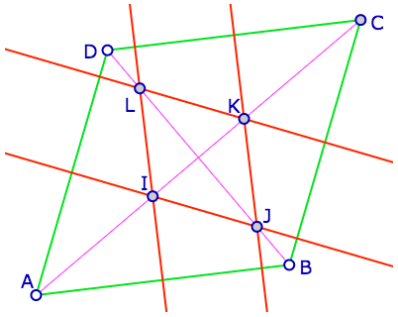


Cette palette de comportement contient :

- des comportements spécifiques (d'où son nom) : ancrage et aimantation en 2D, point flottant en 3D.
- des raccourcis du tableau de bord (inspecteur, suppression, calculatrice)
- des réglages qui ne se font que sur la fenêtre elle-même (nom des points et déplacements d'objets superposés)

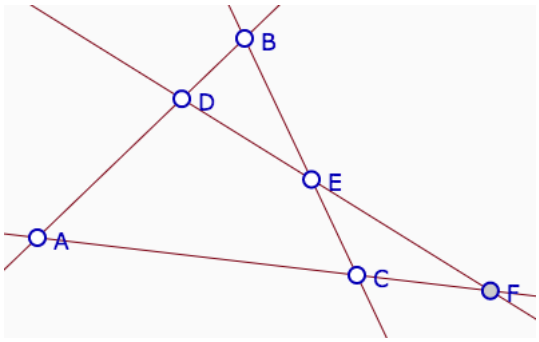
Partie A - Instrumentations géométrique et algébrique
Trois figures planes pour un premier tour d'horizon

Exercice 1 : un premier contact : les médiatrices d'un losange.
a) Se mettre en mode nommage automatique. Sur un cercle de centre A passant par B prendre un point (attention au nommage) D.
b) Construire le point C pour que ABCD soit un losange.
Cela peut se faire en terminant le parallélogramme :
• géométriquement (construire de préférence des segments plutôt que des droites)
• analytiquement : puisque $C = B + \overrightarrow{AD}$ on peut construire dans DGPad (en mode calculatrice) simplement le point $B+D-A$.
b) Construire les 4 segments de ABCD, puis les médiatrices du losange. Elles se coupent deux à deux en IJKL.



c) Conjecturer la nature de IJKL avec les outils du logiciel (mesure dans inspecteur d'objets).
d) Que remarquer sur les diagonales de IJKL par rapport au losange initial ? (cela se montre très facilement avec les propriétés des médiatrices)

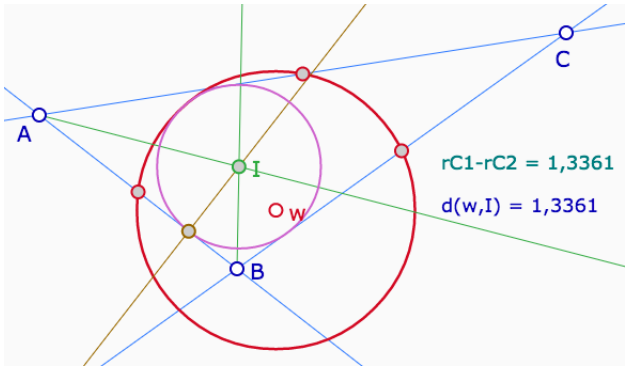
Exercice 2 : Conjecture - Utilisation d'une première macro standard.
a) Se mettre à nouveau en mode nommage automatique, construire, *avec des droites*, un triangle ABC puis une droite transversale (DE) comme ci-contre, compléter avec le point F.



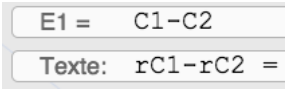
b) Placer le prochain nommage sur I et construire I, J, K les milieux de (B, F), (D, C) et (A, E) respectivement.
c) Une conjecture ?
d) Utilisation des macros de bibliothèque : ouvrir les macros constructions, choisir « Tests » puis « alignement » et montrer les trois points.
Complément plus technique (pour les enseignants). Ce logiciel n'est pas un logiciel de calcul formel, il est numérique, mais il utilise, dans son mode de calcul, beaucoup d'écritures formelles. Nous allons en voir la richesse en interprétant ceci :
e) Ouvrir le **mode expression** et taper **(K-I)/(K-J)**. Valider par le bouton vert. Ce qui est obtenu est mathématiquement (très) pertinent. Comprendre ce qu'il se passe en l'interprétant dans le cadre d'une identification des points à leurs affixes complexes.
Conserver la figure pour la suite

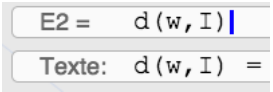
Exercice 3 : Manipuler des outils à 3 items.
On reste sur la même figure, supprimer les points D et E, cela doit supprimer une partie de la figure. Continuer pour qu'il ne reste que les 3 points A, B, C, et les trois droites.
a) Construire les bissectrices de deux angles du triangle et leur intersection I, puis la perpendiculaire à un côté passant par I, et son intersection avec le côté.
b) Terminer le cercle inscrit par l'outil « Cercle et point ».

c) Tracer, sans les nommer, les milieux des côtés du triangle (en montrant 2 points). Puis utiliser l'outil « cercle par trois points) pour tracer le cercle circonscrit aux trois milieux.
d) Utiliser le mode « cacher montrer » pour révéler son centre, le renommer **w** - avec l'inspecteur d'objet - (il s'appelle souvent ω mais il n'y pas encore de lettre grecques).

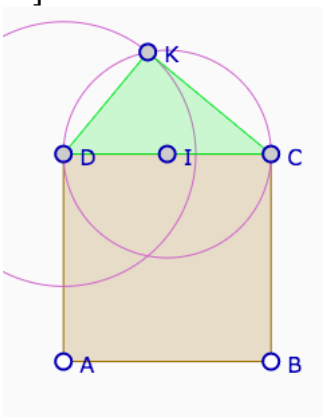


e) utilisation de la calculatrice (ou *mode expression*)
Dans un premier temps faire le calcul de la différence des rayons, en montrant simplement les deux cercles, valider. Puis dans une autre expression, en utilisant la commande **d(,)** *du clavier math de DGPad*, calculer la distance entre les deux centres.
f) que signifie, mathématiquement, que les deux résultats sont identiques ?



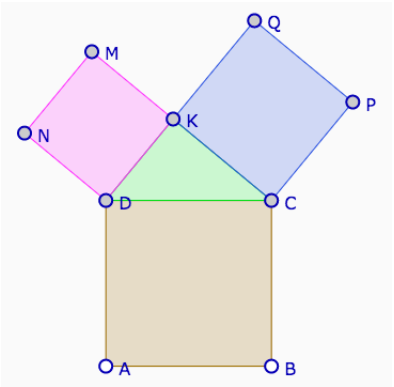


Partie B - Micromonde - fabrication de macro-construction
Exercice 4 : Exemple avec la réalisation de la fractale de pythagore
Conseil : être très attentif à l'ordre des points quand vous créez les polygones.
a) Se donner un curseur **E1** qui va de 0 à π/2, initialisé par exemple à 1.
b) Faire apparaître les axes, prendre une parallèle à l'axe horizontal passant par A et un point B sur cet axe. Cacher les axes, la droite, tracer le segment [AB].
c) Commencer par construire le carré ABCD, soit géométriquement, mais de préférence analytiquement (moins d'objets) : *D=A+i*(B-A)* et *C=B+D-A*
d) Construire le cercle de diamètre [CD] - avec l'outil *cercle par centre et point*, en prenant le milieu I de C et D.
e) Construire ensuite un cercle de centre D *de rayon quelconque*, puis en cliquant sur le cercle ouvrir la calculatrice et mettre comme rayon **d(D,C)*cos(E1)**, ce qui est bien la longueur que l'on veut. Les deux cercles se coupent en **K** au dessus de[CD]. Construire K en s'approchant de l'intersection.
Le polygone carré doit être construit dans l'ordre A B C D.



Note technique : cela aurait été ici plus compliqué de prendre un point K sur le cercle plutôt qu'un curseur, car il aurait fallu modifier la macro à la main dans le fichier.

