TP TIVO : Intégration 3D dans une vidéo

Contents

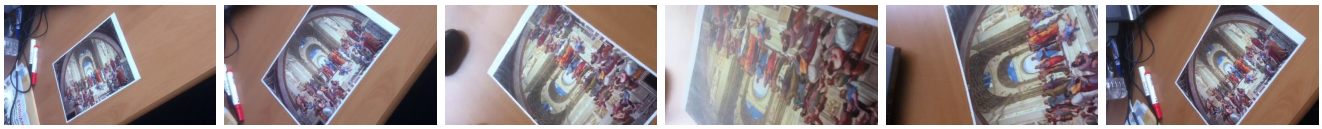

Figure 1: Extraits de la séquence vidéo.

1 Introduction

L'objectif de ce TP est d'intégrer un cube en 3D dans toutes les images d'une séquence vidéo, en utilisant :

- une méthode basée sur la mesure de deux points de fuite orthogonaux, pour obtenir la focale et le point de vue de la caméra dans la première image de la séquence,
- diverses méthodes pour la détection et la description de points d'intérêt dans les images vidéo,
- la méthode RANSAC pour apparier les points d'intérêt entre les images de la séquence et permettre d'actualiser le point de vue dans chaque image.

Ce TP est une implémentation directe de l'article suivant:

Markerless Tracking using Planar Structures in the Scene. Gilles Simon, Andrew W. Fitzgibbon, Andrew Zisserman. *Proc. International Symposium on Augmented Reality*, Oct 2000.

L'article est disponible à l'adresse https://www.robots.ox.ac.uk/~vgg/publications/2000/Simon00/simon00.pdf. Le TP est à réaliser sous Matlab, du code à trous vous est donné.

2 Calcul de la focale

La vidéo consiste en un déplacement de caméra autour d'une feuille A4 sur laquelle l'image d'un tableau a été imprimée (L'École d'Athène de Raphaël, cf. Fig. 1). Cette feuille étant rectangulaire et texturée, on peut l'utiliser à la fois pour le calcul du point de vue à partir de points de fuite et pour le tracking de caméra basé sur l'appariement de points d'intérêt.

Nous commençons par calibrer la caméra, à partir de la première image de la séquence vidéo. Le code déjà écrit dans le fichier `run.m` permet d'ouvrir le répertoire dans lequel se trouve la séquence vidéo, de charger et d'afficher

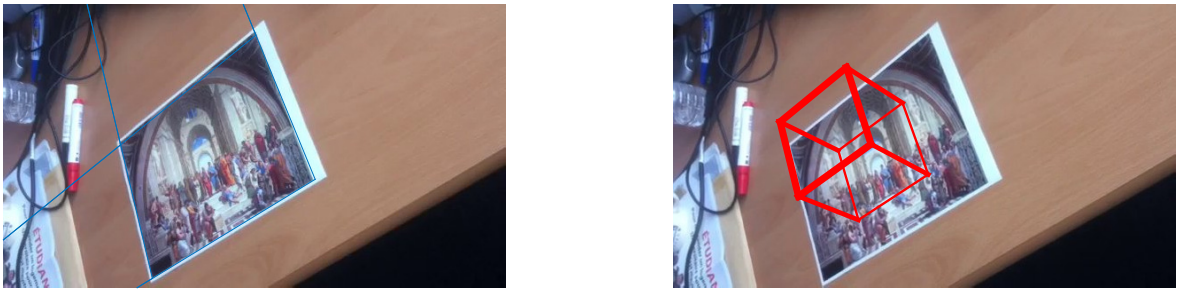

Figure 2

la première image de cette séquence, d'appeler la fonction `calibrate()` et enfin de projeter un cube au milieu de la feuille A4 en utilisant la matrice de projection retournée par la fonction `calibrate()` (Fig. 2). La fonction

$$[\mathtt{P}, \mathtt{H}, \mathtt{K}, \mathtt{s}] = \mathtt{calibrate}(\mathtt{I})$$

a pour but de calculer, à partir des sommets d'un rectangle désignés dans une image `I` :

- la matrice de projection $\mathbf{P}$ permettant de projeter le rectangle dans l'image,
- le rapport d'aspect s du rectangle,
- l'homographie $\mathbf{H}_{w0}$ entre les sommets du rectangle $(0,0)$, $(1,0)$, $(1,s)$, $(0,s)$ et les coins désignés dans l'image,
- la matrice $\mathbf{K}$ des paramètres intrinsèques de la caméra.

