TDM : Vision stéréo, interpolation de vue

ENSEIRB-MATMECA

Résumé :

L'objectif de ce TD sur machine est de mettre en œuvre une chaîne d'interpolation de vues permettant à partir de deux vues en stéréo calibrée d'estimer les cartes de disparité associées, et d'en déduire un ensemble de vues interpolées simulant un point de vue intermédiaire entre les vues de départ.

1 Motivation et objectifs

L'interpolation de vues consiste à synthétiser une vue prise à partir d'un point situé sur le segment défini par les points de vue des images initiales. Ceci est nécessaire par exemple pour les écrans autostéréoscopiques à vues multiples, qui doivent être capables de générer à partir de 2 vues stéréo classiques un nombre $n>2$ de vues intermédiaires afin de pouvoir les projeter, donnant ainsi l'illusion d'un effet de parallaxe plus continu lorsqu'on se déplace latéralement devant l'écran 3D. L'interpolation de vues est également utilisée pour générer un point de vue pour lequel on ne dispose pas de caméra, pour un effet d'animation plus naturel entre deux vues (effets spéciaux cinéma ou clips), ou donner un point de vue original sur une scène (retransmission sportive).

Les objectifs de ce TDM sont d'analyser une paire d'images stéréoscopiques afin d'obtenir
- une estimation de la disparité et de la profondeur des objets de la scène
- la prédiction d'une image intermédiaire par compensation de la disparité
- la génération d'une séquence d'images faisant l'interpolation de vues entre les deux vues initiales

L'archive contenant le code fourni et les images à traiter est disponible à l'adresse suivante :

`https://sites.google.com/site/guillaumebourmaud/home/teaching`

2 Estimation de disparité

1) Dessiner le schéma géométrique de l'acquisition de deux vues stéréo à partir de deux caméras de mêmes paramètres intrinsèques et dont les paramètres extrinsèques diffèrent uniquement par une translation parallèle à l'axe horizontal x, sans rotation.
Montrer dans ce cas que la position correspondant à un pixel (x,y) dans l'une des images est contrainte à se situer sur une ligne horizontale d'ordonnée y dans la deuxième image.

2) Un squelette de script Matlab est fourni : script_stereo.m
Ce squelette charge deux images I1 et I2 (tsukuba2.png et tsukuba4.png), et fournit les commandes de visualisation des résultats.
→ Observer `I1` et `I2` et en déduire la disparité maximale entre les deux, que l'on notera `maxs` (on pourra utiliser le zoom synchronisé par `linkaxes`).
Compléter le code aux endroits indiqués, en suivant les indications suivantes.

3) On peut calculer la disparité de l'image 1 vers l'image 2 de la façon suivante :

```
D1 = estimate_disparity(I1,I2, 0, maxs, win_size);
```

où `[0, maxs]` représente l'intervalle de recherche de la disparité (toutes les valeurs entières de l'intervalle sont testées pour trouver le SAD minimal).

win_size représente la taille de la fenêtre carrée sur laquelle est calculée la similarité visuelle (on pourra prendre dans un premier temps `win_size=16`).
La disparité renvoyée est celle pour laquelle le voisinage du pixel (i1,j1) de l'image I1 et le voisinage du pixel (i1,j1-D1(i1,j1)) correspondant dans l'image I2 sont le plus similaires. (Note : attention à la convention i=ligne=y, j=colonne=x, i.e. I(i,j)=I(y,x)).
→ Dessiner un schéma qui définisse les conventions précédentes de D1 de façon la plus précise et générique possible. On fera apparaître en particulier les matrices I1, I2, D1 ainsi que l'interprétation géométrique des valeurs h,w (taille des images), i1,j1 et D1(i1,j1). Bien différentier le référentiel lié à I1 et celui lié à I2.

4) De même, on peut calculer la disparité en sens inverse de la façon suivante :

```
D2 = estimate_disparity(I2,I1, -maxs, 0, win_size);
```

Il est nécessaire de préciser un intervalle de recherche avec un décalage négatif pour la fonction estimate_disparity. Note : la fonction renvoie la valeur absolue de la disparité (D2>=0), on a donc que I2(i2,j2) correspond à I1(i2,j2+D2(i2,j2)).
→ Dessiner le schéma qui définit les conventions géométriques pour D2 (faire apparaître I1, I2, D2, h,w, i2, j2, D2(i2,j2)).
→ Expliquer pourquoi D1(i,j) et D2(i,j) ne peuvent pas être associés. Si D1 et D2 sont cohérents l'un avec l'autre, établir une relation entre eux de la forme D1(?, ?)=D2(?, ?).

