Coniques

Projet de programmation objet - mathématiques pour l'informatique
imac2

Résumé

Ce projet se place à l'intersection de la programmation C++ et des mathématiques numériques. Il s'agit d'explorer les méthodes de construction d'une conique.

1 Introduction

Ce projet consiste à étudier les coniques, leur construction et leur représentation en géométrie projective.

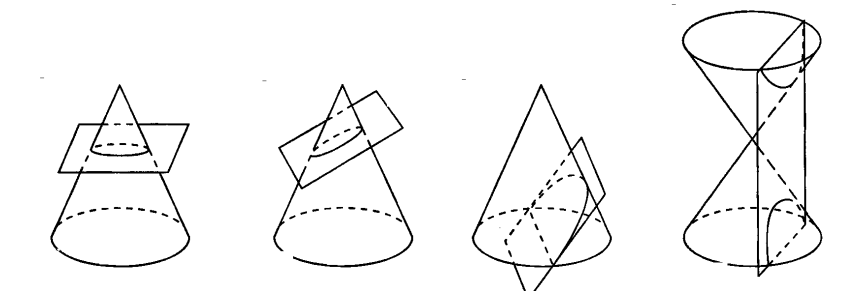
Ce projet sera à réaliser :

- par binôme
- soutenances : jeudi 23 novembre matin (+ merc 22 si trop de groupes)
- pas de rapport
- envoyez moi votre lien git dès que vous commencez.
- ce projet sera évalué avec deux notes : une pour la partie programmation et une notre pour la partie mathématiques.
- la note comptera comme des points en plus dans l'examen des 2 matières

2 Section de conique

Une conique (ou section de conique) correspond à l'intersection entre un plan et un cône infini. Algébriquement, une conique est représentée par 6 valeurs $a, b, c, d, e, f \in \mathbb{R}$ de telle façon qu'un point $\mathbf{x} = (x, y)^\top$ appartient à la conique ssi :

$$\mathrm{a}x^2 + \mathrm{b}xy + \mathrm{c}y^2 + \mathrm{d}x + \mathrm{e}y + \mathrm{f} = 0 \tag{1}$$



cercle	ellipse	parabole	hyperbole
$a = c$ et $b = 0$	$b^2 - 4ac < 0$	$b^2 - 4ac = 0$	$b^2 - 4ac > 0$

2.1 Conique et points de contrôle

Une conique peut être définie par un ensemble de 5 points de contrôle $\{\mathbf{x}_i\}_{i \in 1 \ldots 5}$. Chacun de ces points doit satisfaire l'équation (1), celle-ci peut prendre la forme d'un produit scalaire :

$$\begin{pmatrix} x^2 & xy & y^2 & xw & yw & w^2 \end{pmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = 0$$

qui se généralise sur plusieurs points par le système linéaire suivant :

$$\begin{bmatrix} x_1^2 & x_1 y_1 & y_1^2 & x_1 w_1 & y_1 w_1 & w_1^2 \\ x_2^2 & x_2 y_2 & y_2^2 & x_2 w_2 & y_2 w_2 & w_2^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_n^2 & x_n y_n & y_n^2 & x_n w_n & y_n w_n & w_n^2 \end{bmatrix} \begin{pmatrix} a \\ b \\ c \\ d \\ e \\ f \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Soit A la matrice relative à ce système, pour le résoudre, il faut calculer le noyau de A (*rigth null space*). En pratique, on utilise une décomposition en valeurs singulière, dont voici le code en utilisant la bilbiothèque Eigen :

```
Eigen::JacobiSVD<Eigen::MatrixXd> svd(A,
                  Eigen::ComputeThinU|Eigen::ComputeFullV);
Eigen::VectorXd x = svd.matrixV().rightCols(1);
```

Question : Pourquoi peut-on résoudre ce système avec uniquement 5 points ?

2.2 Géométrie projective

2.3 Droites et points

La géométrie projective rajoute une coordonnée de plus par rapport à la géométrie euclidienne. Alors qu'un point dans $\mathbb{R}^2$ n'a que 2 dimensions, ce même point en aura 3 dans $\mathbb{P}^2$ (coordonnées homogènes). Dans $\mathbb{P}^2$, un point se note $\mathbf{x} = (x, y, w)^\top$.

