BE – Stéréo vision – estimation de profondeur – appariements parsemé et dense

Les algorithmes d'appariement peuvent être scindés en deux grandes groupes qui sont, appariement parsemé et dense. Dans les deux cas, une connaissance du système stéréo est nécessaire.

Pour la première catégorie, les paramètres intrinsèques de chaque caméra ne sont souvent pas connus, l'étalonnage (faible) donne lieu à l'estimation de la matrice fondamentale qui établit le lien les deux images. Ensuite, des points ou régions d'intérêt sont détectés et extraits avant leur appariement.

Pour la catégorie dense, les paramètres intrinsèques et extrinsèques sont estimés, étalonnage fort, sous forme de matrices de projection et matrice essentiel suivis de rectification des images avant de procéder à l'appariement dense stéréo. La taxonomie d'appariement dense proposée par Scharstein et Szeliski (2002) propose les quatre étapes suivantes :

1. calcul des coûts d'appariement ;

2. agrégation des coûts (support) ;

3. calcul et optimisation des disparités ; et

4. raffinement des disparités.

Q1. En vous appuyant sur la littérature, définissez formellement la matrice fondamentale, ses dimensions. Comment cette matrice est estimée. Proposez un algorithme simple pour la calculer ?

Q2. En vous appuyant sur la littérature, définissez formellement la matrice essentielle et ses dimensions. Comment cette matrice est estimée. Proposer un algorithme simple pour la calculer ?

Q3. Qu'est-ce que la rectification ? Pourquoi l'on rectifie les images et pourquoi ?

Appariement parsemé

Nous nous proposons d'étudier et d'appliquer les étapes de l'appariement parsemé dans l'environnement Matlab. Soient les images suivantes dont les effets dus à l''aberration optique ont été corrigées :

Figure 1 - Images du Globe : images gauches et droites rectifiées. Matlab.

Ouvrez ces images nommées respectivement I1op.bmp et I2op.bmp. Affichez ces images grâce à l'instruction :

```
I1=imread('I1op.bmp');
I2=imread('I1op.bmp');

figure;
imshowpair(I1, I2,'montage');
```

Vous pouvez aussi les afficher en les superposant par l'instruction :

```
figure;
imshow(stereoAnaglyph(I1,I2));
```

Q4. Ces images sont-elles rectifiées ? Justifiez votre réponse ?

Utilisez la fonction `detectSURFFeatures()` de matlab pour détecter des points d'intérêt dans chacune des images. Variez le paramètres `'MetricThreshold'` et constater la variation du comportement. Variez votre détecteur de point d'intérêt en utilisant d'autres proposé par Matlab.

Q5. Que constate-t-on ?

Utilisez la fonction `extractFeatures()` pour extraire les descripteurs des points d'intérêt dans chacune des images.

Utilisez la fonction `matchFeatures` pour apparier les points d'intérêt représentés chacun par son descripteur.

Q6. Quelles sont les sorties de cette fonction ?

```
corners1 = detectSURFFeatures(rgb2gray(I1));
[features1, valid_corners1] = extractFeatures(rgb2gray(I1), corners1);
corners2 = detectSURFFeatures(rgb2gray(I2));
[features2, valid_corners2] = extractFeatures(rgb2gray(I2), corners2);
[indexPairs,matchmetric] = matchFeatures(features1,features2);
```

Dans la suite, nous retenons que les points appariés. Appliquons les instructions suivantes à cet effet.

```
matchedPoints1 = valid_corners1(indexPairs(:,1));
matchedPoints2 = valid_corners2(indexPairs(:,2));
```

Visualisons les points appariés dans les images superposées :

```
Figure;
showMatchedFeatures(I1, I2, matchedPoints1, matchedPoints2);
% title('Tracked Features');
```

Q7. Que constatez-vous ?

