Traitement d'images

Emanuel Aldea <emanuel.aldea@u-psud.fr>
http://hebergement.u-psud.fr/emi/TIPolytech

Polytech Paris-Saclay 5$^{\text{ère}}$ année

Représentation des points 3D

Dans l'espace 3D :

$$\underbrace{\mathrm{p} = (X, Y, Z)^T = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{\text{point initial}} \quad \underbrace{\mathrm{p}' = (X', Y', Z')^T = \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}}_{\text{point transformé}}$$

- six degrés de liberté (trois rotations, trois translations)
- on abandonne les ˜ pour simplifier les notations, mais les variables sont homogènes

Représentation des points 3D

Dans l'espace 3D :

$$\underbrace{\mathrm{p} = (X, Y, Z)^T = \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}}_{\text{point initial}} \quad \underbrace{\mathrm{p}' = (X', Y', Z')^T = \begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix}}_{\text{point transformé}}$$

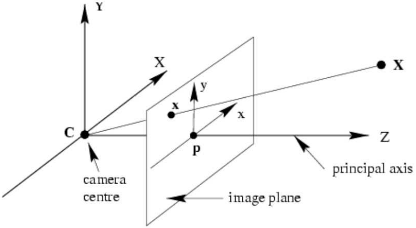
Une transformation Euclidienne avec les versions homogènes :

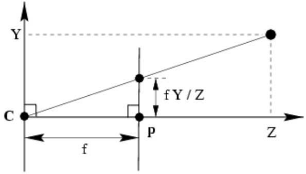
$$\begin{bmatrix} X' \\ Y' \\ Z' \\ 1 \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

ou bien $\mathrm{p}' = \begin{bmatrix} \mathrm{R} & \mathrm{t} \\ 0^T & 1 \end{bmatrix} \mathrm{p}$, avec $\mathrm{R}^T\mathrm{R} = \mathrm{I}$, $\det \mathrm{R} = 1$

- six degrés de liberté (trois rotations, trois translations)
- on abandonne les ˜ pour simplifier les notations, mais les variables sont homogènes

Le modèle de camera pinhole





Projection 3D $\Rightarrow$ 2D par une projection centrale

- Dans le plan focal 3D : $(X, Y, Z)^T \Rightarrow (fX/Z, fY/Z, f)^T$
- Dans le plan 2D de l'image : $(X, Y, Z)^T \Rightarrow (fX/Z, fY/Z) = (x, y)$

Le modèle de camera pinhole

La projection dans le plan image $(fX/Z, fY/Z)$ en coordonnées homogènes donne :

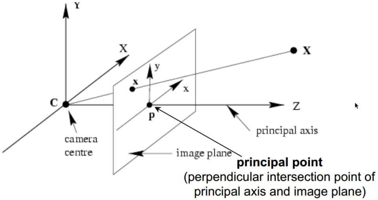
$$\begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathrm{diag}(f, f, 1)[\mathrm{I}|0]\mathrm{X}$$

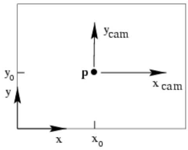
Le modèle de camera pinhole

La projection dans le plan image $(fX/Z, fY/Z)$ en coordonnées homogènes donne :

$$\begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathrm{diag}(f, f, 1)[\mathrm{I}|0]\mathrm{X}$$

La référence choisie dans le plan image n'est pas la projection de l'axe optique :





Dans le système de référence qu'on utilise habituellement :

$$(X, Y, Z) \Rightarrow (fX/Z + p_x, fY/Z + p_y)$$

Le modèle de camera pinhole

La projection dans le plan image $(fX/Z, fY/Z)$ en coordonnées homogènes donne :

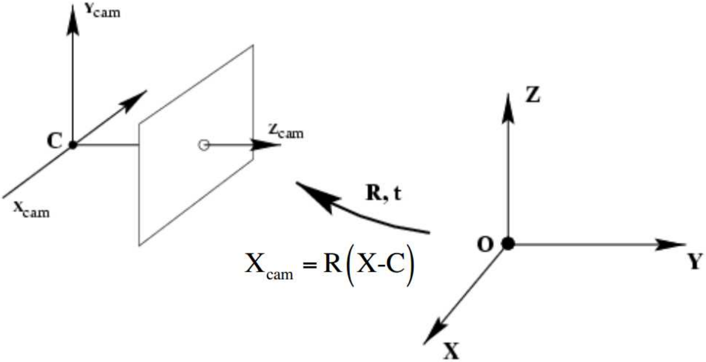
$$\begin{bmatrix} fX \\ fY \\ Z \end{bmatrix} = \underbrace{\begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}}_{\mathrm{K}} \cdot \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \cdot \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \mathrm{diag}(f, f, 1)[\mathrm{I}|0]\mathrm{X}$$