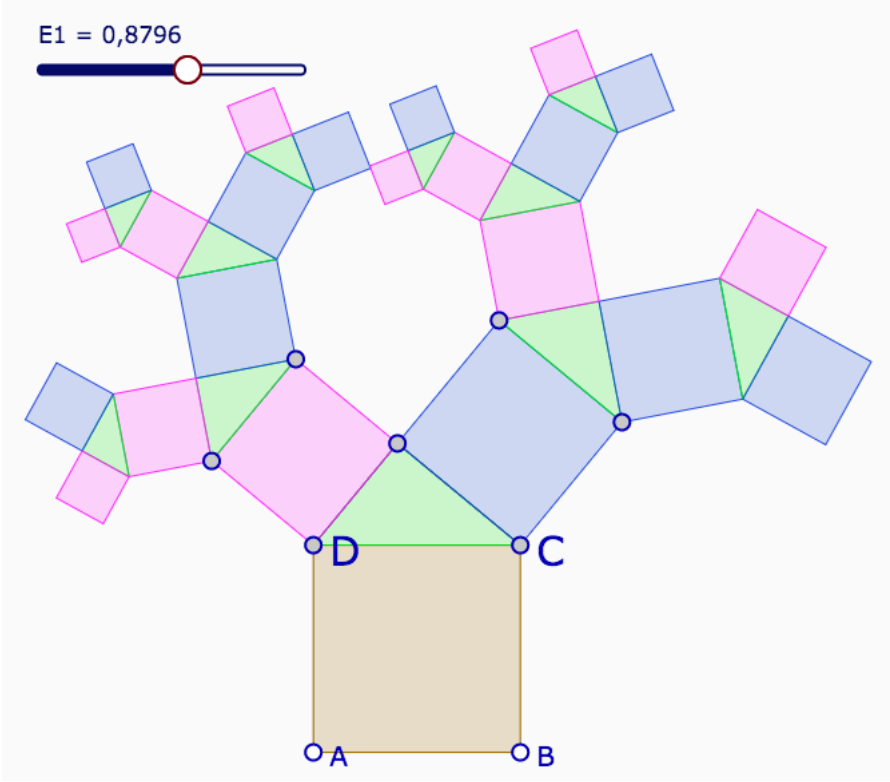
f) Poursuivre en créant deux carrés avec l'outil carré des macros (icône macro, choisir *polygones/quadrilatères/carré*) en montrant bien *dans l'ordre D et K*, puis *K et C*.
• On peut supprimer tous les côtés autres que [DK] et [KC]. Cela fera moins d'objets inutiles pour la suite.
• *Il est inutile de nommer les points*, ils sont mis ci-contre uniquement pour communiquer l'ordre.
g) Construire le triangle **DCK** puis les carrés dans **cet ordre : DKMN et KCPQ**. En effet en respectant cet ordre on reproduit en objet final de la future macro, l'objet initial ABCD, donc cela pourra être reproductible.



h) Phase finale : faire la macro construction en montrant d'abord le curseur **E1** puis le polygone ABCD.

Cela apparaît en objets initiaux, montrer simplement ensuite les trois polygones : le triangle et les deux carrés. Puis donner un nom à votre macro et valider.

Vous n'avez plus qu'à appliquer votre macro au polygone rose et bleu et successivement … Pour une première macro construction, félicitations ! Voici ce que cela donne …



Appliquer la macro plusieurs fois, puis agir ensuite sur le curseur

Partie C - Représentations graphiques et anticipation des constructions

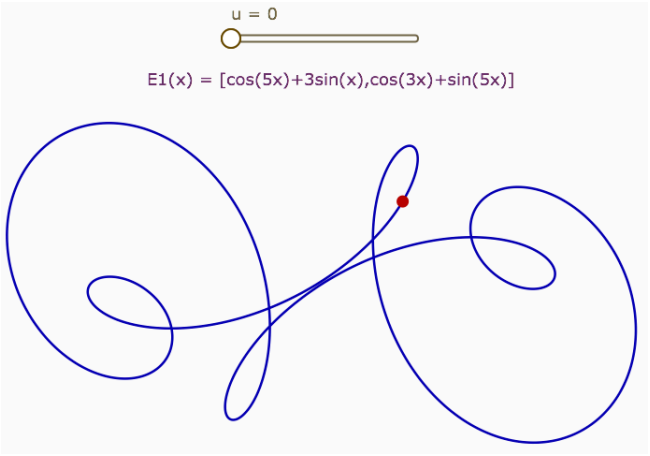
Exercice 5 : tracé d'une fonction du second degré.

a) Créer trois curseurs ***a*** (de -2 à 2), ***b*** (de - 3 à 3) et ***c*** (de -10 à 10) en leur donnant des conditions initiales. On les crée d'abord - ne pas oublier de les **valider**, puis les **déplacer** avant une nouvelle création - et on les nomme ensuite tous les trois avec l'item de tableau de bord **inspecteur d'objet**, ou en le choisissant dans la palette de comportement.
Dans la suite nous allons voir ce que signifie l'anticipation des constructions dans un module de représentation graphique bien fait.
b) reprendre une expression et taper **a*x** - regarder ce qu'il se passe - **confirmer la construction de la courbe** en sélectionnant l'icône associée.
c) afficher le repère, passer en mode consultation pour déplacer la figure (glisser au doigt) et modifier ***a***. revenir en mode standard.
d) on sélectionne la courbe, et on complète le **a*x** initial en **a*x^2** (avec le **clavier math** de DGPad plutôt que le clavier de l'ordinateur). Observer la mise à jour.

e) ajouter **+b**. Regarder la modification, puis continuer ***x**. Observer que *chaque fois que l'expression a un sens algébrique, le module graphique met à jour le tracé de la fonction*. Puis terminer par **+c**. Attention il faut valider à la fin de la modification de l'expression.

Exercice 6 : utilisation des listes pour les courbes paramétrées.
DGPad est un logiciel qui utilise beaucoup les listes. Une liste de deux termes est en général un point. Si ce point contient une variable automatiquement reconnue par DGPad cela devient une fonction paramétrée en 2D.

a) Créer une expression comme **[cos(5x)+3sin(x), cos(3x)+sin(5x)]**, donner des bornes (0 et 2*π). Tracer la courbe avec l'icône qui est apparue.
b) Ajouter un curseur, u, qui va lui aussi de 0 à 2π et construire le point E1(u) si E1 est le nom de l'expression précédente. Vous aves la courbe et un point mobile.
Remarque : la méthode n'est pas encore implémentée en 3D mais des figures déjà préconstruites permettent de le faire.

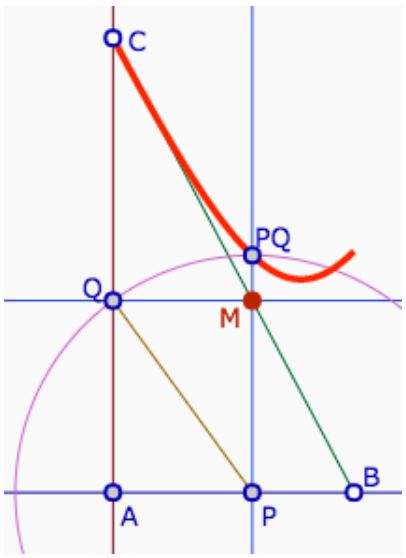


c) Prolongement pour jouer. On considère la fonction f(x) = 2-sin(7x)+0.5cos(30x). Tracer la courbe paramétrée g(x) = [f(x)*cos(x),f(x)*sin(x)], le tout toujours sur [0 2π].
Remarque : *on voit sur ces deux exemples qu'entre un nombre et une variable ou une fonction il n'y a pas besoin de mettre un signe de multiplication mais qu'entre deux variables ou fonction c'est bien entendu nécessaire.*

Exercice 7 : Lieu et minimum d'une fonction.
ABC est un triangle rectangle en A - on prendra (AB) horizontal - et M un point de l'hypothénuse.
Les projections orthogonales de M sur les côtés du triangle forment un segment [PQ]. On s'intéresse à la position de M pour que [PQ] soit de longueur minimale.
a) Argument géométrique ?
b) Complément algébrique : si on note b=AC - *dans DGPad créer la variable b=d(A,C)* - et c=AB, et si A est l'origine du repère, alors pour x=AP on a

$$PQ = \sqrt{\left(1+\frac{b^2}{c^2}\right)x^2 - 2\frac{b^2}{c}x + b^2}$$

Forcer A à être l'origine du repère par A=[0,0], et tracer cette fonction pour voir si elle coïncide avec le lieu.



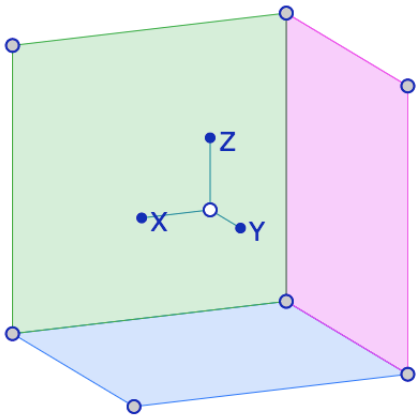
La 3D avec DGPad

Le module 3D se passe souvent dans les macros pour les constructions. Toutefois, comme on va le voir, les palettes contextuelles s'adaptent à la 3D assez souvent (mais pas pour tous les outils bien entendu).

Activité 1 - Découverte des outils - Instrumentation de pratiques efficaces

1.a. Entrer dans le module 3D :
Macro-construction / 3D / Repère 3D / sélectionner l'outil et montrer un point.
Réduire la taille du repère (en mode consultation, avec deux doigts)

1.b. Technique de construction rapide de plans par modification d'une expression.
Rentrer l'expression [-2,-2,2] et la valider. La sélectionner à nouveau et, cette fois, construire un point avec elle.
Puis modifier l'expression pour construire les points [2,-2,2], [-2,2,2], en sélectionnant et enlevant seulement un signe à chaque fois puis en validant par la création d'un point. Modifier l'expression en [2,2,-2] en la validant (sans construire le point, pour avoir désormais -2 en z), puis la sélectionner, créer le point puis les trois autres points : [-2,2,-2],], [2,-2,-2] ,],[-2-,2,-2]. Vous avez 7 points à l'écran. Terminer en construisant trois quadrilatères (outil polygone) qui représenteront 3 plans deux à deux perpendiculaires.



