progr89
Géométrie analytique

Calculatrices : 89 92+ V200
Niveaux : 1ere S Tale Tale S Sup
Descriptif : Groupe de fonctions - Outils de géométrie analytique :
- Images de points par transformations (rotation, translation, symétrie, homothétie, similitude)
- Equations de droites passant par deux points, par un point avec un vecteur directeur ou normal connu, de la tangente à une courbe
- Equations de cercles passant par des points connus, de diamètre connu

Mots-clefs : transformations, droites, cercles
Date de dernière révision : Septembre 2002

Présentation :

Cette application regroupe un ensemble de fonctions et de programmes dédiés aux calculs de géométrie analytique du plan et un menu personnalisé donnant accès à ces fonctions. Les fonctions, programmées en utilisant le langage intégré de la calculatrice, effectuent leurs calculs en utilisant des valeurs numériques exactes ou des symboles littéraux lorsque ceux-ci n'ont pas été affectés d'une valeur numérique.

Ganalyt est le programme principal. Son rôle est de définir les menus personnalisés de la calculatrice et de simplifier l'accès aux autres fonctions.

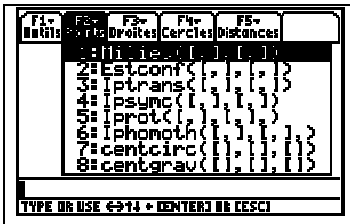
Mode d'emploi :

Si vous connaissez déjà l'utilisation de chaque fonction, vous pouvez naturellement les appeler directement. Si ce n'est pas encore le cas, alors tapez `ganalyt()` (après vous être placé dans le répertoire contenant tous les fichiers) : vous serez guidé par des menus qui vous indiqueront ce que vous devez taper pour utiliser les fonctions.

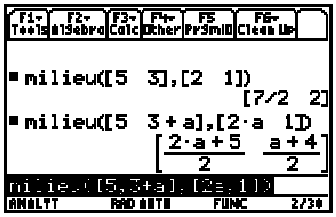
Une fois ce programme lancé vous disposez de 4 menus :

[F1] : Outils
Divers outils (Effaçage de l'écran HOME, sortie du programme).

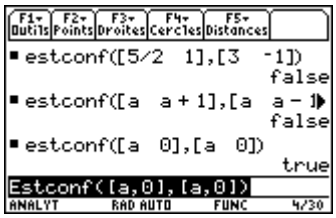
[F2] : Points
Toutes les fonctions de ce menu ont pour caractéristique de retourner un point, c'est-à-dire une matrice d'une ligne et de deux colonnes.



Fonction *milieu* : donne le milieu du segment formé par deux points
Ex : On cherche le milieu du segment [AB], avec A(5,3) et B(2, 1), ainsi que celui du segment [CD], avec C(5, 3+a) et D(2a, 1)



Fonction *Estconf* : indique si deux points sont confondus
Voici quelques exemples simples :



Fonctions *ipxxxxx* : image d'un point par la transformation xxxxx
- *iptrans* (p,v) : p représente le point à translater, v le vecteur de translation
- *ipsymc* (p,c) : p représente le point à transformer et c le centre de symétrie
- *iprot* (p,c,t) : p représente le point à transformer, c et t le centre et l'angle de la rotation.

Note : La lettre T est l'opérateur de la transposition
- *iphomoth* (p,c,k) : p représente le point à transformer, c le centre et k le rapport de l'homothétie
- *ipsimili* (p,c,t,k) : p représente le point à transformer, c le centre, t le centre et l'angle de la rotation et k le rapport de l'homothétie

Note : une similitude est la composée d'une rotation et d'une homothétie de même centre.

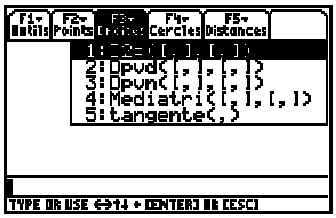
Fonctions centxxxxx : centres divers
- *centcirc* (a,b,c) : centre du cercle passant par les 3 points a, b, c. On utilise la fonction *mediatri* (voir le menu droites) et la fonction intégrée *solve*.

Note : si les points a, b et c sont alignés, la fonction rend le mot clé *undef*.

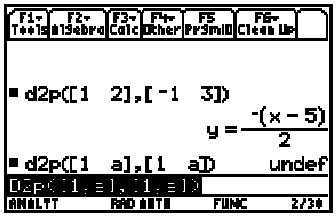
- *centgrav* (a,b,c) : centre de gravité des 3 points a, b, c. Même principe que la fonction milieu

- [F3] : Droites

Les fonctions de ce menu retournent une équation réduite de droite ou le mot clé undef si la droite n'est pas définie.