K - matrice de calibration intrinsèque

- constante tant que l'optique de la camera n'est pas ajustée
- nécessaire pour faire le passage $2D \Leftrightarrow 3D$
- habituellement, estimée en utilisant des objets (mires) de calibration

Passage dans un repère inertiel (fixe)

Dernière étape de la modélisation : on exprime les variables dans un repère qui n'est pas solidaire de la camera et qui est fixe (typiquement pour la robotique mobile) :



$$X_{cam} = R\left(X - C\right)$$

Passage dans un repère inertiel (fixe)

Dernière étape de la modélisation : on exprime les variables dans un repère qui n'est pas solidaire de la camera et qui est fixe (typiquement pour la robotique mobile) : En notant par C le centre de la camera en coordonnées "world", le passage monde vers camera s'écrit

$$X_{cam} = R(X - C) = \begin{bmatrix} R & -RC \\ 0^T & 1 \end{bmatrix} \tilde{X}$$

et donc à la place de la projection des coordonnées camera vers le plan image :

$$x = K\big[I|0\big]X_{cam}$$

on utilise la projection des coordonnées world vers le plan image :

$$x = KR\big[I| - C\big]X$$

Qu'est-ce qu'on peut déterminer là-dedans ?

Il y a d'autres problèmes à régler concernant la projection 3D $\Rightarrow$ 2D :

FIGURE – FOV étroit

Qu'est-ce qu'on peut déterminer là-dedans ?

Il y a d'autres problèmes à régler concernant la projection 3D $\Rightarrow$ 2D :

FIGURE – FOV moyen

Qu'est-ce qu'on peut déterminer là-dedans ?

Il y a d'autres problèmes à régler concernant la projection 3D $\Rightarrow$ 2D :

FIGURE – FOV large

Qu'est-ce qu'on peut déterminer là-dedans ?

Distorsions : visibles surtout pour des optique a FOV large, doivent être corrigées en s'appuyant sur des modèles de distorsion.

Stratégies pour déterminer les paramètres de la projection

Qu'est-ce qu'on peut déterminer là-dedans ?

Distorsions : visibles surtout pour des optique a FOV large, doivent être corrigées en s'appuyant sur des modèles de distorsion.

Stratégies pour déterminer les paramètres de la projection

- On utilise des objets avec des configurations connues, et on minimise une erreur de projection sur un nombre suffisant d'exemples :

Qu'est-ce qu'on peut déterminer là-dedans ?

Distorsions : visibles surtout pour des optique a FOV large, doivent être corrigées en s'appuyant sur des modèles de distorsion.

Stratégies pour déterminer les paramètres de la projection

- On utilise des objets avec des configurations connues, et on minimise une erreur de projection sur un nombre suffisant d'exemples :
- On peut utiliser directement la scène (auto-calibration)

- Z. Zhang, *A flexible new technique for camera calibration*, PAMI, 2000
- M Pollefeys, R Koch, L Van Gool, *Self-calibration and metric reconstruction inspite of varying and unknown intrinsic camera parameters*, IJCV, 1999
- A. Fusiello, *Uncalibrated Euclidean reconstruction : a review*, Image and Vision Computing , 2000

Stéréo

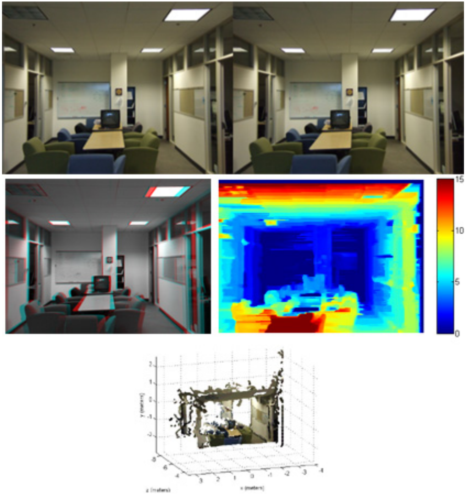
- Avec la projection, on perd l'information de profondeur.

Stéréo

- Avec la projection, on perd l'information de profondeur.
- Mais avec deux vues, on peut remonter à la valeur de Z

Stéréo

- Avec la projection, on perd l'information de profondeur.
- Mais avec deux vues, on peut remonter à la valeur de Z



Stéréo

- Avec la projection, on perd l'information de profondeur.
- Mais avec deux vues, on peut remonter à la valeur de Z

Pour simplifier, supposons que la rotation entre les deux cameras est nulle, et que la translation est purement latérale. En prenant b comme la distance entre les cameras, on obtient rapidement :

$$x_1 = \frac{fX_1}{Z}; \;\; y_1 = \frac{fY_1}{Z}; \;\; x_2 = \frac{fX_2}{Z}; \;\; y_2 = \frac{fY_2}{Z}; \;\; Y_1 = Y_2; \;\; X_1 = X_2 + b$$