Pour passer des coordonnées euclidiennes $\mathbf{x}_{\mathbb{R}^2} = (x', y')^\top$ aux coordonnées homogènes $\mathbf{x}_{\mathbb{P}^2} = (x, y, w)^\top$, il suffit d'ajouter une composante $w = 1$ à $\mathbf{x}_{\mathbb{R}^2}$ qui devient $\mathbf{x}_{\mathbb{P}^2} = (x', y', 1)^\top$. Pour passer des coordonnées homogènes $\mathbf{x}_{\mathbb{P}^2} = (x, y, w)^\top$ aux coordonnées euclidiennes $\mathbf{x}_{\mathbb{R}^2} = (x', y')^\top$, il suffit de diviser chaque composantes de $\mathbf{x}_{\mathbb{P}^2}$ par w (si $w \neq 0$). On obtient alors $\mathbf{x}_{\mathbb{R}^2} = (x', y')^\top = (\frac{x}{w}, \frac{y}{w})^\top$. Si $w = 0$, alors $x' = x$ et $y' = y$ mais $\mathbf{x}_{\mathbb{R}^2}$ représente alors un point à l'infini, qui correspond à une direction plutôt qu'à un point.

Voici quelques propriétés intéressantes des points dans $\mathbb{P}^2$:

- Les points $\mathbf{x}$ et $k\mathbf{x}$ ($k \in \mathbb{R}^*$) représentent le même point.

$$\begin{pmatrix} x \\ y \\ w \end{pmatrix} \doteq \begin{pmatrix} x/w \\ y/w \\ 1 \end{pmatrix} \doteq \begin{pmatrix} kx \\ ky \\ kw \end{pmatrix}$$

  où x/w et y/w représentent les coordonnées cartésiennes du point pour $w \neq 0$ et '$\doteq$' correspond à une égalité à un facteur d'échelle près.
- Une droite de $\mathbb{P}^2$ se note $\mathbf{l} = (a, b, c)^\top$.
- Les droites $\mathbf{l}$ et $k\mathbf{l}$ ($k \neq 0$) représentent la même droite.
- Un point $\mathbf{x} = (x, y, w)^\top \in \mathbf{l}$ ssi $\mathbf{x}^\top\mathbf{l} = \mathbf{l}^\top\mathbf{x} = 0$.
- La droite $\mathbf{l}$ passant par les points $\mathbf{x}_1$ et $\mathbf{x}_2$ vaut : $\mathbf{l} \doteq \mathbf{x}_1 \times \mathbf{x}_2$ (où $\times$ correspond au produit vectoriel de $\mathbb{R}^3$).

- L'intersection $\mathbf{x}$ de 2 droites $\mathbf{l}_1$ et $\mathbf{l}_2$ se calcule avec : $\mathbf{x} \doteq \mathbf{l}_1 \times \mathbf{l}_2$. Notez le caractère dual entre les points et les droites dans $\mathbb{P}^2$.
- Le point $(x, y, 0)^\top$ correspond à un point à l'infini aussi appelé point idéal.
- Deux droites parallèles s'intersectent en un point à l'infini.

2.4 Coniques

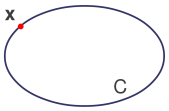
En géométrie projective, l'équation d'une conique prend la forme suivante :

$$\mathrm{a}x^2 + \mathrm{b}xy + \mathrm{c}y^2 + \mathrm{d}xw + \mathrm{e}yw + \mathrm{f}w^2 = 0 \tag{2}$$

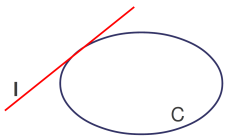
Une conique peut d'ailleurs se représenter matriciellement par :

$$\mathrm{C} = \begin{bmatrix} a & b/2 & d/2 \\ b/2 & c & e/2 \\ d/2 & e/2 & f \end{bmatrix} \tag{3}$$