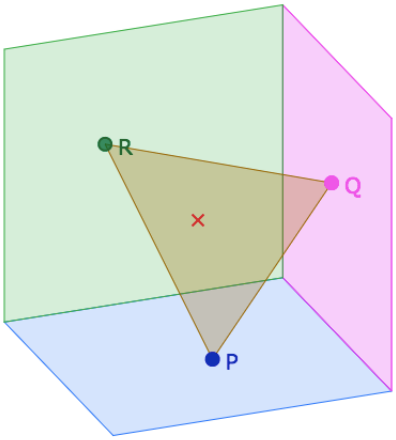
Remarque : *Pour construire les polygones il faut revenir au point initial.*

On peut cacher le trièdre, garder l'origine du repère pour le déplacer. Passer en mode consultation pour vérifier que les plans tournent bien en modifiant l'orientation le trièdre (à un doigt - tablette, trackpad - ou clic gauche glisser - souris sur ordinateur).

1.c. Placer des points dans les plans.

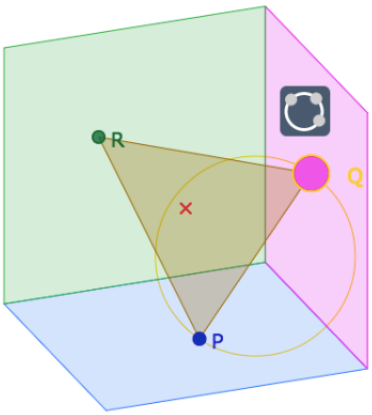
En sélectionnant un plan, il n'y a qu'un objet que l'on peut créer, c'est le « point sur objet », le faire en re-sélectionnant très vite (double clic à la souris) le polyèdre pour que le point soit à l'intérieur du polygone *(point dans)* et non pas sur la frontière *(point sur).*
Construire ainsi un point par plan puis le triangle (polygone) passant par ces trois points.

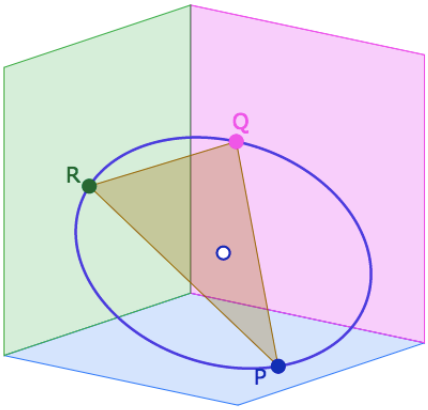
Ci-contre la croix est l'origine du repère.



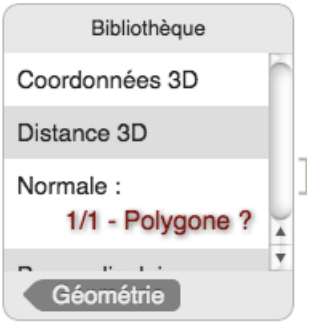
Remarque générale : en 3D, en abandonnant un outil, au début, on peut facilement passer, sans s'en rendre compte du mode standard au mode consultation : pour utiliser un outil de construction, vérifier que vous êtes en mode standard.

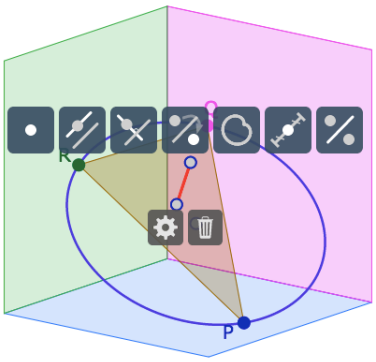
1.d. L'adaptation de la palette contextuelle à la 3D.
Construire, avec la palette contextuelle des points, le cercle circonscrit aux trois points du plan : comme on est en 3D, le même item de palette construit bien l'ellipse attendue, avec son centre comme ci-dessous :

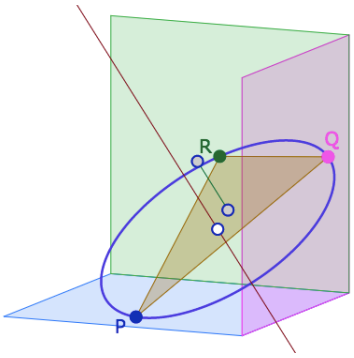




1.e. Axe orthogonal au plan PQR (macro).
Commencer par construire une normale au plan PQR avec les **macro-constructions** : *3D / Géométrie / Normale*.
Puis prendre (palette usuelle) la parallèle à la normale passant par le centre du cercle.

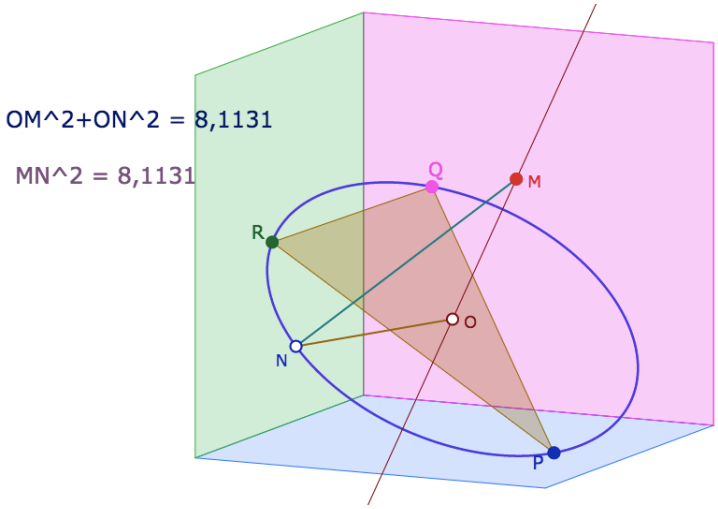






f) Illustrer Pythagore en 3D.

Prendre un point M sur l'axe, N sur le cercle, appeler O le centre du cercle, et calculer les distances (dans l'espace) - toujours avec l'outil **d(,)** du clavier math de DGPad, *qui lui aussi s'adapte*. Vérifier la relation de Pythagore avec une expression qui calcule OM^2+ON^2 et une autre expression qui donne MN^2.



Activité 2 - Les corps rond en perspective axonométrique

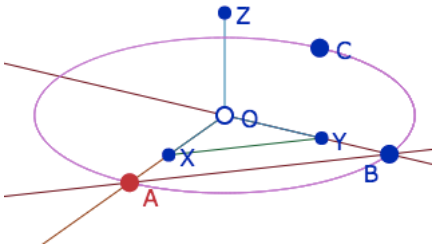
Généralités.

La 3D de CaRMetal ou de DGPad est construite dans une perspective axonométrique orthogonale, un cas particulier des perspectives axonométriques (dites aussi projections parallèles) dans lequel le contour apparent d'une sphère est un cercle, ce qui n'est pas le cas de la perspective cavalière générale : c'est une ellipse.

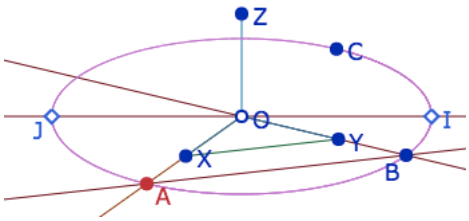
Affichage des coordonnées d'un point.
Pour afficher les coordonnées d'un point, on se place dans la calculatrice et on tape le nom du point. L'expression des coordonnées 3D apparaît, valider, et lui donner un nom.

Partie 1 : les repères sur un cercle du sol

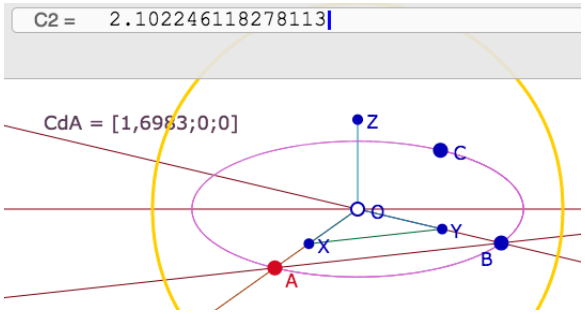
1.a. Construction d'un cercle dans le plan du sol (xOy)
Prendre un repère (nommer O son origine), un point A sur la demi-droite [OX), B par Thalès sur [OY), C est le point s_O(A). Utiliser l'icône « cercle par 3 points ».

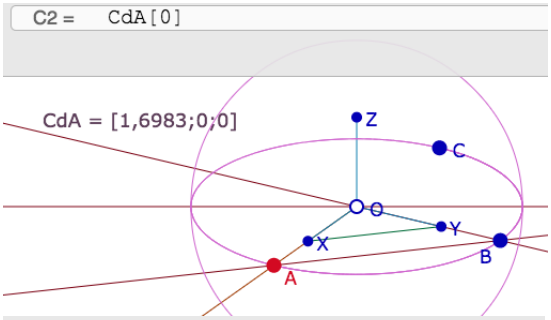


Le problème *: en fait pour le moment, l'outil cercle 3D n'accepte pas d'intersection avec une droite. Donc il faut faire autrement.*
On veut construire le point I intersection du cercle au sol avec la droite intrinsèquement « horizontale » passant par O. Pour cela, il faut mélanger de la 2D et de la 3D, car « intrinsèque ».