D'ici, on obtient deux résultats fondamentaux : $y_1 = y_2$ (les points 3D de la scène se projettent sur des lignes correspondantes) et en notant $d = x_2 - x_1$ la **disparité** :

$$\boxed{d = \frac{fb}{Z} \Leftrightarrow Z = \frac{fb}{d}}$$

Estimation de l'imprécision

Un calcul de propagation de l'erreur de disparité ϵ_d nous montre que :

$$\epsilon_Z \approx \frac{Z^2}{bf} \cdot \epsilon_d$$

Estimation de l'imprécision

Un calcul de propagation de l'erreur de disparité ϵ_d nous montre que :

$$\epsilon_Z \approx \frac{Z^2}{bf} \cdot \epsilon_d$$

Problème : la fonction d'erreur d'estimation de Z est quadratique en Z. Par exemple, si on veut faire de la reconstruction avec un dispositif de la taille d'un téléphone :

- $b = 8cm$, $f = 750px$, $\epsilon_d \approx 2px$
- on obtient $\epsilon_Z \approx 13cm$ pour $Z = 2m$ et $\epsilon_Z \approx 83cm$ pour $Z = 5m$

Solutions rapides pour un gain limité en précision :

Estimation de l'imprécision

Un calcul de propagation de l'erreur de disparité ϵ_d nous montre que :

$$\epsilon_Z \approx \frac{Z^2}{bf} \cdot \epsilon_d$$

Problème : la fonction d'erreur d'estimation de Z est quadratique en Z. Par exemple, si on veut faire de la reconstruction avec un dispositif de la taille d'un téléphone :

- $b = 8cm$, $f = 750px$, $\epsilon_d \approx 2px$
- on obtient $\epsilon_Z \approx 13cm$ pour $Z = 2m$ et $\epsilon_Z \approx 83cm$ pour $Z = 5m$

Solutions rapides pour un gain limité en précision :

- augmenter f ... mais on réduit le FOV de travail ; en pratique en navigation robotique on fait exactement le contraire (FOV $\geq$ 135 deg)

Estimation de l'imprécision

Un calcul de propagation de l'erreur de disparité ϵ_d nous montre que :

$$\epsilon_Z \approx \frac{Z^2}{bf} \cdot \epsilon_d$$

Problème : la fonction d'erreur d'estimation de Z est quadratique en Z. Par exemple, si on veut faire de la reconstruction avec un dispositif de la taille d'un téléphone :

- $b = 8cm$, $f = 750px$, $\epsilon_d \approx 2px$
- on obtient $\epsilon_Z \approx 13cm$ pour $Z = 2m$ et $\epsilon_Z \approx 83cm$ pour $Z = 5m$

Solutions rapides pour un gain limité en précision :

- augmenter f ... mais on réduit le FOV de travail ; en pratique en navigation robotique on fait exactement le contraire (FOV $\geq$ 135 deg)
- augmenter b ... mais c'est peu pratique au delà d'une certaine valeur (10cm pour un outil de poche, 1m pour une voiture etc.) et cela réduit le FOV commun à de petites distances

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant

Voir `http://vision.middlebury.edu/stereo/eval/` pour une comparaison très large des méthodes existantes

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant
- mais l'imprécision sera supérieure aux valeurs vues précédemment, et les points qui sont aux bords des objets peuvent ne pas avoir des correspondants

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant
- mais l'imprécision sera supérieure aux valeurs vues précédemment, et les points qui sont aux bords des objets peuvent ne pas avoir des correspondants

On a également besoin de b. En pratique, pour des paires stéréo, on estime précisément C et R, qui sont proches d'un vecteur aligné à l'axe X et l'identité respectivement, mais pas identiques.

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant
- mais l'imprécision sera supérieure aux valeurs vues précédemment, et les points qui sont aux bords des objets peuvent ne pas avoir des correspondants

On a également besoin de b. En pratique, pour des paires stéréo, on estime précisément C et R, qui sont proches d'un vecteur aligné à l'axe X et l'identité respectivement, mais pas identiques.
Nouveau problème : dans ce cas, le rayons qui remontent au point 3D par ses deux projections ne s'intersectent pas forcement $\Rightarrow$ on utilise des algorithmes de triangulation qui cherchent une solution optimale par rapport à un certain critère, i.e. le point en 3D se trouvant à mi-chemin entre les deux rayons.

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant
- mais l'imprécision sera supérieure aux valeurs vues précédemment, et les points qui sont aux bords des objets peuvent ne pas avoir des correspondants

On a également besoin de b. En pratique, pour des paires stéréo, on estime précisément C et R, qui sont proches d'un vecteur aligné à l'axe X et l'identité respectivement, mais pas identiques.