Sous cette forme, un point $\mathbf{x} = (x, y, w)^\top$ appartient à une conique C ssi : $\mathbf{x}^\top \mathrm{C} \mathbf{x} = 0$



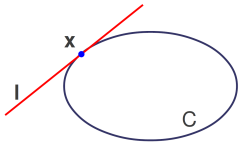
Par ailleurs, une droite $\mathbf{l}$ est tangente à la conique C ssi : $\mathbf{l}^\top \mathrm{C}^{-1} \mathbf{l} = 0$.



$$\mathbf{l}^\top \mathrm{C}^{-1} \mathbf{l} = 0$$

Question : Montrez pourquoi la tangente $\mathbf{l}$ d'une conique C passant par le point $\mathbf{x} \in \mathrm{C}$ s'exprime sous la forme :

$$\mathbf{l} = \mathrm{C}\mathbf{x}$$



3 A réaliser

Pour commencer, vous répondrez aux quelques questions déjà rencontrées dans la partie explicative.

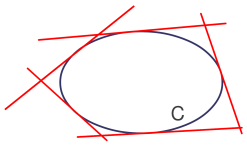
Ensuite, vous devrez réaliser un programme qui calcule la conique passant par 5 points. Ces points pourront être choisis de façon aléatoire ou bien lus sur un fichier. Que se passe-t-il si vous incluez un point à l'infini ?

Vous devrez également implémenter une extension de votre méthode au cas surdéterminé, c'est à dire au cas où la conique doit passer par plus de 5 points, éventuellement placés de façon à ne correspondre à aucune conique. Il faudra alors trouver la conique qui passe au mieux par ces points au sens des moindres carrés. A noter, dans ce problème vous travaillerez sur l'erreur algébrique et non sur l'erreur géométrique (distance minimale entre le point et la conique). D'ailleurs, quelle est la différence entre ces deux distance, la distance algébrique étant la valeur $d = \mathrm{a}x^2 + \mathrm{b}xy + \mathrm{c}y^2 + \mathrm{d}xw + \mathrm{e}yw + \mathrm{f}w^2$?

Enfin, vous pourrez explorer les propriétés des points et des droites en rapport avec les coniques.

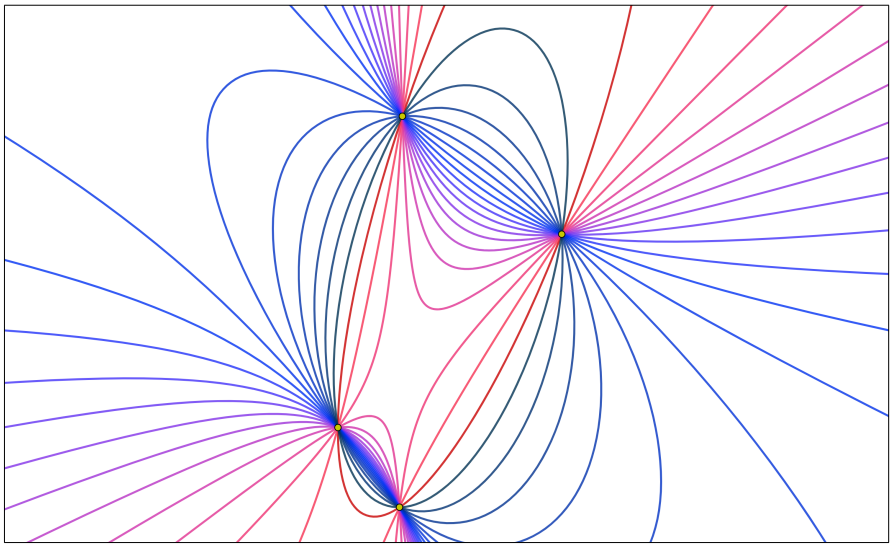
3.1 Mathématiques avancés

1. La section 2.1 décrit comment construire une conique à partir de 5 points de contrôle. Comment peut-on construire une conique à partir de 5 tangentes ?