Fonction *D2p*(a,b) : droite passant par 2 points a et b
Elle retourne une équation réduite si a et b sont distincts, le mot clé *undef* sinon.

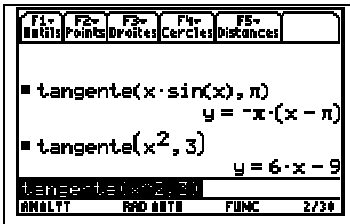


Fonction *Dpvd*(a,v) : droite passant par a et de vecteur directeur v
Elle réutilise la fonction précédente.

Fonction *Dpvn*(a,n) : droite passant par a et de vecteur normal n
Elle réutilise la fonction précédente.

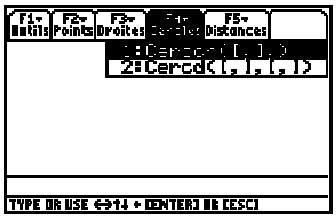
Fonction *mediatri*(a,b) : équation de la médiatrice de 2 points a et b

Fonction *tangente*(ex,xa) : équation de la tangente à la courbe d'equation y=ex au point d'abscisse xa.
On utilise l'expression usuelle y=f'(xa)(x-xa)+f(xa), avec f'(xa) noté ya1 et f(xa) noté ya.



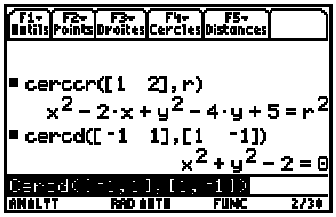
- [F4] : Cercles

Les fonctions de ce menu retournent une équation de cercle en utilisant les lettres x et y comme variables muettes.



Fonction *cerccr*(a,r) : équation du cercle défini par son centre a et son rayon r

Fonction *cercd*(a,b) : équation du cercle défini par son diamètre [a,b].
On aurait pu réutiliser la fonction précédente, mais il est sans doute plus judicieux de faire appel à la fonction intégrée dotP qui retourne le produit scalaire des 2 vecteurs passés en arguments.

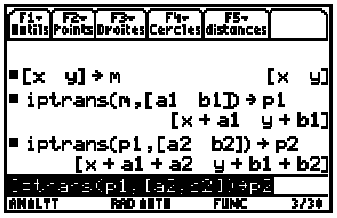


Pour aller plus loin

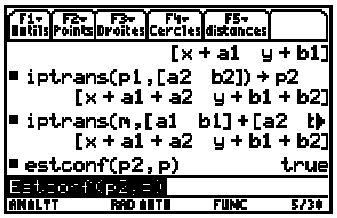
Voyons comment utiliser ces fonctions dans un contexte plus avancé ; que penser des calculs suivants et quelle validité peut-on leur donner ?

<u>Composition de 2 translations</u>

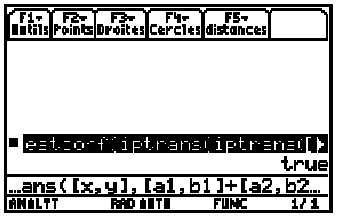
La composée t2ot1 de la translation t1 de vecteur (a1,b1) par la translation t2 de vecteur (a2,b2) est-elle égale à la translation de vecteur (a1+a2,b1+b2) ?
Soit m un point quelconque de coordonnées [x,y].
Soit p1 l'image de m par la translation de vecteur [a1,b1].
Soit p2 l'image de p1 par la translation de vecteur [a2,b2].



Soit p l'image du point m la translation de vecteur [a1,b1]+[a2,b2].
Comparer p2 et p.



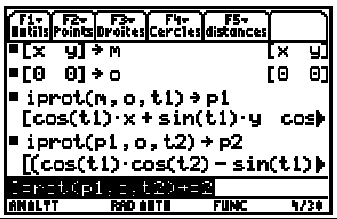
Remarque : On aurait pu se dispenser de toutes ces étapes et poser le calcul suivant
`estconf(iptrans(iptrans([x,y],[a1,b1]),[a2,b2]),iptrans([x,y],[a1,b1]+[a2,b2]))`,
qui est sans doute plus élégant car il ne nécessite aucune variable intermédiaire, mais ne tient pas à l'écran :



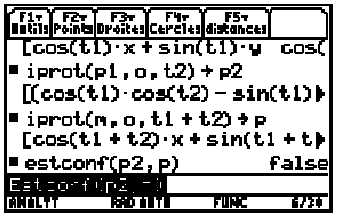
On pourrait faire de même pour étudier la composée de 2 homothéties de même centre.