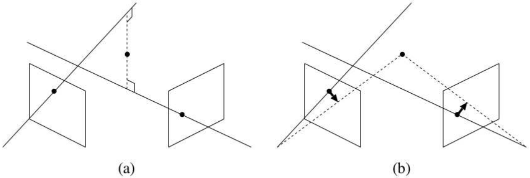


Figure 1: Triangulation. (a) The mid-point method. (b) Optimal correction.

Les données d'entrée

On a besoin de ϵ_d en entrée de l'algorithme, mais cette disparité peut être estimée directement uniquement pour des points d'intérêt :

- on peut faire de la reconstruction 3D sparse
- on peut utiliser des méthodes d'optimisation pour associer tous les pixels d'une image à l'autre en minimisant la différence de couleur entre les pixels correspondants et en régularisant le champ de disparité résultant
- mais l'imprécision sera supérieure aux valeurs vues précédemment, et les points qui sont aux bords des objets peuvent ne pas avoir des correspondants

On a également besoin de b. En pratique, pour des paires stéréo, on estime précisément C et R, qui sont proches d'un vecteur aligné à l'axe X et l'identité respectivement, mais pas identiques.

- Richard I. Hartley, and Peter F. Sturm., *Triangulation*, CVIU, 1997
- Kenichi Kanatani, Yasuyuki Sugaya and Hirotaka Niitsuma, *Triangulation from Two Views Revisited : Hartley-Sturm vs. Optimal Correction*, BMVC, 2008

Nécessité de l'invariance en TI

Les contours

- traitement relativement peu coûteux
- détection robuste de courbes paramétriques (Hough)

Nécessité de l’invariance en TI

Les contours

- traitement relativement peu coûteux
- détection robuste de courbes paramétriques (Hough)
- applications variées dans des environnements spécifiques :
  - détection de la route, des panneaux, du texte
  - imagerie médicale et satellitaire
  - inspection par vision industrielle

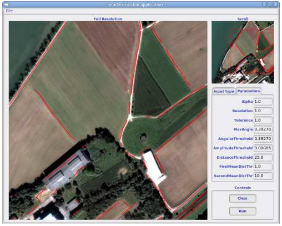
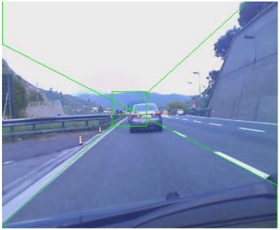
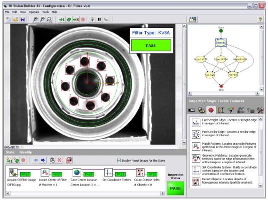


Image aérienne



Détection de voies



Vision industrielle

Nécessité de l'invariance en TI

Les contours

- traitement relativement peu coûteux
- détection robuste de courbes paramétriques (Hough)
- applications variées dans des environnements spécifiques :
  - détection de la route, des panneaux, du texte
  - imagerie médicale et satellitaire
  - inspection par vision industrielle

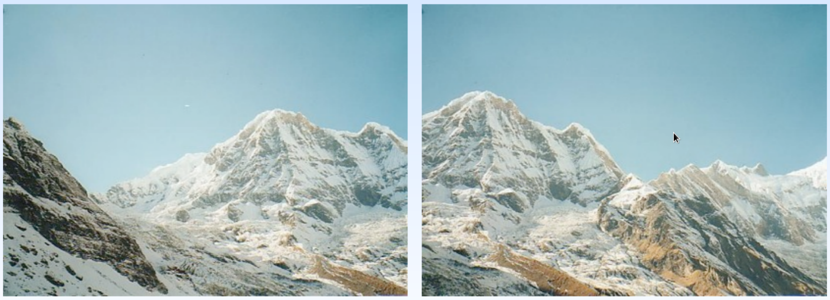
✓ tâches rapides et spécialisées

✓ invariants aux variations d'intensité

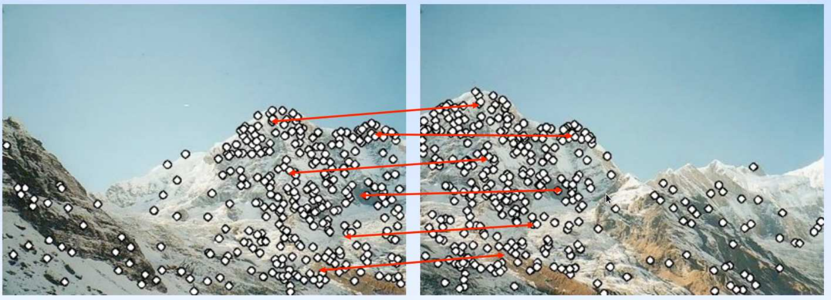
✗ sensibles aux autres transformations géométriques