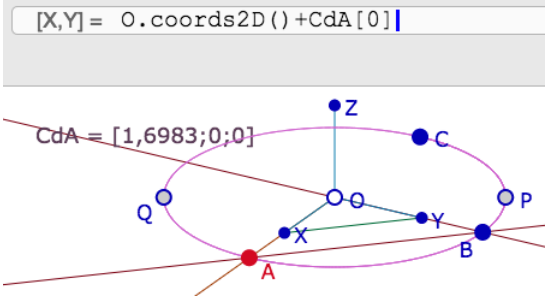


1.b. Commencer par construire la perpendiculaire 2D à la droite (OZ) en O.
Puis construire un cercle 2D de centre O de rayon quelconque (à gauche ci-dessous) et on lui donne comme rayon (*tap* Expression sur le cercle) la valeur de la première coordonnée de A (illustration de droite).
- Cacher le cercle 3D pour construire les intersections entre le cercle 2D et la droite 2D, ce sont les points I et J de l'illustration précédente.

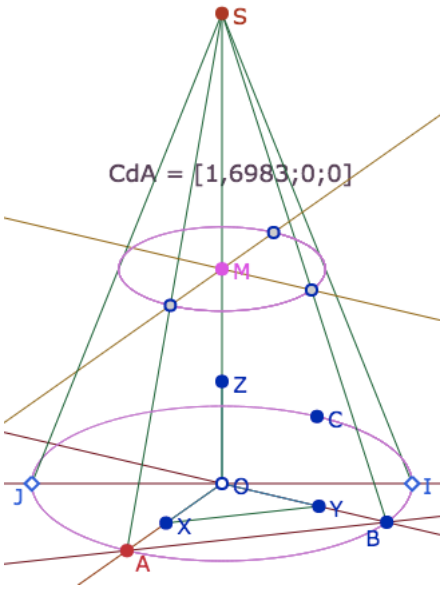




Complément plus technique pour info.
On peut aussi *mélanger la 2D avec la 3D* en travaillant directement sur les coordonnées. Ci-contre, on force le point à être construit en 2D, en prenant les coordonnées 2D du point O et en ajoutant un nombre (qui correspond à ajouter une abscisse, en 2D : c'est le point P.



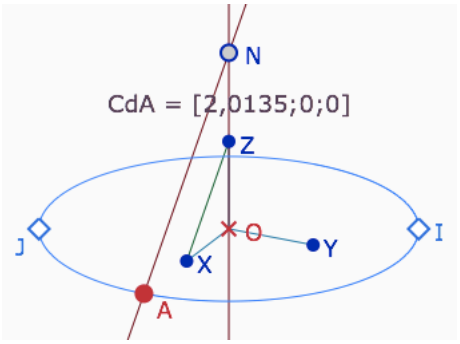
Partie 2 : Construction d'un cône.



2.a. Finir le cône avec quelques génératrices, à partir d'un point S sur la droite (Oz).
2.b. Puis choisir un point M sur le segment [OS] pour visualiser une section plane, parallèle au plan du sol. Construire cette section.
2.c. Vérifier les relation de Thalès en 3D en utilisant simplement l'outil distance 3D … c'est-à-dire l'outil **d(,)** du clavier de DGPad, en montrant les points.

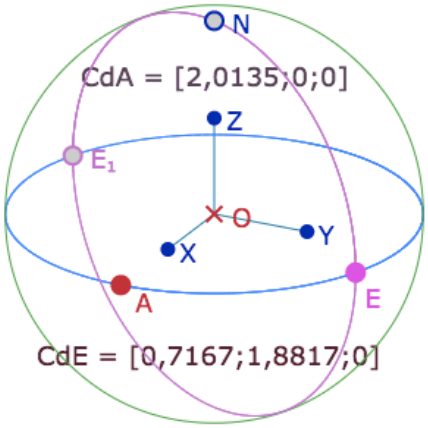
Conserver la figure pour la suite

Partie 3 : Le contour apparent d'une sphère.
La figure avec la sphère préconstruite peut être donnée aux élèves pour construire la section par un plan et utiliser Pythagore. Mais elle peut aussi être construite - ou partiellement terminée - par les élèves, en donnant des indications, comme ici.



3.a. Commencer par supprimer le point **S**. Cela supprime le cône, et la base du cône, avec ses points **I** et **J**, devient l'équateur de la sphère. Construire le pôle nord de la sphère en construisant le point **N** .

On peut construire **N** par Thalès comme ci-contre ou simplement en entrant le point de coordonnées [0,0,*CdA*[0]] pour mettre x_A en Z.



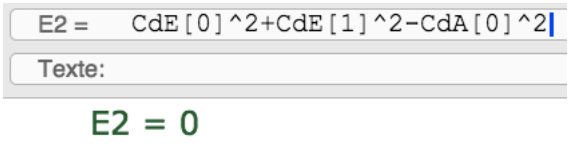
3.b. Le contour apparent dans cette perspective est un cercle : c'est le cercle - du plan de l'écran - de centre O passant par I. C'est une des raisons pour laquelle cette perspective est souvent utilisée (marine, dessin, manuel scolaire)
3.c. Premier méridien. Prendre un point **E** de l'équateur et **E_1** son symétrique par rapport à ***O***. Le cercle de centre ***O*** passant par **E** passe aussi par **N**. D'où un premier méridien de la sphère en construisant le cercle passant par les trois points **E**, **N** et **E_1**.

Déplacer **E** sur l'équateur pour voir que le contact avec le contour apparent décrit tout le contour apparent …

Remarque technique : *à chaque fois que l'on crée un cercle 3D avec l'icône « cercle par 3 points », l'item renvoie aussi le centre du cercle, souvent utile. Ici ce centre est souvent sur O, on le cachera : le point O en croix rouge est là pour nous le rappeler (ou le rappeler aux élèves).*

3.d. Intermède - question pour les élèves

En nommant **CdE** les coordonnées de **E**, pourquoi la somme des carrés des coordonnées de **E** est égale à la somme des carrés des coordonnées de **A** ?



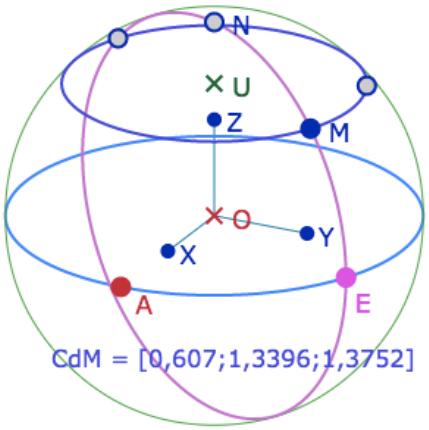
Partie 4. Réalisation d'une section. Vérification par Pythagore.

4.a. Construction d'un parallèle à l'équateur.

Soit **M** un point du méridien précédent, on se propose de construire le parallèle à l'équateur passant par **M**.

Afficher les coordonnées de **M** en déduire deux autres point du cercle cherché, terminer le cercle, et nommer son centre **U**.

Déplacer **M** pour voir que le cercle méridien est contenu dans le contour apparent de la sphère.



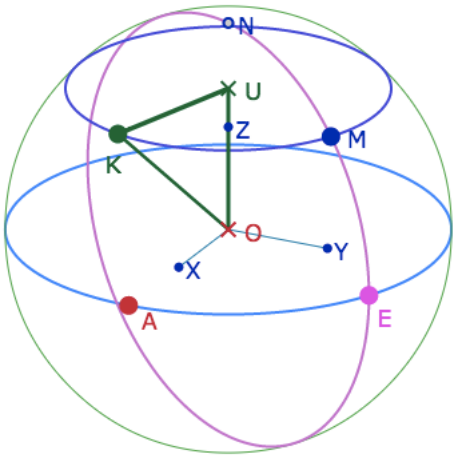
4.b. Utilisation de Pythagore dans l'espace.

Soit **K** un point du méridien construit à l'étape précédente, vérifier, par calcul :

i) que **K** est bien un point de la sphère,

ii) avec Pythagore, que le triangle **OUK** est rectangle.

Manipuler **K**, **M**, et **E** pour contrôler que les vérifications sont bien valides.



