Projet de mathématiques

Mathématiques pour l'informatique

—IMAC 2—

Trifocal Tensor

Ce projet est une introduction aux problèmes de vision par ordinateur et de géométrie projective. Il inclut d'une part l'utilisation d'outils mathématiques tels que la résolution de systèmes linéaires surdéterminés au sens des moindres carrés et d'autre part de la programation en C++.

Ce projet est à réaliser par groupe de 2 en C++.

---

1 Introduction

Extraire de l'information 3d à partir d'une photographie peut paraître trivial à n'importe quelle personne alors qu'un ordinateur peine à en extraire le moindre indice de géométrie 3d. En effet, alors même qu'un enfant identifie l'organisation géométrique de la scène photographiée et reconstruit mentalement les objets qui la composent, l'ordinateur n'y "voit" que des pixels.

Il existe toutefois des méthodes permettant d'établir des relations 3d entre plusieurs images. Si l'on dispose de 2 images, on peut utiliser la géométrie épipôlaire qui nous dit que le correspondant d'un point sur une image appartient à une certaine droite sur la seconde image. Si l'on dispose de 3 images, on peut utiliser un tenseur trifocal (*trifocal tensor* en anglais). Il s'agit en quelque sorte d'une extension de la géométrie épipôlaire pour 3 vues. Si l'on connait la position (en pixel) d'un objet sur 2 images, on le trouve automatiquement sur la troisième. Pour établir une relation entre les images, la géométrie épipôlaire et le tenseur trifocal utilisent tous les deux des appariements de pixels images à images, ce qui consiste à dire : "ce point sur cette image se retrouve ici sur les autres images". Quand on a corrélé un ensemble de pixels représentant les mêmes objets sur plusieurs images, on peut établir notre relation entre les images puis l'utiliser.

Ce sujet traite uniquement de la partie tenseur trifocal entre 3 images. Il aborde les points suivants :

- Gestion des images et listes de points (entrées / sorties).
- Inversion de matrices.
- Matrices pseudo-inverses.
- Noyau d'une matrice
- Création d'un tenseur.
- Transfert de pixel (utilisation du tenseur).

2 Notions mathématiques

Ce chapitre regroupe les notions nécessaires au calcul tensoriel en géométrie projective ainsi qu'en algèbre linéaire.

2.1 Géométrie projective

La géométrie projective fait intervenir une coordonnée de plus par rapport à la géométrie euclidienne, on parle de "coordonnées homogènes". Alors qu'un point dans $\mathbb{R}^2$ n'a que 2 dimensions, ce même point en aura 3 dans $\mathbb{P}^2$. 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 une direction plutôt qu'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, autrement dit une équivalence.
- Une matrice (ou un tenseur) transformant un point de $\mathbb{P}^2$ est elle aussi invariante par facteur d'échelle.
- Le point $(x, y, 0)^\top$ correspond à un point à l'infini se trouvant dans la direction $(x, y)^\top$ exprimée dans le référentiel de l'image.

2.2 Systèmes surdéterminés

Un système surdéterminé est un système où il y a plus d'équations que d'inconnues. Les équations supplémentaires ne sont pas nécessairement en accord avec les précédentes (elles ne sont pas forcément combinaisons linéaires des premières). Il est possible de trouver une solution au sens des moindres carrés. Evidemment, la solution trouvée ne pourra en général pas satisfaire toutes les équations mais tentera de les satisafire toutes "au mieux".

Pour résoudre un système surdéterminé $A\mathbf{x} = \mathbf{b}$, on peut utiliser la matrice pseudo-inverse de A notée A^+ avec $A^+ = (A^\top A)^{-1}A^\top$. Le résultat du système est alors $\mathbf{x} = A^+\mathbf{b}$.

2.3 Noyau d'une matrice

Par la suite, nous serons amener à résoudre des systèmes linéaires du genre $A\mathbf{x} = \mathbf{0}$ et nous cherchons évidemment une autre solution que la solution triviale $\mathbf{x} = \mathbf{0}$. Pour cela, nous pouvons utiliser la décomposition en valeurs singulière d'une matrice (*SVD* en anglais) qui calcule le vecteur solution $\mathbf{x}$ tel que $\|\mathbf{x}\| = 1$ (ce qui implique $\mathbf{x} \neq \mathbf{0}$) et qui minimize $\|A\mathbf{x}\|$ (ce qui implique que $\mathbf{x}$ est bien la solution que nous cherchons).