✗ problème pour la reconnaissance de formes

Motivation - images panoramiques

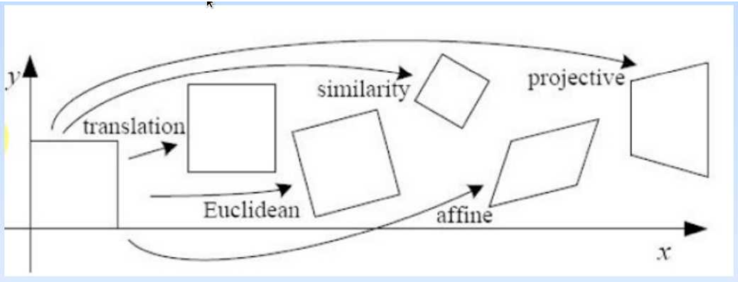


Motivation - images panoramiques



Motivation - images panoramiques

Le problème



- translation
- Euclidienne (translation + rotation)
- similarité (tr. + rot. + échelle)
- affine (rot. + échelle + shear + translation)
- projective

Nécessité de l'invariance en TI

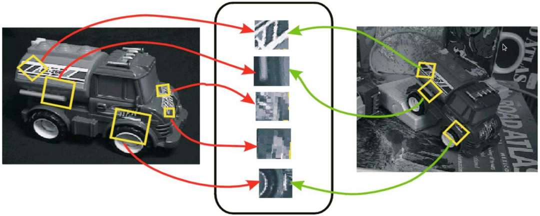
Objectif

- identifier des structures qui sont invariantes par rapport aux rotations, changements d'échelle, etc.
- ces structures sont appelées couramment points d'intérêt ou coins

Nécessité de l'invariance en TI

Objectif

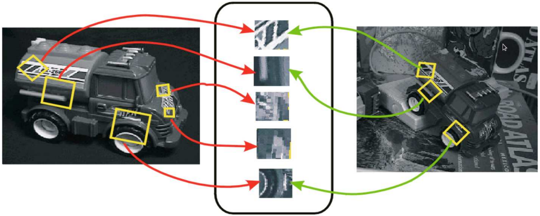
- identifier des structures qui sont invariantes par rapport aux rotations, changements d'échelle, etc.
- ces structures sont appelées couramment points d'intérêt ou coins



Nécessité de l'invariance en TI

Objectif

- identifier des structures qui sont invariantes par rapport aux rotations, changements d'échelle, etc.
- ces structures sont appelées couramment points d'intérêt ou coins



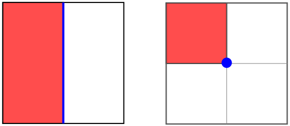
Comment faire pour :

- les identifier de manière non supervisée ?
- les associer de manière robuste ?

Détecteurs de coins : les bases

Définition

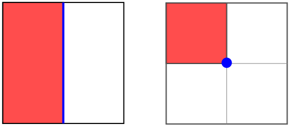
Coin : un endroit de l'image qui présente une forte variation d'intensité en deux directions différentes.



Détecteurs de coins : les bases

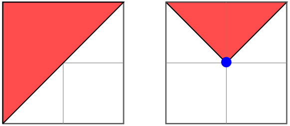
Définition

Coin : un endroit de l'image qui présente une forte variation d'intensité en deux directions différentes.



Toujours besoin de calculer les gradients locaux, mais

- pas suffisant de faire comme auparavant (dans le repère image) !



Détecteurs de coins : les bases

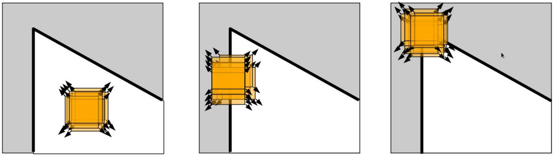
Définition

Stratégie : le contenu d'un patch centré dans le coin devrait varier dans toutes les directions

Détecteurs de coins : les bases

Définition

Stratégie : le contenu d'un patch centré dans le coin devrait varier dans toutes les directions



Comportement :

- régions homogènes : pas de changement
- contours : pas de changement le long du contour
- coins : changement important dans toutes les directions
- qualité du pixel : le plus petit changement
- introduit par Moravec en 1980
- pas isotrope : certains contours mal détectés, fausse réponse coin

Détecteurs de coins : les bases

Changement d'intensité par shift de $(\Delta x, \Delta y)$

$$E(x, y, \Delta x, \Delta y) = \sum_x \sum_y w(x, y) \left[I(x, y) - I(x + \Delta x, y + \Delta y) \right]^2$$

