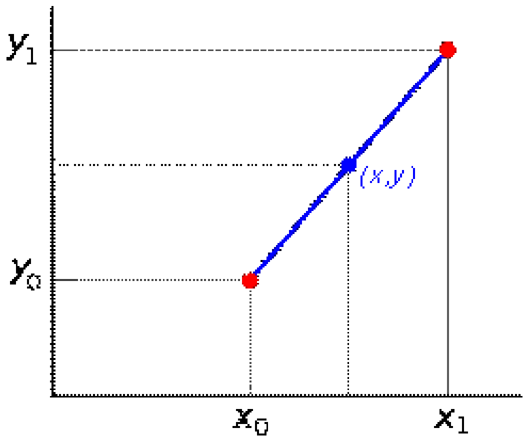
Interpolation et Trigonométrie 2D & 3D

lundi 17 mars 2014 21:48

// Lerp() ou l'interpolation linéaire

L'interpolation linéaire est la méthode, proche de la règle de 3, la plus simple permettant de déterminer une valeur entre deux point lorsque sont incrémentation est linéaire (ou uniforme).

En géométrie l'interpolation linéaire nous permet ainsi de définir la position x,y d'un point C sur la droite AB.



Dans processing ce calcule est simplifié par la méthode suivante :

Lerp(start, stop, inc);

Où **start** est la valeur de départ, **stop** est la valeur d'arrivé et **inc** est le rang, compris entre 0 et 1, où se trouve notre valeur. On dit du rang qu'il est normalisé car compris entre 0 et 1.

Ainsi pour x = lerp(100, 200, 0.5), x = 150 et pour y = lerp(100, 200, 0.25), y = 125.

Cette méthode peut notament permettent de réaliser des attracteurs simple à l'aide de la méthode suivante

Soit ox et oy les coordonées du point d'origine
Soit tx et ty les coordonnées de la cible
Soit x et y les coordonnées du notre point.
Soit progress la position dans le rang entre le point o et t
Soit vp la vitesse de progression défini par un aléatoire entre 0.1 et 0.01

Progress += vp;
X = lerp(ox, tx, progress);
Y = lerp(ox, oy, progress);

Il est possible d'effectuer des interpolations sur des distances en courbe de Catmull-Rom ou courbes de bézier à l'aide des méthode c**urvePoint() et bezierPoint.**

Il faut toutefois noté que là où **Lerp(start, stop, inc);** attend 3 valeurs, les méthodes **curvePoint()** et **bezierPoint()** en attendent 5 à savoir

curvePoint(coordonnée 1er pt, coordonnée 2nd pt, coordonnée 3e pt, coordonnée 4e pt, inc);

bezierPoint(coordonnée 1er pt ancrage, coordonnée 1er point de control, coordonnée 2nd point de control, coordonnée 2nd point d'ancrage);

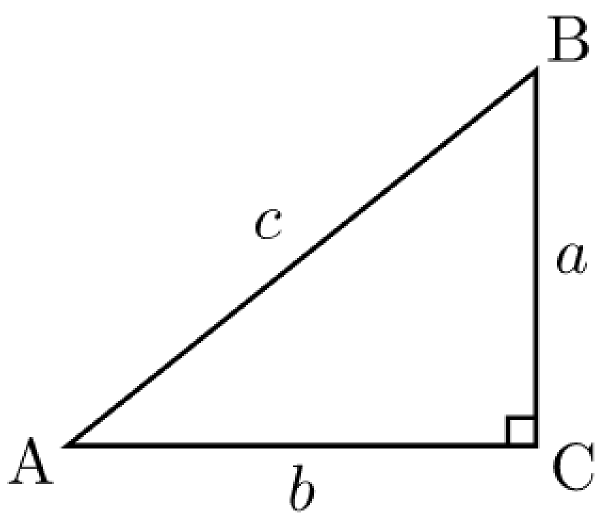
NB : Pour rappel ces deux courbes sont définies par des algorythmes différents

La courbe de Catmull rom est un courbe passant par les point d'une ligne polygonal