En pratique, on utilise souvent un programme applicant la *SVD* à la matrice A qui se décompose alors en $A = UDV^\top$. Il est important de vérifier que les valeurs singulières de A représentées par les éléments de la matrice diagonale D sont bien triées par valeurs décroissante.

La solution du système est alors $\mathbf{x} = \mathbf{v}$ où $\mathbf{v}$ représente la dernière colonne de la matrice $V^\top$, c'est-à-dire la colonne associée à la plus petite valeur singulière de A.

3 Tenseur trifocal

Cette partie traite de la création et de l'utilisation du tenseur trifocal. Elle introduit d'abord une approche faisant appel à votre intuition, pour vous faire comprendre l'utilisation que l'on veut faire des tenseurs. Elle présente ensuite une approche plus mathématique sur les moyens d'y parvenir.

3.1 Les différentes étapes

Un tenseur trifocal est utilisé pour établir une relation géométrique entre 3 images d'une même scène prises de 3 points de vu différents. La première étape de notre système consiste à charger ces 3 images (figure 1). On peut noter que les 3 images représentent la même scène prise sous des points de vues différents. On remarque aussi que cette scène contient de la profondeur, notamment au niveau du socle. Enfin, il s'agit d'une scène statique, ce qui permet de n'utiliser qu'un seul appareil photo tout en étant certain qu'il s'agisse de la même scène. Pour une scène dynamique, il faut utiliser 3 appareils photos synchronisés ou 3 caméras.

FIGURE 1 – Les 3 images de départ.

Une fois chargées, il faut sélectionner quelques points de correspondances entre les images (au moins 7). Il est préférable de sélectionner des points caractéristiques de l'image comme des coins ou des points facilement repérables (figure 2). Ces points de correspondance sont nécessaires pour calculer notre tenseur. Celui-ci nous permet ensuite de tirer profit de la relation géométrique établie entre les images.

FIGURE 2 – Sélection de points de correspondances sur les 3 images.

En sélectionnant un même objet sur deux images, le tenseur permet de calculer la position 2d du pixel correspondant au même objet sur la troisième image (figure 3). Cette opération s'appelle "le transfert".

FIGURE 3 – Un objet sélectionné sur deux images se retrouve automatiquement sur la troisième.

3.2 Interprétation géométrique

Pour établir ce genre de relation géométrique entre 3 images, le plus simple est de connaitre la position et les paramètres de chaque caméra. Pour transférer deux pixels sélectionnés sur les 2 premières images, il suffit pour chaque caméra de calculer la droite passant par le centre de projection de la caméra et par le pixel sélectionné. L'intersection 3d de ces deux droites nous donne la position 3d de l'objet sélectionné sur les deux images. Cette méthode s'appelle la triangulation. Il suffit alors de projeter ce point 3d sur la troisième image en utilisant le centre de projection de la troisième caméra, et le tour est joué.

Malheureusement, dans notre cas, nous ne disposons ni de la position des caméras, ni de leurs paramètres. Il est toutefois possible de modéliser notre système de trois caméras en utilisant des outils de vision par ordinateur. Cependant ce modèle est dit perspectif, par opposition à métrique. Cela signifie qu'en triangularisant des points de la scène avec cette méthode, le modèle 3d reconstruit serait complètement "distordu". Pour le rétablir, il faudrait connaitre les paramètres des caméras qu'il nous manque justement. D'un autre côté, l'avantage de ce modèle est de pouvoir utiliser cet espace perspectif pour trianguler un point 3d (distordu) et de le reprojeter sur la troisième image, et le tout de façon cohérente (figure 4). Donc même si l'espace de reconstruction est distordu, le transfert de point fonctionne parfaitement.

Notons que les deux droites utilisées pour la triangulation ne se croisent en général pas parfaitement à cause de l'imprécision de la sélection des points, des imperfections de optiques des caméras ou tout simplement à cause de la discrétisation en pixels que nous utilisons sur nos caméras. Lors d'une triangulation, l'intersection 3d de ces deux droites est trouvée avec la méthode des moindres carrés. Il en est de même (implicitement) avec les tenseurs lors de l'opération de transfert.