Pour aller plus loin dans l'estimation de profondeur des points d'intérêts nous avons besoins de vérifier leur exactitude en les comparant à un modèle du système stéréo que nous allons estimer. Pour ce faire, nous allons utiliser la fonction `estimateEssentialMatrix()` pour estimer la matrice essentielle qui dans ce cas n'est autre que la matrice fondamentale. Nous allons alors utiliser la matrice fondamentale f :

$$F = \begin{pmatrix} f_{11} & f_{12} & f_{13} \\ f_{21} & f_{22} & f_{23} \\ f_{31} & f_{32} & f_{33} \end{pmatrix}$$

pour vérifier l'exactitudes des appariements. Les points vérifiant l'équation épipolaire suivante sont considérés comme pertinents (*inliers*). Ceux ne le vérifiant, comme aberrants (*outliers*) et donc écartés dans la suite.

$$(x_i, y_i, w_i)\, F \begin{pmatrix} x'_i \\ y'_i \\ w'_i \end{pmatrix} = 0$$

```
Load cp;
[E, epipolarInliers] = estimateEssentialMatrix(...
    matchedPoints1, matchedPoints2, cameraParams, 'Confidence', 99.99);
```

ou dans le cas où les paramètres des caméras ne sont pas disponibles :

```
[F, epipolarInliers] = estimateFundamentalMatrix (...
    matchedPoints1, matchedPoints2);
```

suivi de la selection des inliers et affichage.

```
inlierPoints1 = matchedPoints1(epipolarInliers, :);
inlierPoints2 = matchedPoints2(epipolarInliers, :);
showMatchedFeatures(I1, I2, inlierPoints1, inlierPoints2);
title('Epipolar Inliers');
```

Q8. Variez le paramètre `'Confidence'` et constatez le comportement dans les résultats.

Le résultat final est illustré dans la figure suivante.

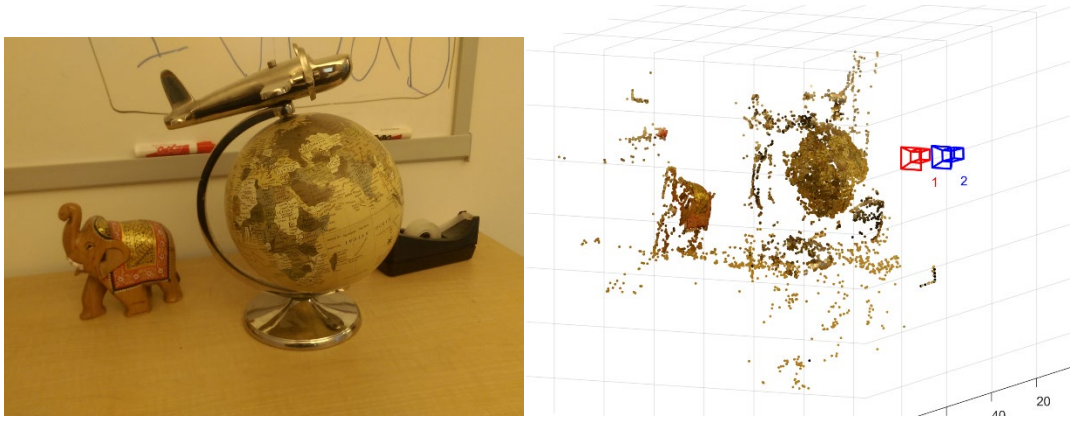

Figure 2 – Résultats finaux en 3D, obtenus sur les images proposées. A gauche, image gauche. A droite le résultat en 3D.

Appariement dense

La taxonomie des algorithmes d'appariement dense a été pour la première fois proposé par Scharstein et Szeliski (2002). Elle est basée sur l'observation que les algorithmes stéréo exécutent généralement un sous-ensemble des quatre étapes suivantes :

1. calcul des coûts d'appariement ;

2. agrégation des coûts (support) ;

3. calcul et optimisation des disparités ; et

4. raffinement des disparités.