Quelques autres figures 3D, en ligne, réalisées avec DGPad

Perpendiculaire commune à deux droites : http://goo.gl/CyYcUc
Trois pyramides dans un cube : http://goo.gl/Y36pQz
Les hauteurs d'un tétraèdre (Hs est aimanté): http://goo.gl/WEz0jV
Volume d'un prisme (réalisation Monique Gironce) : http://goo.gl/0QYFwJ
Thales en général dans l'espace : http://goo.gl/RgORJG
Cerceau et théorème de Huygens : http://goo.gl/hxD8pq
Les 11 patrons du cube : http://goo.gl/ea8psJ
Ruban de Moebius à k tours (pour le fun) : http://goo.gl/3xqpN3

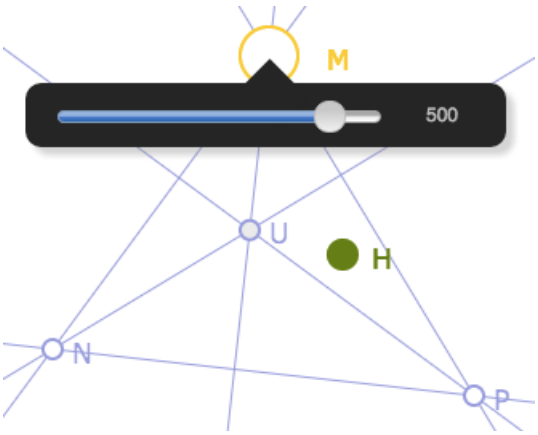
Trajet minimale sur un parallélépipède

Figure générale : http://goo.gl/WuSD9X
Cas 3 patrons donnant la même longueur d12=d31=d32 : http://goo.gl/DYaz3z
Cas d13=d31 et d12=d21 : http://goo.gl/OLQgOc

Compléments techniques sur DGPad (oct 2015 – avant Blockly)
Aimantation - Programmation de listes - Widgets - Attributs conditionnels

Activité 1. Aimantation et programmation booléenne de points. (simulation pour le collège)
Activité de simulation, réalisée par les enseignants pour la classe
On se propose de faire une figure simplifiée de cette figure : http://goo.gl/rqvUA7
Il s'agit de faire réfléchir les élèves sur le fait que si H est orthocentre de ABC, alors tout point de la figure est orthocentre des trois autres points … car la figure des droites est des perpendiculaires reste fixe.
• Mathématiquement, on joue sur la symétrie de la relation perpendiculaire.
• Visuellement, on va préparer une figure où ce sont les points qui bougent sur une configuration fixe.

Etape 1 : Aimantation préalable



1.a. construire un triangle MNP et U son orthocentre (avec les 3 hauteurs).
Construire U comme intersection de deux hauteurs, puis construire la troisième hauteur ensuite.
1.b. Prendre un point H (plein rouge), et aimanter H sur les 4 points M, N, P, U avec une aimantation de 500 pixels.

Etape 2 : construction des sommets conditionnés (ou booléens).
Désormais on veut construire A, B, C - respectivement sur M, N et P, sauf si H est dessus le point. Par exemple A sera sur M sauf si H est sur M et alors A sera sur U.
Comme les points sont aimantés, la distance de H au point sur lequel il se trouve est nulle. On peut donc utiliser cette propriété de distance nulle ou pas pour rendre les points « comme aimantés » (mais en fait « rendu booléens).
Autrement dit A est le point **U*(d(H,M)==0)+M*(!d(H,M)==0)**
C'est-à-dire *A est en U si H est en M et A est en M sinon*
2.a. Organisation de l'écriture (car bug DGPad sur ambiguïté dans les expressions)
Pour le point A
• Cacher P et placer H sur P.
• Créer un point A. « Tap » sur le point et ouvrir expression (icône calculatrice), supprimer les coordonnées (le point A disparaît, et mettre à la place l'expression précédente (utiliser le clavier math) sans le ! qui n'existe pas. Placer le ! avec le clavier standard après avoir tout tapé. Valider.
2.b. Faire de même pour B et C (on peut aussi, avec le clavier standard copier coller la formule précédente et modifier les termes, ce qui évite de déplacer H et cacher le point dessous).
Pour B : **U*(d(H,N)==0)+N*(!d(H,N)==0)**
Pour C : **U*(d(H,N)==0)+N*(!d(H,N)==0)**
2.c. Cacher finalement les 4 points M, N, P, U. (Déplacer H pour cacher le point qui est dessous). Construire et colorer le triangle ABC.
La figure correspond bien à ce que l'on voulait.

Activité 2. L'objet « liste » de DGPad. Programmation des listes (simulation en lycée)

Les expressions de DGPad peuvent contenir des programmes. Nous ferons un seul exemple, essentiellement pour montrer la méthode et aborder le traitement des listes,

Le Thème : la modélisation du cœur de tournesol. Elle est donnée par les points P_k d'affixe $C\sqrt{k}e^{ik\theta}$ où θ est lié au nombre d'or Ψ par $\theta = \frac{2\pi}{1+\Psi}$, soit environ 137,5°.

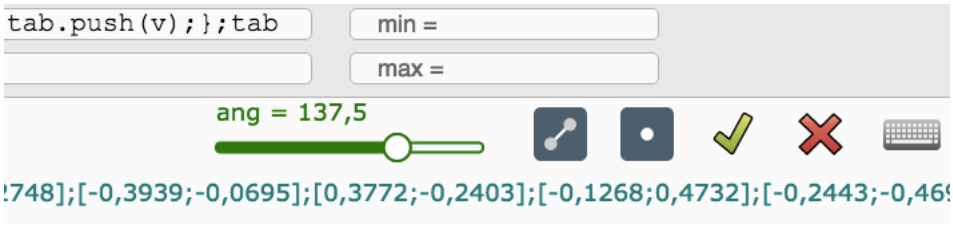
1.a. Commencer par créer un curseur nommé ang, qui va de 120 à 160. Ce sera un angle en degré qu'il faudra transformer en radian pour l'écriture polaire de l'écriture complexe en forme cartésienne. Dans le cadre suivant, l'écriture de la construction. Elle sera étudiée et commentée en détail ensemble.

```
var tab=[];for (var k=1;k<=500;k++){var u=0.2*sqrt(k);var a=k*ang*π/180;
var v=[u*cos(a),u*sin(a)];tab.push(v);};tab
```

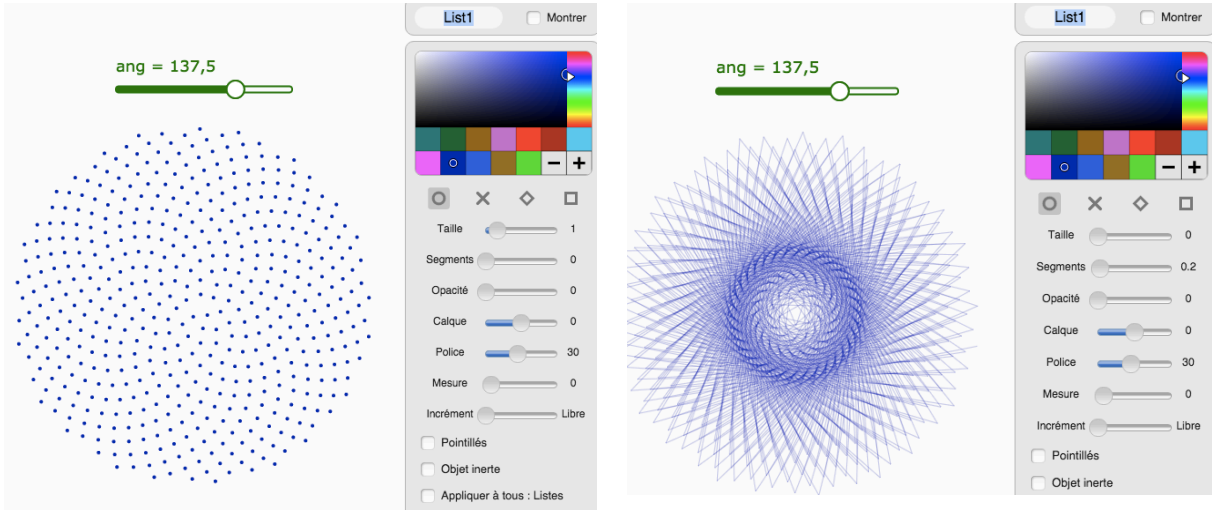
La syntaxe d'une expression-programme est la suivante

- on crée une liste en interne (en JavaScript), tout le JS est disponible avant le dernier « point virgule »(avec des simplifications comme sqrt() ou π à la place de Math.sqrt() et de Math.PI.
- cette liste est renvoyée à DGPad par la variable placée après le dernier « ; ».

1.b. Taper cette expression, éventuellement dans un fichier texte et le copier coller (le copier coller se fait sous clavier standard, pas clavier math de DGPad).
S'il n'y a pas de problème de syntaxe l'expression devient une longue liste de 500 points (crochet de 2 coordonnées). Il apparaît alors deux icones :



Choisir le segment, puis aller dans l'inspecteur d'objets : on peut y régler la taille des segments et des points au dixième de pixels, les deux ne pouvant être nuls
(à gauche *segment* est à 0, à droite *segment* est à 0,2 et *point* à 0).



1.c. Agir ensuite sur le curseur …

Activité 3. L'outil Widget

L'item widget est la réunion de trois outils : du HTML (bouton JS), l'implémentation du KaTeX (bouton TeX) et la possibilité de lancer des scripts (bouton exp).

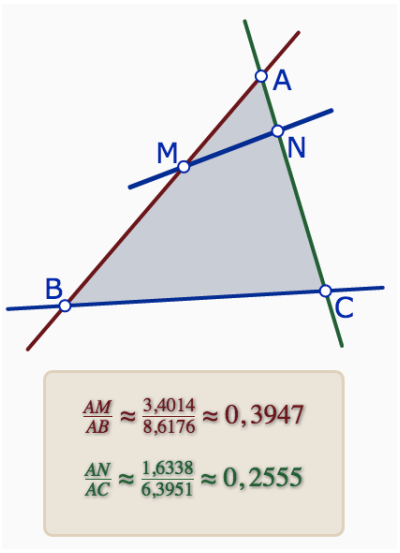
Si tous les trois peuvent être mélangés, cela demande de la pratique. Pour le moment, les outils ne sont pas encore assez documentés. Pour cette troisième activité, on se limitera à explorer le TeX, et à l'enrichir (couleur) en HTML.