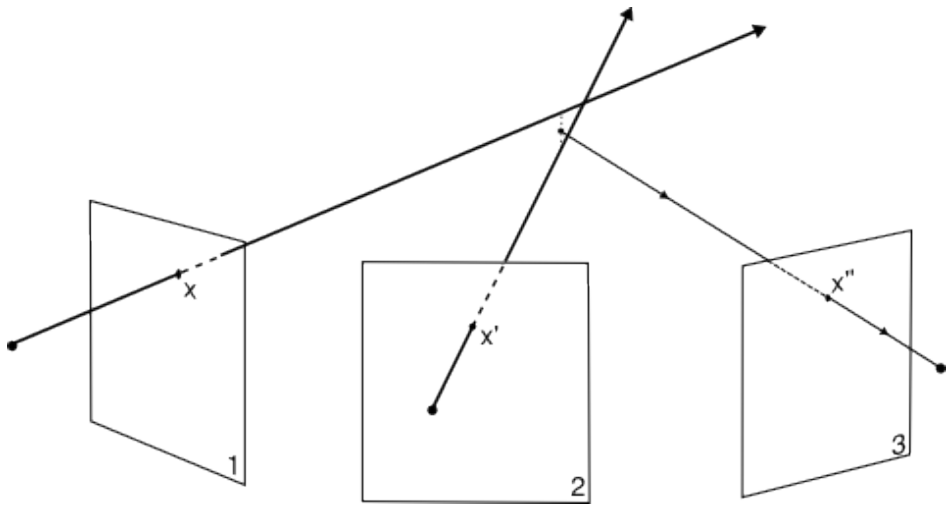

FIGURE 4 – Triangulation de **x** et **x'** dans un espace perspectif puis reprojection de l'intersection (**x"**).

3.3 Notation tensorielle

Comme très souvent en vision par ordinateur, les méthodes mises en œuvre font souvent appel à l'algèbre linéaire. Pour mettre en relation deux points, on utilise des matrices M telles que :

$$\mathbf{x}' = \mathbf{M}\mathbf{x}$$

Dans notre cas, la mise en relation de trois points s'effectue à l'aide de tenseurs :

$$x^i x'^j x''^k \epsilon_{jqs} \epsilon_{krt} T_i^{qr} = 0_{st}$$

Un tenseur trifocal n'est autre qu'une matrice $3 \times 3 \times 3$. Soit T un tenseur, T_k^{ij} représente l'élément i, j, k de T. Pour un point $\mathbf{x}$, x^i représente sa $i^{ième}$ composante.

Dans l'équation précédente, l'opérateur ϵ_{rst} est défini pour $r, s, t = 1, ..., 3$ de la façon suivante :

$$\epsilon_{rst} = \begin{cases} 0 \text{ sauf si } r, s \text{ et } t \text{ sont différents} \\ +1 \text{ si } rst \text{ est une permutation impaire de 123} \\ -1 \text{ si } rst \text{ est une permutation paire de 123} \end{cases}$$

L'utilisation des tenseurs implique une certaine notation. Il s'agit d'une convention de répétition consistant à utiliser le fait que les indices vont devenir eux-même l'indication de la répétition. Par exemple, l'équation $x^i y^i$ représente en fait $\sum_{i=1}^{n} x^i y^i$ où n est la taille des vecteurs x et y.

De même, on a $A_{ii} x^j = A_{11} x^j + A_{22} x^j + \cdots + A_{nn} x^j$. L'indice j n'étant pas répété dans l'équation de départ, il s'agit d'un indice sur lequel il n'y a pas de répétition à effectuer.

L'équation $a_{ij} x^j = b^i$ avec $n = 3$ représente le système d'équation :

$$\begin{cases} a_{11}x^1 + a_{12}x^2 + a_{13}x^3 = b^1 \\ a_{21}x^1 + a_{22}x^2 + a_{23}x^3 = b^2 \\ a_{31}x^1 + a_{32}x^2 + a_{33}x^3 = b^3 \end{cases}$$

En résumé, toute expression qui comporte plusieurs occurrences d'un indice représente une répétition sur toutes les valeurs possibles de l'indice en question.

3.4 Le tenseur trifocal

La relation entre un tenseur trifocal T et trois points $\mathbf{x}$, $\mathbf{x}'$ et $\mathbf{x}''$ représentant le même objet 3d sur chaque image se traduit par l'équation $x^i x'^j x''^k \epsilon_{jqs} \epsilon_{krt} T_i^{qr} = 0_{st}$. Après un développement du ϵ_{jqs} et du ϵ_{krt} , on obtient l'équation simplifiée suivante :

$$x^k (x'^i x''^m T_k^{jl} - x'^j x''^m T_k^{il} - x'^i x''^l T_k^{jm} + x'^j x''^l T_k^{im}) = 0^{ijlm}$$

La notation 0^{ijlm} renseigne sur quelles variables il faut faire une répétition d'équation. Nous remarquons que k n'en fait pas parti ce qui signifie que la boucle des k est doit s'effectuer sur une équation alors que les autres variables génèrent de nouvelles équations.