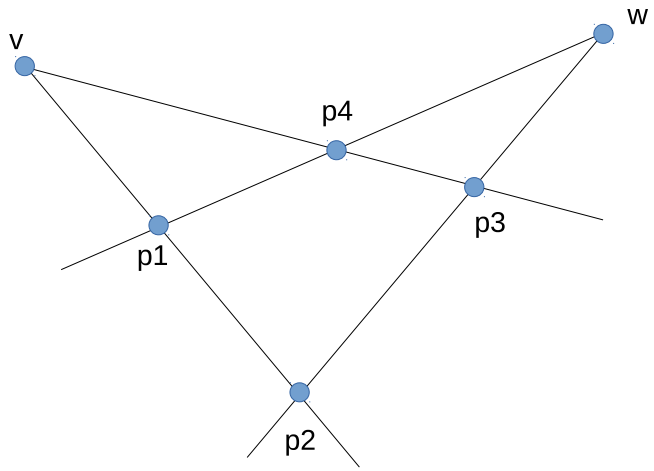

Figure 3

Le code déjà écrit permet de définir 4 points en cliquant dans l'image et d'appeler les fonctions que vous devez écrire. La première fonction à écrire est la fonction

$$[\mathtt{v}, \mathtt{w}] = \mathtt{vanishpoints}(\mathtt{corners}).$$

`corners` est un tableau de taille 2 lignes, 4 colonnes, comprenant les coordonnées (u, v) de chaque point, rangées colonne par colonne. `v` et `w` sont des vecteurs colonnes de taille 2 comprenant les coordonnées des deux points de fuite (cf. Fig. 3). Vous devez utiliser les résultats de géométrie projective vus en cours pour calculer `v` et `w` en utilisant une succession de produits vectoriels (`cross` en Matlab).

Le code correspondant au calcul de la focale f à partir des deux points de fuite `v` et `w` vous est donné (cf. formules étudiées en cours). Une fois que f est connue, le calcul de la matrice

$$K = \begin{pmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{pmatrix}$$

est immédiat et donné dans le code.

3 Calcul du point de vue initial

3.1 Calcul d'une homographie entre quatre points

Une homographie 2D est une matrice 3x3 qui transforme un ensemble de points 2D exprimés en coordonnées homogènes en un autre ensemble de points exprimés eux aussi en coordonnées homogènes. Par exemple, lorsque la caméra subit une rotation pure, tous les points de la scène sont transformés entre deux images consécutives selon la même homographie, quelque soit leur position dans l'espace. Ou encore, lorsque la caméra subit un mouvement quelconque (rotation + translation), tous les points de l'image situés sur un même plan dans l'espace sont transformés selon la même homographie. C'est cette dernière propriété que nous allons démontrer et utiliser.

Nous avons vu en cours qu'un point (X_i, Y_i, Z_i) exprimé dans le repère monde est projeté dans l'image en un point de coordonnées (x_i, y_i) donné par :

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \lambda_i \mathbf{P} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix} = \lambda_i \begin{pmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ Z_i \\ 1 \end{pmatrix},$$

où λ_i est un facteur d'échelle lié à l'utilisation des coordonnées homogènes. Supposons que les points (X_i, Y_i, Z_i) soient coplanaires. Sans perte de généralité, nous pouvons considérer qu'ils appartiennent au plan $Z = 0$. Nous obtenons alors :

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \lambda_i \begin{pmatrix} P_{11} & P_{12} & P_{14} \\ P_{21} & P_{22} & P_{24} \\ P_{31} & P_{32} & P_{34} \end{pmatrix} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix},$$

c'est-à-dire :

$$\begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix} = \lambda_i \mathbf{H}_{w0} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix}, \tag{1}$$

où $\mathbf{H}_{w0}$ est une matrice 3x3. Supposons maintenant que les mêmes points coplanaires soient projetés dans une seconde image. Nous obtenons alors des points (x'_i, y'_i) donnés par :

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \lambda'_i \mathbf{H}_{w1} \begin{pmatrix} X_i \\ Y_i \\ 1 \end{pmatrix}, \tag{2}$$

où $\mathbf{H}_{w1}$ est une autre homographie. En combinant les équations (1) et (2), on obtient :

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \lambda'_i/\lambda_i \mathbf{H}_{w1} \mathbf{H}_{w0}^{-1} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix},$$

c'est-à-dire :

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \alpha_i \mathbf{H}_{01} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}, \tag{3}$$

où $\alpha_i = \lambda'_i/\lambda_i$ et $\mathbf{H}_{01} = \mathbf{H}_{w1}\mathbf{H}_{w0}^{-1}$ est l'homographie qui transforme les points coplanaires de l'image 0 vers l'image 1 (CQFD).