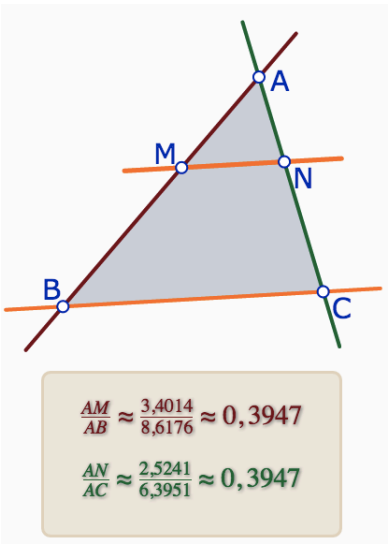
TeX

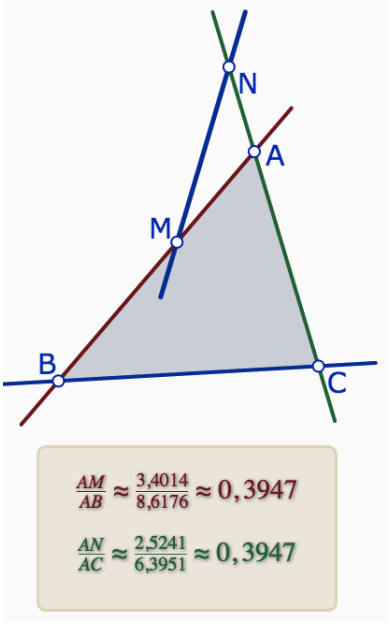
L'outil le plus simple est le LaTeX pour écrire quelques formules mathématiques. KaTeX est une implémentation récente en JavaScript, 30 fois plus rapide que MathJax utilisé précédemment … mais qui ne contient pas encore tous les items LaTeX (et en particulier ne contient pas *\overrightarrow{}*, utilisé pour des vecteurs à deux lettres comme $\overrightarrow{AB}$).

On peut mettre aussi des valeurs numérique dans un texte en TeX, en mettant une variable entre deux %.

Nous allons réaliser en détail cette figure où N est aimanté par les deux solutions qui réalise les égalités de rapport :







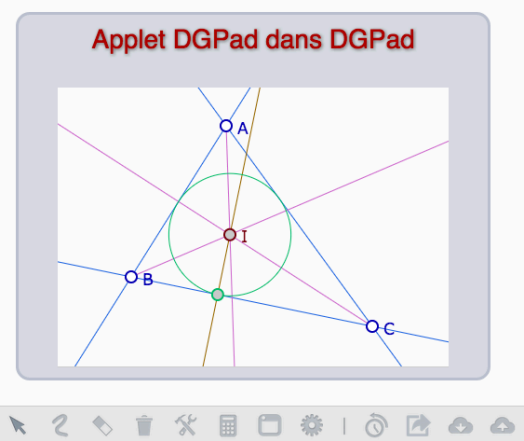
1.a. Réaliser la figure de base avec N aimanté par les deux solutions.

1.b. Ajouter un widget qui comprend le code suivant (sans retour à la ligne) qui contient deux fois un code couleur, puis une ligne de TeX avec la valeur approchée.

```
<span style="font-size:24px;color:#760011">
$\frac{AM}{AB}\approx\frac{%d(A,M)%}{%d(A,B)%}\approx%d(A,M)/d(A,B)%$</span>
<span style="font-size:24px;color:#006633">
$\frac{AN}{AC}\approx\frac{%d(A,N)%}{%d(A,C)%}\approx%d(A,N)/d(A,C)%$
</span>
```

Insérer une figure DGPad dans un widget.

Probablement un des intérêts didactiques les plus immédiats pour des widgets de base, par exemple pour donner un travail à faire - une figure à construire - dont le modèle peut être manipulable, sans pouvoir être copiable.



Principe :

- Faire la figure,
- L'exporter en version en « html+JS »
- Copier TOUT le code
- L'insérer dans un widget, tout simplement

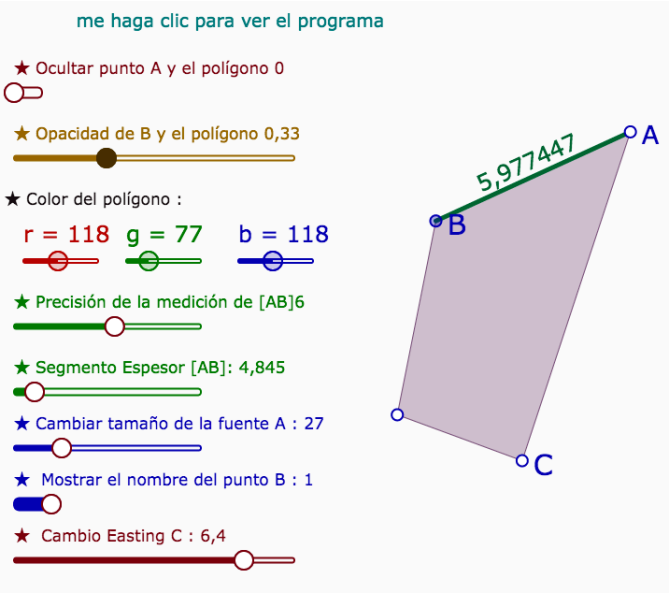
Remarque : si on veut mettre un titre comme ci-dessus, ajouter une ligne du type :
`<span style='font-size:28px;color:#AA0000'>Applet DGPad dans DGPad</span>`

Activité 4. Les attributs conditionnels

Proposé ici juste pour information, pour commenter éventuellement si on a du temps. La figure manipulable est dans votre dossier.

Voici les principaux attributs paramétrables

```
A.setHidden(E1);
Poly1.setHidden(E1);
Poly1.setOpacity(E3);
B.setOpacity(E3);
Poly1.setRGBColor(r,g,b);
S1.setPrecision(E5);
A.setFontSize(E6);
B.setShowName(E7);
C.setxy(E8,C.gety());
S1.setSize(E9);
" me haga clic para ver el programa"
```



- Dans l'expression programme on ne va pas à la ligne (ici juste pour la lisibilité)
- Elle doit toujours se terminer par un commentaire entre guillemets qui est le corps visible de l'expression dans la page DGPad.

Autre fiche – autres pistes - autres figures
(toujours avant l'animation dans DGPad et Blockly)

Partie 1. Les expressions de DGPad.

1.1. Tracé de fonctions.

Le mode expression permet de réaliser beaucoup de choses. Les curseurs et les fonctions, y compris les courbes paramétrées (2D seulement pour le moment).

1.1.a. Considérer la fonction f, ci contre, définie sur [0, 2π[. Ne pas la tracer, conserver seulement son expression. Puis la fonction g, que l'on tracera.

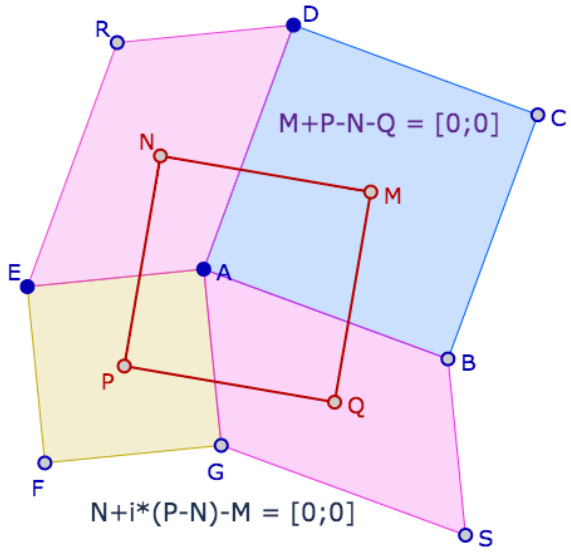
```
f(x) = 2-sin(7x)-0.5cos(30x)
g(x) = [f(x)*cos(x),f(x)*sin(x)]
```

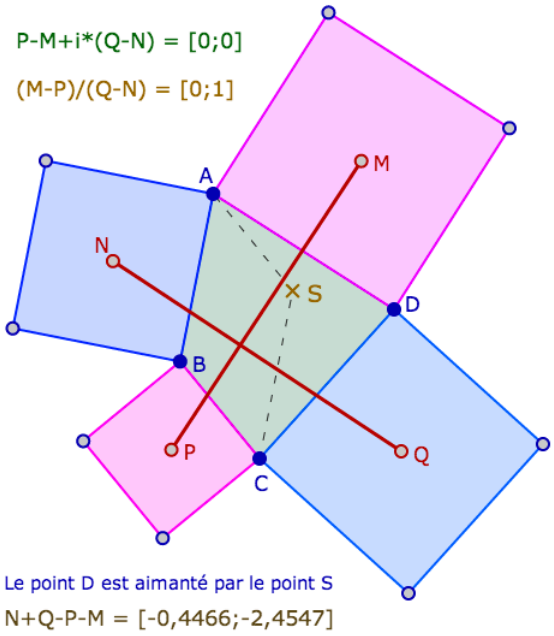
1.1.b. Ajouter ensuite un curseur u, de 0 à 2π, et placer le point g(u).