5) Rappeler sur feuille quel est le lien entre la disparité *d* et la profondeur *z*.
Pour Tsukuba2-4, on pourra supposer que la focale est f=380 pixels et la baseline b=5cm.
Calculer la carte de profondeur `Z1` pour l'image 1 à partir de la carte de disparité `D1`.
L'afficher sous la forme d'une surface 3D (voir code fourni).

6) Vérifier que les disparités estimées sont correctes en prédisant l'image 1 à partir de l'image 2.
On utilisera pour cela la fonction interp2 (ou la version interp2color.m fournie), qui s'utilise de la façon suivante :
- A l'aide de meshgrid, il est possible de définir deux matrices X1 et Y1 qui associent à chaque case son ordonnée et son abscisse : X1(i1,j1)=j1 et Y1(i1,j1)=i1.

```
[X1,Y1]=meshgrid(1:w,1:h);
```

Définir à partir de X1 et Y1 deux matrices X2 et Y2 qui associent à chaque pixel (i1,j1) de I1 la coordonnée correspondante (i2,j2) dans I2: X2(i1,j1)=j2 et Y2(i1,j1)=i2.
Quelle carte de disparité faut-il utiliser pour cela ? D1 ou D2 ? Justifier à l'aide d'un des schémas des questions 3 et 4 ou d'équations.

```
X2 = ???;  Y2 = ???;
```

- Appliquer interp2 pour prédire I1 à partir de I2

```
I1p = interp2color(I2,  X2,Y2);
```

→ Afficher la différence entre I1 et I1p et vérifier que la plupart de l'image est correctement prédite. Où se trouvent les erreurs principales ? Pourquoi ?

3 Interpolation de vue

Afin de générer une vue interpolée Im entre les deux images I1 et I2, nous allons devoir générer une carte de déformation appropriée, dépendant d'un paramètre *a* continu entre 0 et 1.
a=0 correspond à une image interpolée identique à I1, a=1 correspond à une image interpolée identique à I2, et les valeurs intermédiaires correspondent à des points de vues se trouvant entre les deux points de vue initiaux.

Le modèle d'interpolation des déplacements est linéaire : la disparité dépend linéairement du paramètre a, ce qui permet de calculer la carte de disparité de I1 vers Im comme a*D1, et la carte de disparité de I2 vers Im comme (1-a)*D2.

1) L'image tsukuba3.png contient une vue intermédiaire correspondant au paramètre a=0.5.
Générer les cartes de transformation [Xm_1,Y] associant à chaque pixel (i1,j1) de I1 les coordonnées (Y(i1,j1),Xm_1(i1,j1)) dans Im. Utiliser ces cartes pour prédire I1 à partir de Im, et afficher l'erreur.

2) Ces cartes ne permettent pas de déformer I1 vers Im. Pour cela, il faut inverser le modèle de déformation, ce qui peut être fait avec la fonction fournie reversewarpx.m :

```
X1_m = reversewarpx(Xm_1);
```

On obtient ainsi la transformation de coordonnées dans le sens inverse.
Utiliser les cartes de transformation ainsi obtenues pour prédire Im à partir de I1 seul, puis à partir de I2 seul. Combiner les deux prédictions en les moyennant.

3) Généraliser le procédé afin de prédire n'importe quelle image intermédiaire, fonction du paramètre a, en moyennant de façon appropriée les images prédites à partir de I1 et I2.
Puis générer une séquence d'images intermédiaires pour a variant de 0 à 1, et les enregistrer dans un tableau de cellule. Afficher l'animation permettant de passer de façon continue de l'image I1 à I2 à l'aide de ces images interpolées. On pourra utiliser la fonction loop_im_seq.m fournie, qui affiche une séquence stockée dans un tableau de cellules.

4) Appliquer l'algorithme à d'autres paires d'images : par exemple cones/im2.png et cones/im6.png.