En développant encore d'avantage, on peut remarquer que pour $i = j$ ou $l = m$, on obtient les mêmes équations. On peut aussi constater qu'une permutation entre i et j (ou entre l et m) ne fait que changer le signe des équations. On peut donc fixer arbitrairement $j = m = 3$ et faire varier i et l de 1 à 2. La variable k varie de 1 à 3 mais ne génère pas de nouvelle équation. L'équation de départ devient alors :

$$x^k(x'^i x''^3 T_k^{3l} - x'^3 x''^3 T_k^{il} - x'^i x''^l T_k^{33} + x'^3 x''^l T_k^{i3}) = 0^{il}$$

3.5 Calcul d'un tenseur trifocal

Un tenseur trifocal étant une matrice $3 \times 3 \times 3$, il possède 27 éléments. Il faut donc 27 équations pour le calculer. Ces équations dérivent de la formule du paragraphe précédent. Pour obtenir suffisamment d'équation, il faut donc disposer d'au moins 7 correspondances $\mathbf{x_p} \leftrightarrow \mathbf{x'_p} \leftrightarrow \mathbf{x''_p}$ ($p = 1, ..., n$, $n \leq 7$) qui permettent d'établir au moins 28 équations de la forme :

$$\sum_{k=1}^{3} x_p^k(x_p'^i x_p''^3 T_k^{3l} - x_p'^3 x_p''^3 T_k^{il} - x_p'^i x_p''^l T_k^{33} + x_p'^3 x_p''^l T_k^{i3}) = 0^{il}$$

i et l varient de 1 à 2 et p varie de 1 à n (n = le nombre de correspondances) ce qui génère 4 équations par correspondance soit au total $4n$ équations. Le problème suivant consiste à définir comment construire un système linéaire permettant de calculer T. En fait, il suffit simplement d'écrire T sous la forme d'un vecteur $\mathbf{t}$ à 27 éléments et de résoudre le système $A\mathbf{t} = \mathbf{0}$. Dans ce système, la matrice A est une matrice $4n \times 27$, et le vecteur solution est le vecteur nul de dimension $4n$. Il est possible de résoudre ce système avec les outils décrits dans la partie 2.3.

La difficulté est donc de construire la matrice A. Une fois $\mathbf{t}$ calculé, il suffit de réarranger ses composantes sous forme de tenseur.

3.6 Le transfert

Le transfert se calcule à partir de la même équation que celle utilisée pour le calcul de T, mais dans ce cas là, les inconnues sont les trois composantes du point transféré et non les 27 composantes de $\mathbf{t}$.

$$x^k(x'^i x''^3 T_k^{3l} - x'^3 x''^3 T_k^{il} - x'^i x''^l T_k^{33} + x'^3 x''^l T_k^{i3}) = 0^{il}$$

Dans cette équation, nous disposons maintenant de T ainsi que de deux points sur trois. Traitons le cas où nous cherchons $\mathbf{x''}$, nous devons là encore résoudre ce problème avec une équation de la forme $A\mathbf{x''} = \mathbf{0}$. En faisant varier i et l de 1 à 2 et k de 1 à 3, on obtient quatre équations à deux inconnues. Il faut donc là encore utiliser les outils décrits dans la partie 2.3 pour résoudre le système.

4 Travail demandé

L'objectif de ce projet est d'implémenter un programme de gestion de tenseur trifocal. Ce programme devra satisfaire les conditions suivantes :

- être écrit en `C++`, utiliser un `makefile` et ne plus afficher de warning lors de la compilation.
- fonctionner au moins sous Linux (éventuellement portable sous Windows et/ou Mac).
- lancement sous console avec les option suivantes :
  - `-h` pour afficher l'aide en anglais (inspirez vous du man).
  - [image1].jpg [image2].jpg [image3].jpg : qui correspondent au nom des trois images à charger.
  - [image1].jpg [image2].jpg [image3].jpg [list1].list [list2].list [list3].list : les 3 noms de fichiers images peuvent être suivis des 3 noms de listes de points de correspondances respectives.
  - d'autres options que vous jugerez intéressantes.