2. Un faisceau de coniques correspond à une combinaison linéaire de 2 coniques. Étant donnés 2 coniques C_a et C_b, nous pouvons ainsi définir le faisceau $\mathrm{C}(\alpha, \beta) = \alpha\mathrm{C}_a + \beta\mathrm{C}_b$ avec $\alpha, \beta \in \mathbb{R}$. En pratique, il est plus simple de calculer un faisceau avec la formule :

$$\mathrm{C}(t) = \cos(t)\mathrm{C}_a + \sin(t)\mathrm{C}_b \quad \text{avec} \quad t \in [0, \pi]$$



Pour calculer un faisceau, pensez à normer vos deux coniques C_a et C_b en divisant chacune de leur composante par leur norme respective :

$$\|C\| = \sqrt{a^2 + b^2 + c^2 + d^2 + e^2 + f^2}$$

Affichez quelques coniques du faisceau $\mathrm{C}(t)$. Vérifiez que ces coniques passent toutes par les points d'intersection entre C_a et C_b.

3. Dans certains cas, une conique peut être une paire de droites. Quelle est l'équation de cette conique ? Comment la construire ?
4. Il se trouve qu'une conique peut même être un point (un cercle de rayon nul). Quelle est l'équation de cette conique ?

4 Partie programmation

4.1 Les coniques

Naturellement, il vous est demandé de coder une implémentation des coniques en `C++`. Vous êtes assez libre sur la forme que prendra votre code.

4.2 Les exemples

Vous fournirez une batterie d'exemples destinés à montrer que votre code fonctionne, ainsi que pour aider un utilisateur à commencer à utiliser vos coniques.

4.3 Compilation

Il est largement conseillé d'utiliser une version moderne du C++ (11, 14, 17 ou 20). La compilation de votre projet devra être gérée avec **cmake** (au moins la version 3.13). Tester votre projet avec plusieurs compilateurs (clang / gcc / etc.) est considéré comme une bonne pratique (à mentionner dans votre rapport).

4.4 Lisibilité

J'attends un code lisible et bien organisé. Bien organisé signifie que vous répartirez judicieusement votre code sur plusieurs fichiers, et que les fonctions ou classes composants chaque fichier seront elles aussi bien réparties. Par ailleurs, il est nécessaire de commenter votre code en gardant à l'esprit qu'il doit être compréhensible même sans ces commentaires : essayez de nommer vos variables et vos fonctions explicitement et n'hésitez pas découper de longs algorithmes en plus petites fonctions.

4.5 Gestionnaire de versions

Vous devrez utiliser le gestionnaire de version Git et héberger votre dépôt sur une plateforme en ligne, telle que GitHub, GitLab ou Bitbucket. Pensez à faire des `commit` très régulièrement, ça sera pour moi une indication permettant de vérifier que vous n'avez pas tout fait au dernier moment.

4.6 Côté technique

Voici ce qu'il serait bon de rencontrer dans votre projet :

- polymorphisme
- usage d'outils de la STL
- exceptions pour traiter les erreurs
- `asserts` pour détecter et prévenir les erreurs de programmation
- espaces de nommage (*namespace*)
- fonctions lambdas `constexpr`
- (bonus) fonctions `variadics`

Voici ce qu'il serait regrettable de rencontrer dans votre projet :

- une mauvaise organisation de vos fichiers et de vos classes

— des variables, des fonctions ou des types mal nommés
— des références non constantes alors qu'elles devraient l'être
— des passages par copies non justifiés
— des erreurs de segmentation
— des fuites de mémoires
— des bugs
— du code de debug non retiré
— du code mort (= des portions de code non utilisés par le programme)
— du code dupliqué
— du code illisible d'un point de vue sémantique (des `for` dans des `for` dans des `if` dans des `for` dans des `else` par exemple)
— du code illisible d'un point de vue esthétique (indentation irrégulière, mélange camelCase / snake_case, mélange tabs / spaces)
— des mega fonctions de plus de 30 lignes
— des mega fichiers de plus de 500 lignes
— des variables globales non `constexpr`
— des crashs lorsque l'utilisateur effectue une opération non prévue

4.7 Pour finir

N'hésitez pas à vous approprier le sujet : tout ajout non demandé sera le bienvenu. Bon courage !!