On suppose à présent que l'on connaît quatre paires de points $\{(x_i, y_i) \leftrightarrow (x'_i, y'_i)\}_{1 \leq i \leq 4}$ qui se correspondent entre l'image 0 et l'image 1, et l'on cherche à calculer l'homographie $\mathbf{H}_{01}$. Soient h_{ij} les coefficients de cette homographie. Chaque paire de points donne trois équations :

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \alpha_i \begin{pmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix},$$

c'est-à-dire :

$$\begin{pmatrix} x'_i \\ y'_i \\ 1 \end{pmatrix} = \alpha_i \begin{pmatrix} h_{11}x_i + h_{12}y_i + h_{13} \\ h_{21}x_i + h_{22}y_i + h_{23} \\ h_{31}x_i + h_{32}y_i + h_{33} \end{pmatrix}.$$

De la troisième équation, on déduit $\alpha_i = 1/(h_{31}x_i + h_{32}y_i + h_{33})$, et nous obtenons finalement deux équations à 9 inconnues :

$$\begin{array}{rcl} x'_i(h_{31}x_i + h_{32}y_i + h_{33}) & = & h_{11}x_i + h_{12}y_i + h_{13} \\ y'_i(h_{31}x_i + h_{32}y_i + h_{33}) & = & h_{21}x_i + h_{22}y_i + h_{23} \end{array},$$

ce qui peut s'écrire sous forme de calcul matriciel :

$$\begin{pmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \\ 0 & 0 & 0 & x_i & y_i & 1 & -y'_i x_i & -y'_i y_i & -y'_i \end{pmatrix} \mathbf{h} = \mathbf{0},$$

où $\mathbf{h}$ est le vecteur des inconnues $\begin{pmatrix} h_{11} & h_{12} & h_{13} & h_{21} & h_{22} & h_{23} & h_{31} & h_{32} & h_{33} \end{pmatrix}^T$. Grâce aux quatre correspondances de points, on obtient enfin le système d'équations linéaires

$$\mathbf{A}\mathbf{h} = \mathbf{0},$$

où $\mathbf{A}$ est une matrice de taille 8x9. Le vecteur $\mathbf{h}$, dont on obtient l'homographie $\mathbf{H}_{01}$, est alors l'espace nul de $\mathbf{A}$ (`h = null(A)` en Matlab).

Compléter la fonction

$$[\mathtt{H}] = \mathtt{homography}(\mathtt{points1}, \mathtt{points2})$$

qui permet de calculer l'homographie `H` à partir des tableaux de points `points1` et `points2`, de taille 2x4, qui contiennent les coordonnées de quatre points dans l'image 1 et de leurs homologues dans l'image 2.

3.2 Calcul de s, $\mathbf{P}$ et $\mathbf{H}_{w0}$

Nous utiliserons la fonction `homography` pour calculer les homographies inter-images lors du tracking de caméra. Pour le moment, nous allons l'utiliser pour calculer l'homographie world-image $\mathbf{H}_{w0}$ qui transforme les quatre sommets du rectangle $(0,0)$, $(1,0)$, $(1,s)$, $(0,s)$ vers les quatre coins désignés dans le repère image (équation (1)). En fait, le rapport d'aspect s du rectangle n'est pas connu (nous allons le calculer). Nous commençons donc par calculer l'homographie $\mathbf{H}$ entre les sommets du *carré* $(0,0)$, $(1,0)$, $(1,1)$, $(0,1)$ et les coins de l'image (cf. fichier `calibrate.m`) puis, en remarquant que

$$\begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & s & s \\ 1 & 1 & 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{pmatrix},$$

nous obtenons

$$\mathbf{H} = \mathbf{H}_{w0}\mathbf{D}, \tag{4}$$

où $\mathbf{D} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{pmatrix}$. Or, comme nous l'avons vu lors du passage à l'équation (1),

$$\mathbf{H}_{w0} = \begin{pmatrix} P_{11} & P_{12} & P_{14} \\ P_{21} & P_{22} & P_{24} \\ P_{31} & P_{32} & P_{34} \end{pmatrix}. \tag{5}$$