  Toutes les options devront être gérées sur la ligne de commande.
- ne pas utiliser d'autres interfaces que celles de la SDL (pas de `cin >> ...`).
- pouvoir cliquer des correspondances et les sauvegarder dans un fichier, ou pouvoir charger une liste déjà établie. Le format de sauvegarde défini dans la suite de ce document devra être parfaitement respecté.
- Pouvoir calculer un tenseur.
- Pouvoir effectuer le transfert de points sur le tenseur calculé.
- Vous devrez rendre votre projet sous forme d'archive `nom1_nom2.tgz` compressant un répertoire du même nom contenant votre programme ainsi qu'un exemple prêt à l'emploi et votre rapport.

Le travail à effectuer sera réparti de la façon suivante :

- la gestion des images, des listes de points (pour les correspondances) et l'affichage SDL.
- le calcul d'un tenseur
- le transfert de points.
- un rapport d'une dizaine de pages au format pdf.

4.1 Partie informatique et mathématique

La gestion des images et l'affichage s'effectuera avec la SDL et la SDL-image.
Les outils mathématiques (matrices, vecteurs, etc.) seront gérés par la bibliothèque mathématique eigen : http ://eigen.tuxfamily.org

Un tenseur poura être représenté par une `class Tensor` surchargeant l'oppérateur (i, j, k). Soit T un tenseur, T_k^{ij} représente l'élément i, j de la $k^{ième}$ matrice.

Pour chaque image, il est nécessaire de pouvoir sauvegarder ou charger une liste de pixels afin de faciliter l'utilisation de votre programme. Les listes de n vecteurs doivent être sous la forme d'une matrice $n \times 3$:

```
row n (replace n by the number of rows)
col 3

112.0 41.0 1.0
84.0 221.0 1.0
...
```

Les points seront ordonnés dans l'ordre de leur sélection pour s'assurer de la cohérence entre chaque liste. Ils seront stockés sous forme de matrice dans votre programme.

4.2 Tenseur

Le calcul du tenseur se fera automatiquement si les points sont chargés. Ce calcul sera déclenché manuellement si tous les points sont sélectionnés manuellement ou si l'on complète des listes de points déjà chargées en ajoutant des correspondances. Votre programme devra vérifier la validité des listes avant de calculer le tenseur. Celui-ci devra fonctionner quelque soit le nombre de correspondance (à partir de 7).

Le transfert devra fonctionner quelque soit les deux images cliquées.

4.3 Rapport

Vous fournirez en version papier et en version électronique un rapport de 10 pages maximum. **Consacrez suffisament de temps au rapport car il représente une bonne partie de la note finale**. Essayez de respecter au mieux les directives suivantes :

- **Votre rapport doit commencer par une page listant vos travaux avec les indications suivantes** :
  - éléments demandés et codés qui fonctionnent.
  - éléments demandés et codés qui ne fonctionnent pas.
  - éléments demandés mais pas codés.
  - éléments non demandés et codés qui fonctionnent.
  - éléments non demandés mais pas codés ou qui ne fonctionnent pas.
- Ne perdez pas de temps à réexpliquer le sujet du projet, l'enseignant le connaît déjà, faites seulement un bref résumé de quelques lignes. De manière plus générale, ne détaillez pas des méthodes déjà expliquées dans l'énoncé à moins que vous les ayez modifiées.
- Un rapport sert surtout à montrer comment vous avez fait face aux problèmes (d'ordre algorithmique). Certains problèmes sont connus (on en parle dans l'énoncé), d'autres sont imprévus. Montrez que vous les avez remarqués et compris. Donnez la liste des solutions à ce problème et indiquez votre choix. Justifiez votre choix (vous avez le droit de dire que c'est la méthode la plus facile à coder).
- Il ne doit figurer aucune ligne de code dans votre rapport. Un rapport n'est pas un listing de votre programme où vous détaillez chaque fonction. Vous devez par contre détailler vos structures de données et mettre du pseudocode pour expliquer vos choix

algorithmiques.
Il est autorisé d'utiliser des "raccourcis" tels que "`initialiser T à 0`" plutôt que de détailler la boucle faisant la même chose.

– n'hésitez pas à mettre des images dans votre rapport pour illustrer vos propos et vos résultats.

5 Pour finir

Vous pouvez laisser libre cours à votre imagination, toute amélioration sera la bienvenue. Vous trouverez quelques informations complémentaires à l'adresse :

`http://www-igm.univ-mlv.fr/~vnozick/`

Bon courage.