<u>Composée de 2 rotations de même centre</u>
Pour simplifier l'expression des calculs, on peut supposer, à un changement d'origine près, que le centre des rotations en question est l'origine du repère.
On considère r1(O,t1) et r2(O,t2). Comparons r2or1 et r(O,t1+t2) ;

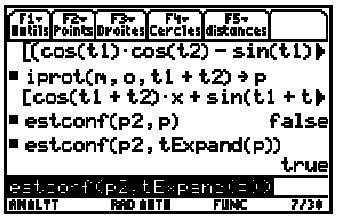
Soit m un point quelconque de coordonnées [x,y], O le point de coordonnées [0,0].
Soit p1 l'image de m par la rotation r1.
Soit p2 l'image de m1 par la rotation r2.



Soit p l'image du point m par la rotation d'angle t1+t2.
Comparer p2 et p.



Aïe !
L'explication en est que la transformation de cos(a+b) ne se fait pas automatiquement et que la calculatrice ne reconnait pas dans ce cas l'égalité des deux expressions exprimées sous des formes différentes.
On peut forcer le développement des expressions de ce type en utilisant la fonction intégrée *texpand* :



Ouf !

Sources :

```
Centcirc (a,b,c)
Func
Local  e1,e2
mediatri(a,b)→e1
mediatri(a,c)→e2
solve(e1 and e2,{x,y})
EndFunc

Centgrav (a,b,c)
Func
1/3*(a+b+c)
EndFunc

Cerccr (a,r)
Func
Local cas
when(r<0,0,1,1)→cas
If cas=0 Then
undef
Else
```

```
(x-a[1,1])^2+(y-a[1,2])^2=r^2
EndIf
EndFunc

Cercd (a,b)
Func
dotP([[x,y]]-a,[[x,y]]-b)=Ø
EndFunc

D2p (a,b)
Func
Local  m,e
[[x,y]]→m
det(augment(m-a;b-a))=Ø→e
when(estconf(a,b),undef,solve(e,y),solve(e,y))
EndFunc

Dist2p (a,b)
Func
norm(b-a)
EndFunc

Distpd (a,e)
Func
Local  xa,ya,cx,cy,ex,cas
part(e,1)-part(e,2)→ex
d(ex,x)→cx:d(ex,y)→cy
when(cx=Ø and cy=Ø,Ø,1,1)→cas
If cas=Ø Then
undef
Else
a[1,1]→xa:a[1,2]→ya
ex|x=xa and y=ya→ex
abs(ex)/(√(cx^2+cy^2))
EndIf
EndFunc

Dpvd (a,b)
Func
d2p(a,a+b)
EndFunc

Dpvn (a,b)
Func
dpvd(a,[[b[1,2],-b[1,1]]])
EndFunc

Estconf (a,b)
Func
Local  xa,ya,xb,yb
a[1,1]→xa:a[1,2]→ya
b[1,1]→xb:b[1,2]→yb
when(xa=xb and ya=yb,true,false,false)
EndFunc

Ganalyt ()
```

```
Prgm
init()
Custom

Title  "Outils"
Item  "ClrHome"
Item  "Quitter()"

Title  "Points"
Item  "Milieu([,],[,])"
Item  "Estconf([,],[,])"
Item  "Iptrans([,],[,])"
Item  "Ipsymc([,],[,])"
Item  "Iprot([,],[,],)"
Item  "Iphomoth([,],[,],)"
Item  "centcirc([],[],[])"
Item  "centgrav([],[],[])"
Title  "Droites"
Item  "D2p([,],[,])"
Item  "Dpvd([,],[,])"
Item  "Dpvn([,],[,])"
Item  "Mediatri([,],[,])"
Item  "tangente(,)"

Title  "Cercles"
Item  "Cerccr([,],)"
Item  "Cercd([,],[,])"

EndCustm
CustmOn
EndPrgm

Init ()
Prgm
DelVar x,y
EndPrgm

Iphomoth (p,c,k)
Func
k*(p-c)+c
EndFunc

Iprot (p,c,t)
Func
Local  m
[[cos(t),sin(t)][-sin(t),cos(t)]]→m
c+(m*(p-c)ᵀ)ᵀ
EndFunc

Ipsimili (p,c,t,k)
Func
iprot(iphomoth(p,c,k),c,t)
EndFunc

Ipsymc (p,c)
Func
```

```
p+2*(c-p)
EndFunc

Iptrans (p,v)
Func
p+v
EndFunc

Mediatri (a,b)
Func
dpvn(milieu(a,b),b-a)
EndFunc

Milieu (a,b)
Func
(a+b)/2
EndFunc

Quitter ()
Prgm
CustmOff
EndPrgm

Tangente (ex,xa)
Func
Local  ya,ya1,e
ex|x=xa→ya
d(ex,x)|x=xa→ya1
y=ya1*(x-xa)+ya→e
solve(e,y)
EndFunc
```