D'autre part,

$$\mathbf{P} = \begin{pmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{pmatrix} = \mathbf{K}\,(\mathbf{R}|\mathbf{t}) = \mathbf{K}\,(\mathbf{r}_1|\mathbf{r}_2|\mathbf{r}_3|\mathbf{t})\,, \tag{6}$$

où $\mathbf{r}_1$, $\mathbf{r}_2$, $\mathbf{r}_3$ sont les trois colonnes de la rotation $\mathbf{R}$. En rassemblant les équations (4),(5) et (6), on obtient :

$$\mathbf{H} = \mathbf{K}\,(\mathbf{r}_1|\mathbf{r}_2|\mathbf{t})\,\mathbf{D} = \mathbf{K}\,(\mathbf{r}_1|s\mathbf{r}_2|\mathbf{t})\,,$$

c'est-à-dire :

$$(\mathbf{r}_1|s\mathbf{r}_2|\mathbf{t}) = \mathbf{K}^{-1}\mathbf{H} = (\mathbf{h}_1|\mathbf{h}_2|\mathbf{h}_3)\,.$$

Comme $||\mathbf{r}_1|| = ||\mathbf{r}_2||$ (toutes les colonnes d'une matrice de rotation sont de norme 1), on obtient $s = ||\mathbf{h}_2||/||\mathbf{h}_1||$ (`norm` en matlab). Une fois que s est connu, on obtient $\mathbf{r}_2 = \mathbf{h}_2/s$, et enfin :

$$\mathbf{P} = \mathbf{K}\,(\mathbf{r}_1|\mathbf{r}_2|\mathbf{r}_1 \times \mathbf{r}_2|\mathbf{t})\,.$$

Compléter la fonction

$$[\mathtt{P}, \mathtt{s}] = \mathtt{H2P}(\mathtt{H}, \mathtt{K})$$

qui calcule le rapport d'aspect s et la matrice de projection $\mathbf{P}$ à partir de $\mathbf{H}$ et de $\mathbf{K}$ selon la méthode proposée. Cette fonction est appelée à la fin de la fonction `calibrate`, juste avant le calcul de $\mathbf{H}_{w0} = \mathbf{H}\mathbf{D}^{-1}$ (d'après l'équation (4)) qui est donné dans le code.

Vous devriez à présent pouvoir exécuter le fichier `run.m` qui appelle la fonction `calibrate()` et projette un cube rouge en utilisant la matrice de projection $\mathbf{P}$ calculée (Fig. 2).

4 Mise à jour de la matrice de projection

4.1 Calcul de la matrice de projection dans une image i

On cherche maintenant à calculer les matrices de projection correspondant à toutes les images de la séquence. Nous venons de voir comment calculer la matrice de projection pour l'image 0, voyons comment la calculer pour l'image 1. Comme nous l'avons vu (cf. équation (3)), les points de l'image 0 sont transformés dans l'image 1 selon l'homographie $\mathbf{H}_{01} = \mathbf{H}_{w1}\mathbf{H}_{w0}^{-1}$. En conséquence, si on est capable de calculer l'homographie $\mathbf{H}_{01}$ à partir d'un appariement de points entre l'image 0 et l'image 1, on a immédiatement $\mathbf{H}_{w1} = \mathbf{H}_{01}\mathbf{H}_{w0}$ ($\mathbf{H}_{w0}$ étant connue grâce à l'étape initiale). La matrice de projection correspondant à l'image 1 est alors obtenue en appelant la fonction $[\mathtt{P}, \mathtt{s}] = \mathtt{H2P}(\mathtt{H}, \mathtt{K})$ avec la matrice $\mathbf{H}_{w1}$. Le problème se résume donc à calculer $\mathbf{H}_{01}$.

Pour l'image 2, on utilise $\mathbf{H}_{02} = \mathbf{H}_{w2}\mathbf{H}_{w0}^{-1}$, qui permet d'obtenir $\mathbf{H}_{w2}$ à partir de $\mathbf{H}_{02}$, puis la matrice de projection dans l'image 2. Deux stratégies sont toutefois possibles pour le calcul de la matrice $\mathbf{H}_{02}$:

1. Soit on calcule l'homographie $\mathbf{H}_{12}$ à partir d'un appariement de points entre l'image 1 et l'image 2, et on en déduit $\mathbf{H}_{02}$ par composition de matrices : $\mathbf{H}_{02} = \mathbf{H}_{12}\mathbf{H}_{01}$,
2. Soit on calcul directement l'homographie $\mathbf{H}_{02}$ à l'aide d'un appariement de points entre l'image 0 et l'image 1.