***Syntaxe** : on remarquera qu'il n'est pas nécessaire de mettre un signe de multiplication entre un nombre et une variable, mais qu'il faut le mettre entre deux variables ou fonctions.*

1.2. Utilisation formelle des points dans les constructions. Utilisation des complexes. On se propose de travailler sur ces deux exercices, très classiques (mais désormais plus vraiment au programme de lycée). Voir la nuance entre l'usage d'expressions formelles utilisé dans les calculs et l'exercice traité en calcul formel (par exemple avec wxMaxima)

1.2.a. Illustration de gauche. On part de trois points A, D, E. On construit (algébriquement) les deux parallélogrammes, puis, avec des complexes, les deux carrés. On vérifie en deux étapes que les centres des figures partielles forment un carré. Voir en quoi les expressions produites sont des vérifications numériques.



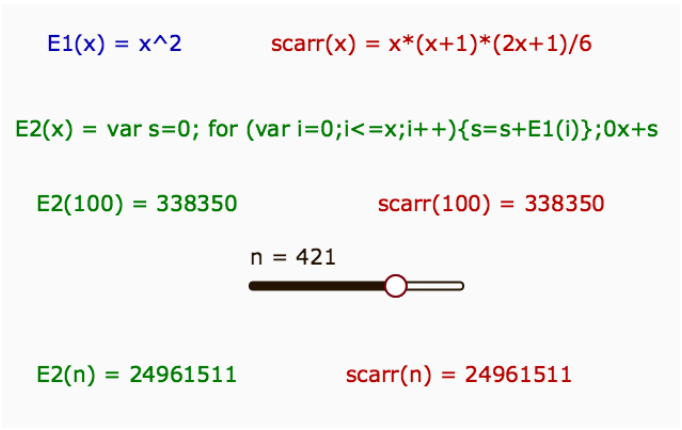


1.2.b. Illustration de droite (et réflexions sur les écritures utilisées). On part cette fois d'un quadrilatère quelconque ABCD. On s'intéresse aux diagonales [PM] et [NQ] et au cas particulier (aimantation) où ABCD est un parallélogramme.

Partie 2 : les programmes-expression. L'objet list

Règle des programmes-expression
Le dernier point virgule est une séparation :
• ce qu'il y a avant est du JavaScript « pur », avec quelques simplifications (pas de Math.cos mais juste cos par exemple etc - utilisation du clavier math de DGPad).
• ce qu'il y a après le dernier point virgule est du DGPad et est dans DGPad, c'est la sortie.

Cela signifie qu'il n'y a aucune extension géométrique du JavaScript (comme Point, Line, Segment, etc, des CarScripts de CaRMetal) dans les expressions programme. Mais, comme on va le voir plus loin, cela n'empêche pas de proposer des créations de listes de points.



2.1. *Exemple simple* (analyse en séance) : la somme des carrés, avec un curseur.

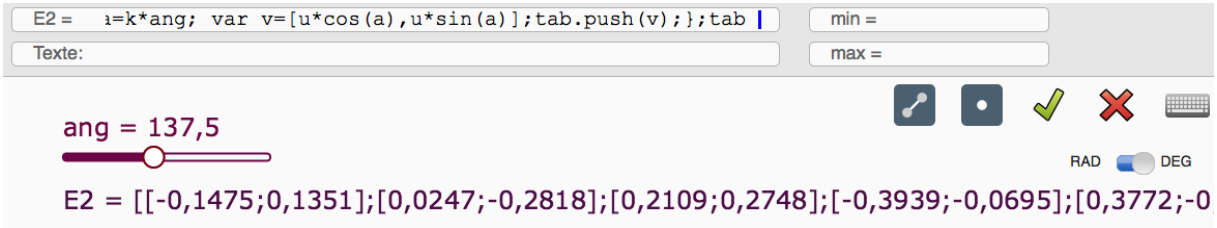
2.2. L'objet list. On reprend la modélisation du tournesol. Elle est donnée par les points P_k d'affixe $C\sqrt{k}e^{ik\theta}$ où θ est lié au nombre d'or Ψ par $\theta = \dfrac{2\pi}{1+\Psi}$, soit environ 137,5°.

2.2.a. Commencer par créer un curseur nommé **ang**, qui va de 120 à 160. Ce sera un angle en degré. Une version récente (encore seulement en ligne) introduit le choix degré/radian dans la figure, donc il n'y a pas à convertir en radian, ce qui serait nécessaire dans les versions actuelles sur tablette. Dans le cadre suivant, l'écriture de la construction. Elle sera étudiée et commentée en détail ensemble.

```
var tab=[];for (var k=1;k<=500;k++){var u=0.2*sqrt(k);var a=k*ang;
var v=[u*cos(a),u*sin(a)];tab.push(v);};tab
```

La syntaxe pour un objet **list** est la suivante
• on crée une liste en interne (en JavaScript), tout le JS est disponible avant le dernier « point virgule »(avec des simplifications comme sqrt() ou π à la place de Math.sqrt() et de Math.PI.
• cette liste est renvoyée à DGPad par la variable répétée après le dernier « ; ».

2.2.b. Taper cette expression, éventuellement dans un fichier texte et le copier coller (le copier coller se fait sous clavier standard, pas clavier math de DGPad).
S'il n'y a pas de problème de syntaxe l'expression devient une longue liste de 500 points (crochet de 2 coordonnées). Il apparaît alors deux icones :



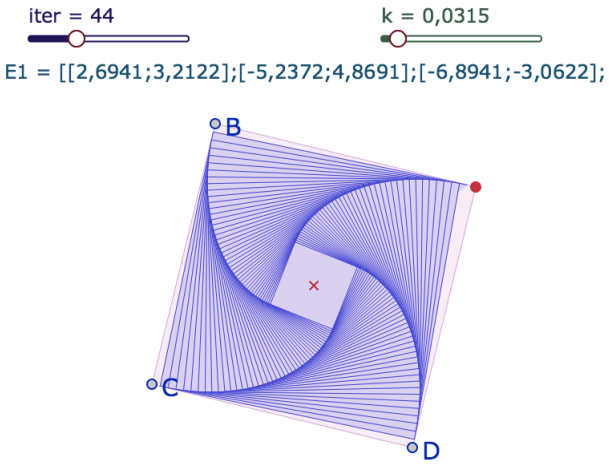
Choisir le segment, puis aller dans l'inspecteur d'objets : on peut y régler la taille des segments et des points au dixième de pixels, les deux ne pouvant être nuls
(la version « tournesol » est avec segments invisibles bien entendu).

2.2.c. Exporter dans un fichier texte (notePad) et observer le contenu. Se rendre compte de l'extraordinaire économie de moyens qui permet une grande réactivité, même sur tablette.

2.3. Application immédiate : itération sur un carré.

2.3.a. On se donne le centre O d'un carré et son sommet A. Commencer par construire les autres sommets du carré (avec les nombres complexes).

On se donne un nombre d'itération et un coefficient k (entre 0 et à,25). Sur chaque côté du carré on se propose de place un point tel que le côté est « multiplié par k », par exemple $\overrightarrow{AM} = k\overrightarrow{AB}$, en itérant le procédé sur les nouveaux côtés obtenus.



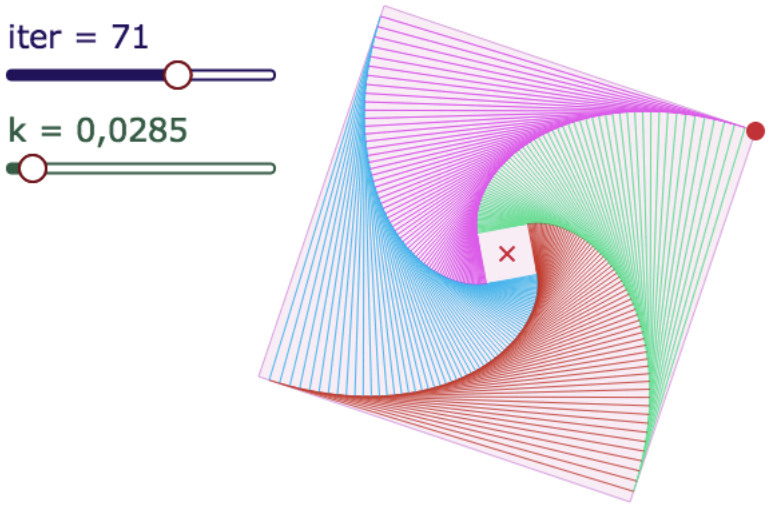
2.3.b. Analyse du code dans une expression programme (donné pour la syntaxe) :

```
var m=A;var n=B;var p=C;var q=D;var tab=[];
for (i=0; i<iter; i=i+1)
   {var m2=[m[0]+k*(n[0]-m[0]),m[1]+k*(n[1]-m[1])];
    var n2=[n[0]+k*(p[0]-n[0]),n[1]+k*(p[1]-n[1])];
    var p2=[p[0]+k*(q[0]-p[0]),p[1]+k*(q[1]-p[1])];
    var q2=[q[0]+k*(m[0]-q[0]),q[1]+k*(m[1]-q[1])];
    tab.push(m2,n2,p2,q2);
   m=m2;n=n2;p=p2;q=q2};
tab
```

Attention : il n'y a pas de retour à la ligne dans le code initial, ici rédigé ainsi pour une meilleure lecture.