Détecteurs de coins : les bases

Changement d'intensité par shift de $(\Delta x, \Delta y)$

$$E(x, y, \Delta x, \Delta y) = \sum_x \sum_y w(x, y)\big[\underbrace{I(x, y)}_{\text{intensité}} - I(x + \Delta x, y + \Delta y)\big]^2$$

Détecteurs de coins : les bases

Changement d'intensité par shift de $(\Delta x, \Delta y)$

$$E(x, y, \Delta x, \Delta y) = \sum_x \sum_y w(x, y) \big[\underbrace{I(x, y)}_{\text{intensité}} - \underbrace{I(x + \Delta x, y + \Delta y)}_{\text{intensité shiftée}} \big]^2$$

Détecteurs de coins : les bases

Changement d'intensité par shift de $(\Delta x, \Delta y)$

$$E(x,y,\Delta x,\Delta y) = \sum_x \sum_y \underbrace{w(x,y)}_{\text{support}} \left[\underbrace{I(x,y)}_{\text{intensité}} - \underbrace{I(x+\Delta x, y+\Delta y)}_{\text{intensité shiftée}} \right]^2$$

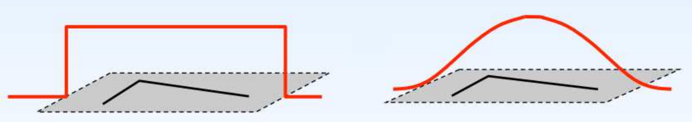


FIGURE – Types de fonction support $w(x,y)$

$E(x,y)$ large indique potentiellement un coin.

Détecteurs de coins : les bases

Approximation au premier ordre du dév. en série Taylor

$$f(x+\Delta x, y+\Delta y) = f(x,y) + f_x(x,y)\Delta x + f_y(x,y)\Delta y$$

On l'utilise pour réécrire la variation d'intensité par shift :

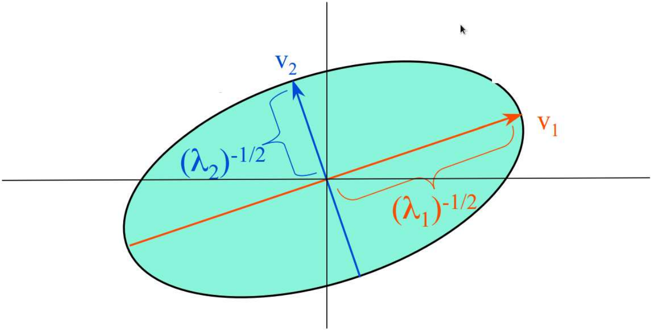
$$\begin{aligned}
\sum \left[I(x+\Delta x, y+\Delta y) - I(x,y)\right]^2 \approx & \sum \left[I(x,y) + \Delta x I_x(x,y) + \Delta y I_y(x,y) - I(x,y)\right]^2 \\
\approx & \sum \Delta x^2 I_x^2 + 2\Delta x \Delta y I_x I_y + \Delta y^2 I_y^2 \\
\approx & \sum [\Delta x \Delta y] \left[\begin{array}{cc} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{array}\right] \left[\begin{array}{c} \Delta x \\ \Delta y \end{array}\right] \\
\approx & [\Delta x \Delta y] \left(\sum \left[\begin{array}{cc} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{array}\right]\right) \left[\begin{array}{c} \Delta x \\ \Delta y \end{array}\right]
\end{aligned}$$

$$\begin{aligned}
E(x,y,\Delta x,\Delta y) \approx & [\Delta x \Delta y] \left(\sum w(x,y) \left[\begin{array}{cc} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{array}\right]\right) \left[\begin{array}{c} \Delta x \\ \Delta y \end{array}\right] \\
\approx & [\Delta x \Delta y] \underbrace{\left[\begin{array}{cc} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{array}\right]}_{\text{tenseur de structure}} \left[\begin{array}{c} \Delta x \\ \Delta y \end{array}\right]
\end{aligned}$$

Détecteurs de coins : le tenseur de structure

Propriétés

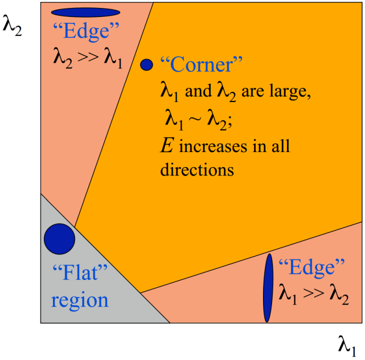
- les vecteurs propres indiquent les direction principales de variation de gradient dans le voisinage du point
- ex. : si $\lambda_2 > \lambda_1$, variation faible en v_2 et plus forte en v_1
- si coin, λ_1, λ_2 sont larges