Pour une image $i > 2$ quelconque, on retrouve les deux mêmes stratégies :

1. Soit on calcule l'homographie $\mathbf{H}_{i-1,i}$ à partir d'un appariement de points entre l'image $i-1$ et l'image i, et on en déduit $\mathbf{H}_{0i}$ par composition de matrices : $\mathbf{H}_{0i} = \mathbf{H}_{i-1,i}\mathbf{H}_{i-2,i-1}\ldots\mathbf{H}_{12}\mathbf{H}_{01}$,
2. Soit on calcul directement l'homographie $\mathbf{H}_{0i}$ à l'aide d'un appariement de points entre l'image 0 et l'image i.

La stratégie 1 a pour avantage que l'appariement de points entre deux images proches est plus facile qu'entre deux images éloignées. En revanche, à chaque calcul d'homographie $\mathbf{H}_{i-1,i}$ une erreur d'imprécision inévitable se propage et au bout d'un certain nombre d'images on peut assister à un problème de dérive. À l'inverse, la stratégie 2 ne pose pas de problème de dérive, mais l'appariement de points entre images distantes est plus compliqué, voire impossible quand il n'y a plus de points communs entre les deux images. Dans ce TP, les deux stratégies seront implémentées et comparées. Dans les deux cas il est nécessaire d'apparier des points entre deux images et de calculer une homographie à partir des paires de points obtenus, ce que nous voyons dans la section suivante.

4.2 Appariement de points d'intérêt et calcul d'homographie par RANSAC

L'appariement de points d'intérêt repose sur deux étapes :

1. la détection des points d'intérêt,
2. leur appariement basé sur des descripteurs.

La toolbox Computer Vision System permet de détecter différents types de points d'intérêt : FAST, Good Features to Track, Harris, BRISK, SURF, MSER (cf. http://fr.mathworks.com/help/vision/ug/local-feature-detection-and-extraction.html). Elle propose aussi différents types de descripteurs : HOG, LBP, SURF, FREAK, BRISK et Block. Elle offre enfin la fonction `matchFeatures(features1,features2)` qui permet d'apparier deux listes de points d'intérêt détectés et décrits en utilisant les méthodes précédentes (cf. http://fr.mathworks.com/help/vision/ref/matchfeatures.html).

Une fois que n points ont été appariés, il reste à calculer l'homographie qui transforme les n points de la première image en leurs homologues dans la seconde image. Nous avons écrit une méthode de calcul d'homographie à partir de quatre correspondances de points, nous voulons à présent exploiter le fait que nous avons n correspondances de points. Nous pourrions tirer quatre correspondances au hasard parmi les n et calculer l'homographie à partir de ces seules paires de points. Cette méthode serait toutefois hasardeuse et non robuste car il est fréquent que des points soient mal appariés entre deux images. Nous allons donc utiliser la méthode RANSAC (Random Sample Consensus) qui procède ainsi :

On répète itérativement `ntests` de fois :

1. tirage aléatoire de 4 paires de points,
2. calcul d'une homographie $\mathbf{H}$ à partir des quatre points,
3. transformation de tous les points de l'image 1 par l'homographie $\mathbf{H}$ et calcul de la distance euclidienne entre les points transformés et les points correspondant dans l'image 2,
4. comptage du nombre de correspondances dont la distance est inférieur au seuil `thres`.

Au final, on retourne l'homographie qui a obtenu le plus grand nombre de points (consensus) à l'étape 4.

Le code qui vous est donné implémente la stratégie 1 de tracking (section 4.1) à l'aide de points de Harris décrits par des blocs. Adaptez ce code de manière à :

1. remplacer la stratégie 1 par la stratégie 2 (toujours avec Harris/Blocks). Normalement, le tracking doit "planter" au bout de quelques images car l'appariement de blocs n'est pas résistant à des changements de perspective très importants ;
2. toujours en utilisant la stratégie 2, remplacer Harris/Blocks par d'autres combinaisons Détecteur/Descripteur, plus invariantes aux changements de point de vue.

Comparez les différentes stratégies et faites un petit compte rendu de vos observations. En particulier, n'hésitez pas à consulter les articles référencés en bas des deux pages web indiquées ci-dessus ou au moins wikipedia pour vous faire une idée des différents détecteurs et descripteurs, de manière à pouvoir expliquer vos observations.