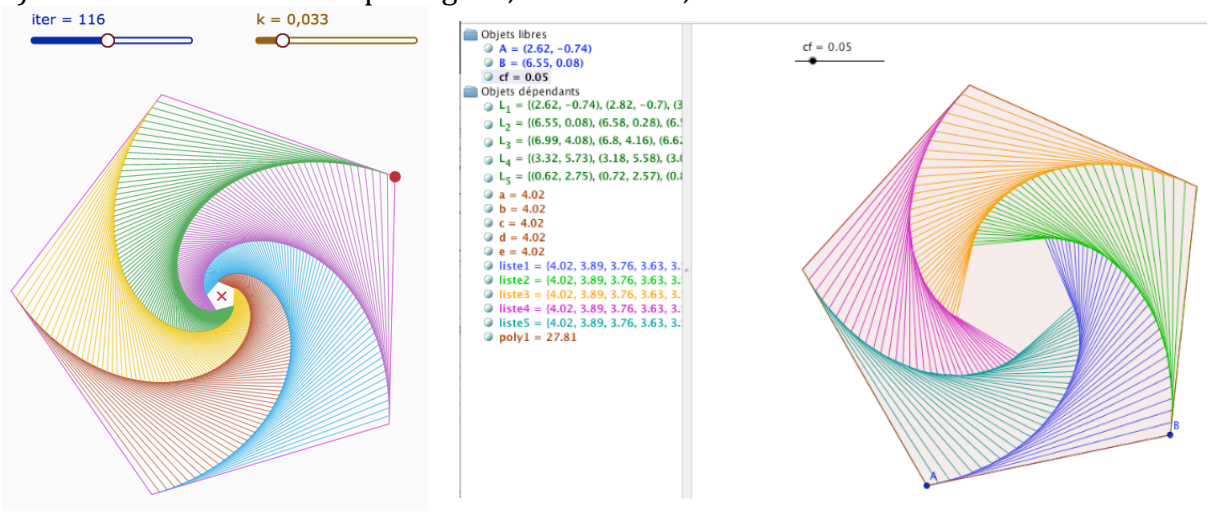
2.3.c. L'utilisation de NaN comme rupture de la chaine.
Il suffit qu'une des deux coordonnées d'un point soit NaN pour que cela arête la suite des points dans la liste : les segments repartent au point suivant.

Reprendre la figure précédente, construire 4 listes extraites de la liste précédente pour leur mettre une couleur spécifique. Pour cela construire chaque segment de la liste, soit deux points, et ajouter un point [NaN, NaN]. Puis modifier les couleurs dans l'inspecteur d'objet.



Exemple de mises en œuvre envisageabes (hors séance juste pour des idées)

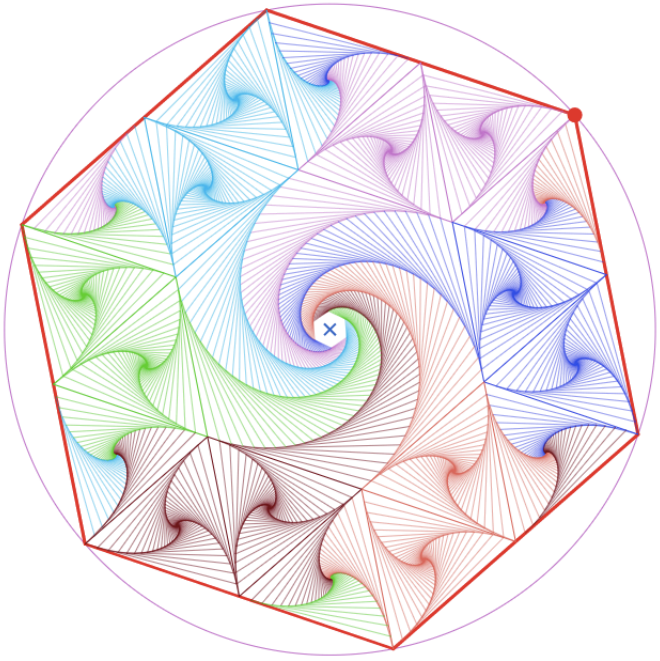
a) La même chose avec un pentagone, sous DGPad, et sous GGB.



b) Autre exemple

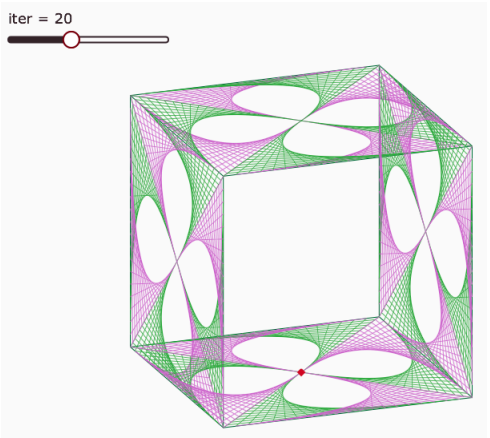
(petit délire …)

https://huit.re/ListeHexa1

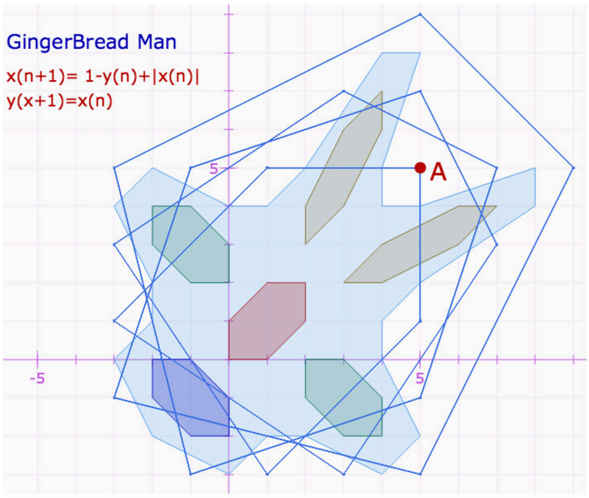


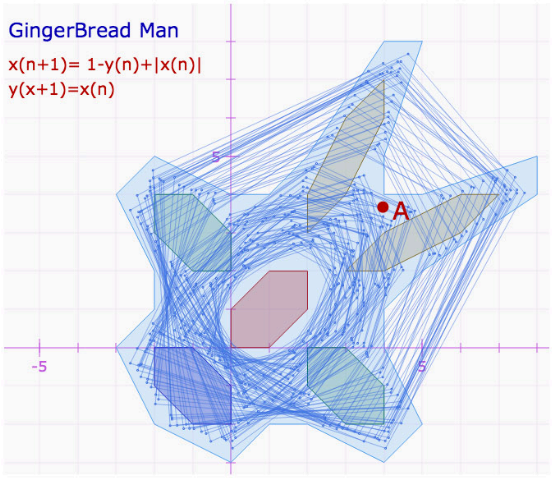
c) sur un cube, comme par exemple ci-contre, mais aussi de nombreuses variantes booléennes

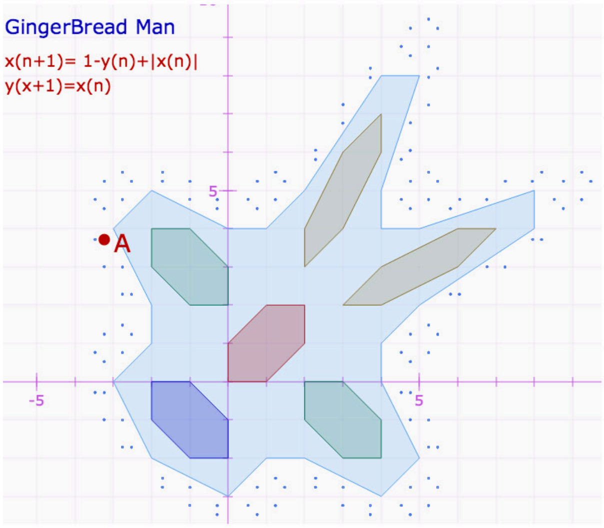
http://goo.gl/jRTAys

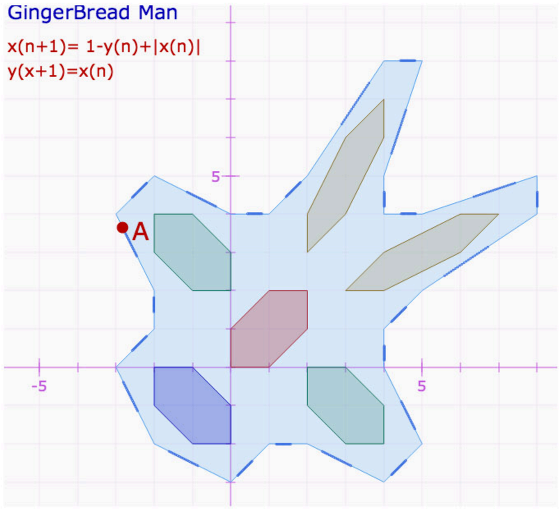


d) les systèmes dynamiques chaotiques, comme GingerBread Man
(selon que l'on place les segments ou les points, on illustre des choses assez différentes)









e) ou des systèmes plus scolaires : itération sur du second degré et sur les fonctions homographiques (désormais bac+1). On sait que le premier est vite chaotique : sur l'illustration on a deux ou 4 valeurs d'adhérence.