Détecteurs de coins : le tenseur de structure

Propriétés

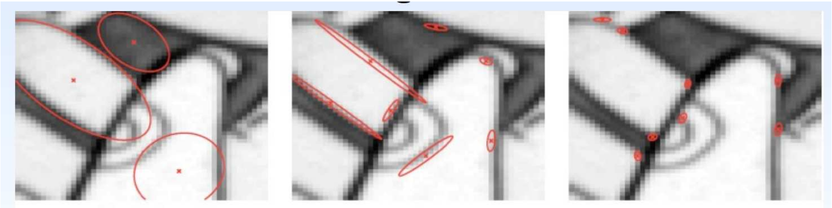
- les vecteurs propres indiquent les direction principales de variation de gradient dans le voisinage du point
- ex. : si $\lambda_2 > \lambda_1$, variation faible en v_2 et plus forte en v_1
- si coin, λ_1, λ_2 sont larges



Détecteurs de coins : le tenseur de structure

Propriétés

- les vecteurs propres indiquent les direction principales de variation de gradient dans le voisinage du point
- ex. : si $\lambda_2 > \lambda_1$, variation faible en v_2 et plus forte en v_1
- si coin, λ_1, λ_2 sont larges



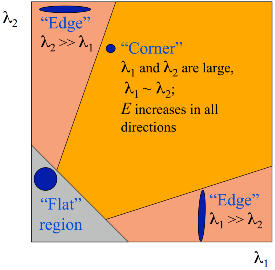
Détecteurs de coins : décision

Décision en fonction des valeurs propres du tenseur

- on peut calculer λ_1, λ_2 explicitement mais trop lourd
- méthode préférée :

$$R = det(M) - \alpha trace^2(M) = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2$$

- valeur du paramètre α en général 0.04 - 0.06
- valeurs propres intéressantes = maximum local de R



Détecteurs de coins : rappel

Étapes de l'algorithme

1. calcul des gradients $I_x = \frac{\partial}{\partial x} g(\sigma_D) \star I$, $I_y = \frac{\partial}{\partial y} g(\sigma_D) \star I$
2. calcul du tenseur de structure :

$$M = g(\sigma_I) \star \left[\begin{array}{cc} \sum I_x^2 & \sum I_x I_y \\ \sum I_x I_y & \sum I_y^2 \end{array} \right]$$

3. calcul de la fonction de réponse R :