Ces algorithmes utilisent des fenêtres de support (ou simplement support), généralement centré sur le pixel pour lequel le calcul de disparité est en cours. Certains de ces algorithmes peut être proprement décomposé en étapes 1, 2, 3 suivantes. Par exemple, les différences de somme des carrés traditionnelles (SSD) peut être décrit comme :

1. Le coût d'appariement est la différence quadratique des valeurs d'intensité pour une disparité donnée.

2. L'agrégation se fait en additionnant le coût correspondant sur les fenêtres carrées à disparité constante.

3. Les disparités sont calculées en sélectionnant la valeur agrégée minimale (gagnante).

Dans la suite, nous allons expérimenter ces étapes grâce à une série de fonctions Matlab est fournie.

Soient les images de texture suivantes dont les effets d'aberrations optiques ont été corrigées et sont rectifiées :

Figure 3 – Middlebury 2003 Stereo datasets with ground truth

Ces images sont issues de la base Middlebury, sont nommées im2.bmp et im6.bmp. Les images disp2.bmp et disp6.bmp sont les vérités terrain. Les vérités terrain peuvent être utilisées à chaque étape pour estimer la pertinence de la solution et le calcul d'erreur. Ouvrez et affichez ces images grâce à l'instruction :

```
G=imread('im2.bmp');
D=imread('im6.bmp');

figure;
imshowpair(G, D,'montage');
```

Vous pouvez aussi les afficher en les superposant par l'instruction :

```
figure;
imshow(stereoAnaglyph(rgb2gray(G),rgb2gray(D)));
```

Appliquons la function `calculate_cost(leftImage, rightImage, minDisp, maxDisp)` qui prend quatre arguments.

```
Mindisp = 0;
Maxdisp = 64;
[CoutG, CoutD] = calculate_cost(G, D, mindisp, maxdisp);
figure; disp = 5 ; imshow(CoutG(:,:,disp), []); title('Le coût pour une
disparité donné);
```

Q9. Variez la valeur de `disp` et affichez `CoutG`. Commentez les valeurs de coût obtenues pour quelques points de profondeurs différentes dans les images de texture.

Q10. Variez la valeur de `Maxdisp` et commentez les résultats.

Appliquez à présent la séquence d'instructions suivantes qui nous permet de calculer la carte de disparité dense sans l'agrégation des coûts :

```
[DispG] = winner_takes_all(CoutG);
DispG = DispG + mindisp;
figure, imshow(DispG, [mindisp maxdisp]);
title(['Disparité gauche estimée sans agrégation des coûts : ']);
```

Q11. Commentez le résultat obtenu.

Et avec l'agrégation des coûts dans une fenêtre carré centrée sur chaque pixel avec un rayon r :

```
R = 5; % Rayon
CoutGa = aggregate_cost_block(CoutG, R);
[DispGa] = winner_takes_all(CoutGa);
DispGa = DispGa + mindisp;
figure, imshow(DispGa, [mindisp maxdisp]);
title(['Disparité gauche estimée avec agrégation - block centré de rayon R
: ' ]);
```

Q12. Quel est votre constat quant aux résultats affichés ?

Et avec l'agrégation des coûts dans une fenêtre avec application de la fonction gaussienne centrée sur chaque pixel avec un rayon r :

```
R=5; %Rayon
S=2; %Sigma
CoutGb = aggregate_cost_gauss(CoutG, R, S);
[DispGb] = winner_takes_all(CoutGb);
DispGb = DispGb + mindisp;
figure, imshow(DispLb, [mindisp maxdisp]);
title(['Disparité estimée avec agrégation gaussienne aggregation']);
clear CoutGb;
```

Q13. Variez les valeurs de R et de S(Sigma). Quel est votre constat quant aux résultats affichés ?

Q14. **Optionnelle.** Proposez et décrivez sommairement un article de l'état de l'art proposant une solution pour l'appariement parsemé ou dense par apprentissage profond.

Travail à rendre

Un compte-rendu par binôme est à produire sur le sujet de ce bureau d'étude. Il doit comporter :

- les résultats obtenus à chaque étape,
- vos codes commentés,
- vos réponses aux questions posées,
- vos analyses, synthèse et conclusion.

Il est à déposer sur Moodle dans le répertoire dédié.