La courbe est de bézier est définie par des tangeantes et des angles

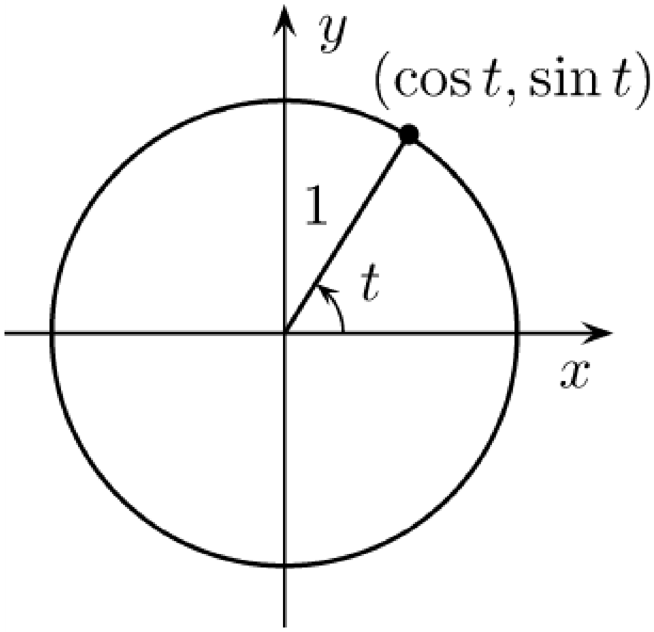
// Trigonométrie 2D

1- Rappel :
  a. La trigonométrie traite de la relations entre les distances et les angles au sein d'un triangle, et notamment au sein du triangle rectangle.
  b. Soit un triangle ABC "droit" en C alors :
    i. Le total de ses angles = **180°**
    ii. L'angle ACB est la valeur la plus grande
    iii. Le côté AB situé à l'opposé de l'angle ACB est **l'hypothenuse** et se trouve être la distance la plus grande du triangle
    iv. Si on regarde de plus près l'angle ACB alors
      1) Sin(ACB) = a/c;
      2) Cos(ACB) = b/c;
      3) **Tan(BAC) = a/b**;



Ici nous prêterons notamment attention à ces dernière fonctions sinus, cosinus et tangeante

Lorsque nous traçons un cercle de centre A et de rayon AB(hypothenuse) nous obtenons un cercle trigonométrique.

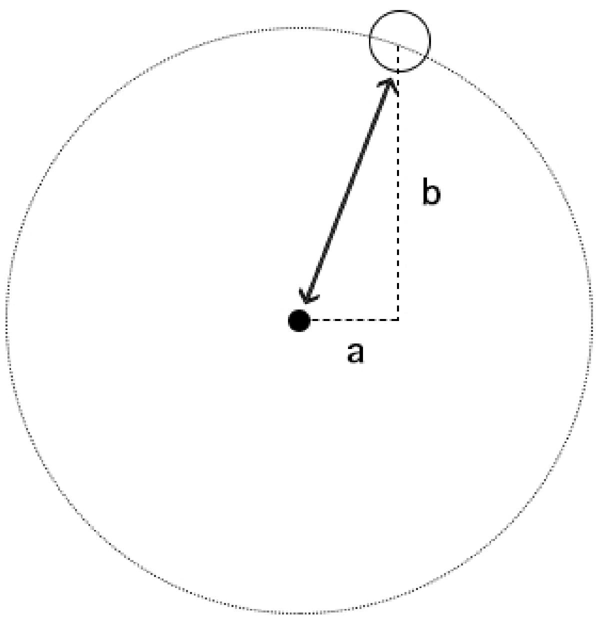


Soit AB de valeur 1. Nous pouvons alors définir les coordonnées du point B à l'aide du système de coordonnées polaire, à savoir :

X = cos(t)*radius;
Y = sin(t)*radius;

Enfin retenons que la fonction tangeante d'un angle est le rapport exercé par le sinus sur son cosinus. Soit **tan(t) = sin(t)/cos(t);**

//Attracteur et repulseur simple



Dans le cadre de processing les fonctions trigonométrique peuvent donc nous permettre à la fois de calculer des angles et des distances.

Imaginons un point au coordonnées x,y que nous souhaitons attirer par un autre point au coordonnée tx, ty.

Afin de créer notre attracteur répulseur d'une manière simple, il suffira de passer notre point de coordonnées cartésienne à polaire de centre tx, ty. Ainsi en réduisant/augmentant son rayon nous verrons varier sa distance avec l'attracteur.

Cette méthode permet de simplifier les calculs (notamment dans la cadre d'une série de point) étant donné que seul le rayon change.

Afin de changer les coordonnées d'un point d'un espace cartésien à polaire il nous suffit de définir ces x et y de sorte que :

> **X = origineX+cos(angle)*rayon;**
> **Y = origineY+in(angle)*rayon;**

Or dans notre cas nous connaissons les valeurs suivante :

> X et y à passer en coordonnées polaire
> origineX et origineY, les coordonées de notre attracteur, centre de notre cercle
> Le rayon de notre cercle, cad la distance entre nos deux points, soit : dist(x, origineX, y, origineY)

Nous devon sdonc définir la valeur manquante, à savoir l'angle ACB.
Or nous savons que **tan(ACB) = a/b;**

Et si **tan(ACB) = a/b;** les fonction trigonométrique nous apprennent que l'angle ACB est égales à l'inverse de la tangente ACB.

> Soit **angle = atan(a/b);**

Dans processing cela nous donnera donc :

> **Angle = atan2(y-originey, x, originex);**

Par ailleurs il existe diverse méthodes inverse à savoir

> **acos();**
> **asin();**
> **atan();**