$$R = det(M) - \alpha trace^2(M)$$

4. seuillage de R
5. suppression de valeurs non maximales de R

Détecteurs de coins : exemple

Figure – Paire initiale

Détecteurs de coins : exemple

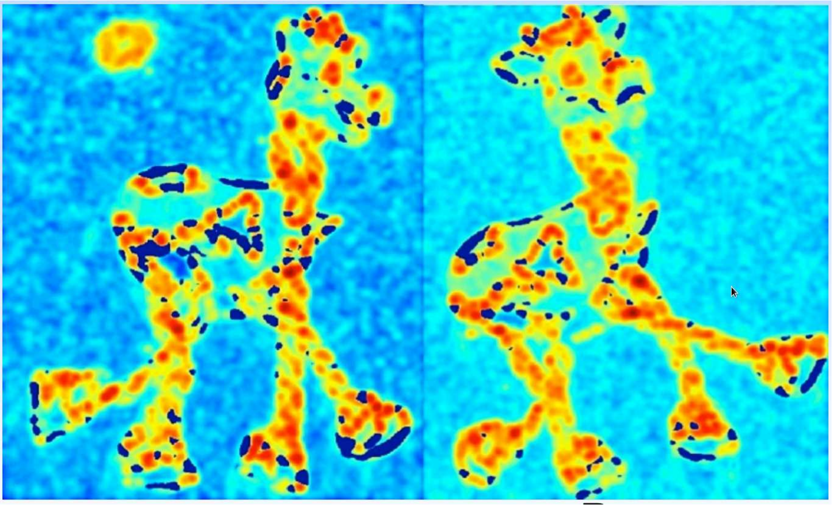


FIGURE – Fonction de réponse R

Détecteurs de coins : exemple

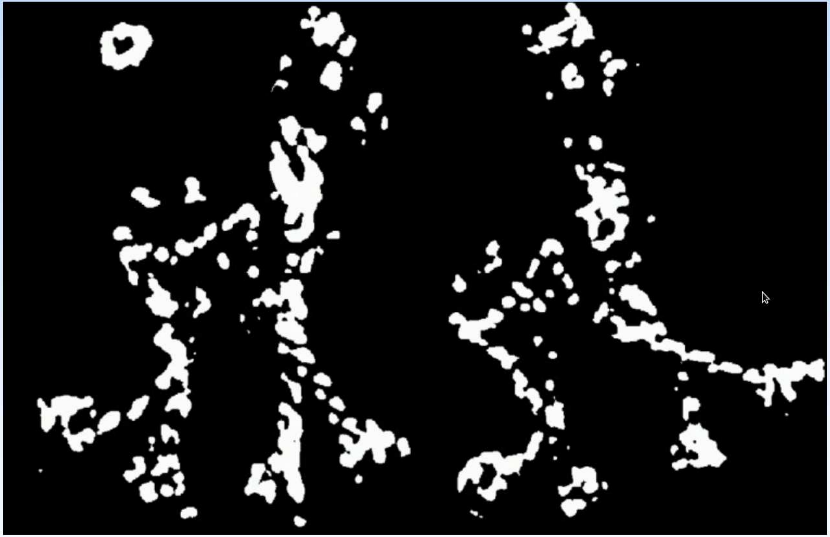


FIGURE – Seuillage de R

Détecteurs de coins : exemple

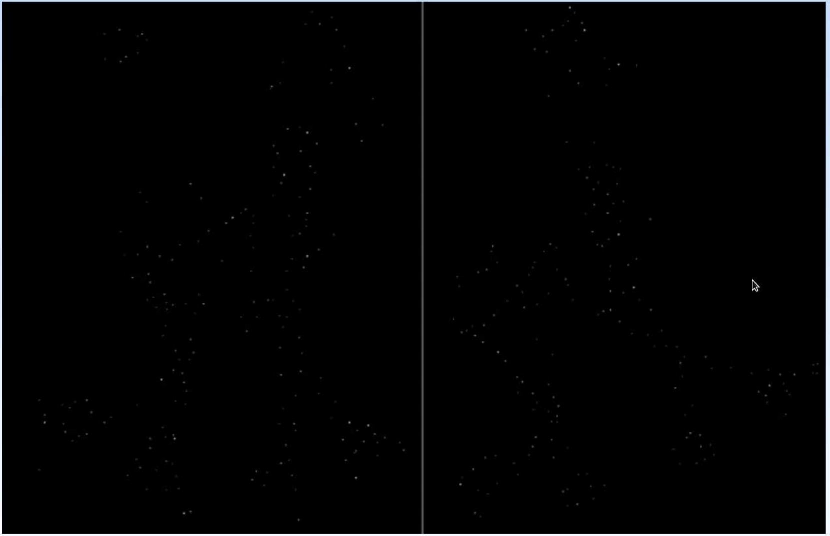


FIGURE – Suppression non maximale de R

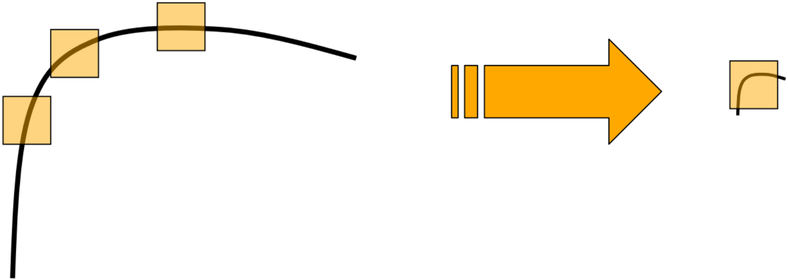
Détecteurs de coins : exemple

Figure – Résultat détection

Détecteur de Harris

Conclusions

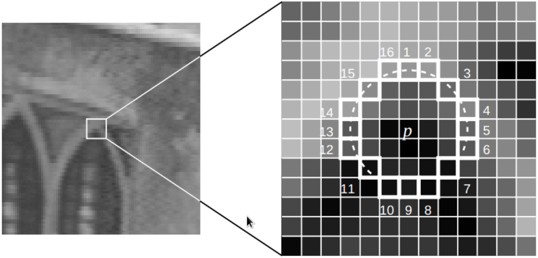
- ✓ détection invariante à la rotation
- ✓ détection invariante aux changements d'intensité
- ✗ pas robuste au changement d'échelle



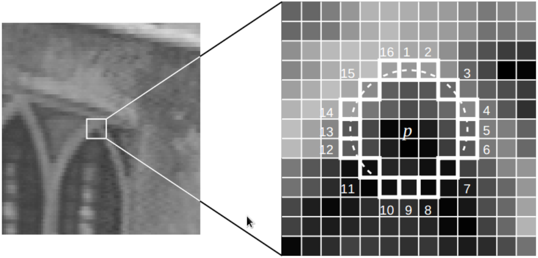
Le détecteur FAST

Features from Accelerated Segment Test

- extrêmement rapide
- pas d'opérations complexes (convolution, calcul de gradients etc.)
- peu robuste
- pas de descripteur



Le détecteur FAST - stratégie



$$S_{p\to x} = \begin{cases} d, & & I_{p\to x} & \leq I_p - t \\ s, & I_p - t < & I_{p\to x} & < I_p + t \\ b, & I_p + t \leq & I_{p\to x} & \end{cases}$$