

**Introduction générale**

Les travaux décrits dans le présent document appartiennent au domaine de la vision par ordinateur, domaine qui consiste à réaliser des algorithmes capables de traiter des images ou des séquences d'images, et de résoudre des problèmes tels que reconnaître un visage, suivre un objet, se localiser dans l'espace...

Il s'agit principalement d'actions que les êtres humains, et certains animaux, sont capables d'accomplir. Le plus souvent, on aimerait pouvoir faire accomplir à des machines des tâches qui nécessitent l'utilisation de la vision pour pouvoir être réalisées correctement.

Mais bien que l'un des buts de la vision par ordinateur, ou vision artificielle, soit de parvenir à faire réaliser à des machines ce que l'œil humain fait naturellement, les recherches en vision ne cherchent pas pour autant à copier le fonctionnement de l'œil.

En effet même si le but à atteindre peut sembler identique à ce que fait un œil naturel, il n'est pas toujours pertinent de vouloir imiter la nature sous forme d'algorithme. Ainsi, dans un autre domaine on a vu apparaître des programmes d'échec capables de rivaliser avec les meilleurs joueurs mondiaux. Pourtant leurs algorithmes ne reflètent pas la façon de jouer d'un être humain. Un bon joueur va utiliser son expérience, des connaissances acquises sur la stratégie et la tactique, un peu de lecture des coups à venir. Mais une machine n'acquiert pas d'expérience, ou du moins elle n'est pas programmée pour accomplir cette tâche, et le concept de stratégie n'a pas de sens pour elle. Par contre elle possède une mémoire parfaite, et est capable de lire de façon exhaustive des séquences de coups très longues.

Alors qu'un joueur humain va directement s'intéresser aux quelques coups intéressants de la partie à un moment donné, réfléchir aux séquences auxquels ils mènent, et faire son choix, le joueur artificiel va tester tous les coups possibles, même les plus stupides, regarder toutes les séquences, et choisir celle qui finalement est la meilleure d'après ses algorithmes. Les deux raisonnent différemment, mais tous deux jouent des coups logiques pouvant mener à la victoire.

L'homme et la machine n'ont pas les mêmes points forts en matière de vision, ni les mêmes défauts. Ainsi, l'image fournie par un « œil artificiel », comme une caméra, est parfaite : elle correspond exactement à l'environnement tel qu'il est. L'œil humain, lui, construit une image mentale de l'environnement parfois très différente de la réalité. L'œil ayant un champ de vision nette très faible, il effectue des mouvements rapides et c'est le cerveau qui se charge de construire une image complète, ce qui amène à certaines déficiences étonnantes.

Une étude de l'Université de Harvard, a ainsi démontré que des gens occupés à compter le nombre de passes entre des joueurs de basket étaient incapables de percevoir la présence d'un individu déguisé en gorille passant au milieu des joueurs et esquissant un pas de danse.

D'un autre côté, le duo œil/cerveau a des capacités d'analyse qu'il est difficile d'obtenir artificiellement. Un être humain ou un primate évolué est capable, à partir d'images d'un objet prises sous différents angles, de reconnaître cet objet vu sous un angle inédit. Comment imiter artificiellement cette faculté de retrouver la forme en trois dimensions d'un objet à partir d'images en deux dimensions, et de retrouver des éléments cachés

Le domaine du multimédia est en pleine expansion et s'appuie beaucoup sur les systèmes de traitement d'images qui permettent d'organiser l'information afin de faciliter son accès. Des moteurs de recherche multimédia permettent de naviguer dans de grandes bases de données d'images ou de catégoriser automatiquement les objets présents dans des scènes. Le développement récent des «smartphones» et autres téléphones portables munis de caméras a encore donné un élan plus important à ce secteur. Il existe par exemple des applications permettant d'acheter au meilleur prix un ticket de concert, juste en prenant une photographie de l'affiche de ce concert<sup>4</sup>. Une autre application est la création de guides touristiques tirant parti à la fois de la caméra et du récepteur GPS intégré dans certains «smartphones» pour fournir des informations sur les lieux visités<sup>5</sup>.

Finalement, on peut encore citer quelques-unes des nombreuses applications de la vision par ordinateur dans le domaine de la biométrie. Des algorithmes ont été développés de manière spécifique afin d'extraire et de comparer de nombreuses caractéristiques du corps humain : reconnaissance rétinienne, de l'iris, des empreintes digitales, de la paume de la main, du visage, ou même de la démarche. La vision par ordinateur est donc un champ d'étude extrêmement riche, diversifié et en pleine évolution.

Ce travail de fin d'études se concentre sur le domaine du suivi d'objet, que nous avons réalisé en trois parties. Dans le premier chapitre nous faisons un état de l'art des différents algorithmes qui existe dans le domaine du suivi d'objet dans une séquence vidéo, suivi d'une présentation des algorithmes d'extraction de caractéristiques de l'image SIFT, et SURF.

Dans le second chapitre, nous présentons comment utiliser les descripteurs SURF et les différentes approches théoriques de leurs généralisations au système de suivi d'objet.

Enfin, dans le dernier chapitre nous dévoilerons les résultats obtenus sur des benchmarks du workshop 2013 d'iccv.

# Chapitre 1

## Suivi d'objet dans une séquence video : Etat de l'art



### 1.1 Introduction:

Dans ce chapitre nous présentons un bref état de l'art sur le suivi d'objets dans une séquence d'images vidéo. Nous donnerons plus d'importances aux primitives qui ont été définies et largement utilisées. Nous exposerons aussi les méthodes les plus connues ayant montré leur efficacité dans le suivi.

Le problème de suivi reste ouvert en raison des facteurs externes qui composent l'image tels que l'ombre, le changement d'éclairage, de forme, l'occlusion.

Dans ce contexte, nous nous intéressons dans ce stage au descripteur SURF et son application au suivi.

### 1.2 Suivi d'objet dans une séquence vidéo :

Le suivi d'objet dans une séquence vidéo est le processus qui consiste à localiser et à déterminer le mouvement d'un objet dans une séquence d'images pris par une caméra. L'objectif du suivi est de réussir à suivre un objet au fil du temps sans perdre sa position qui doit être la plus précise possible. Il doit être capable de faire face au contrainte de changement d'échelle, d'éclairage, d'occlusion et ainsi que les changements projectifs et homographiques et affines.

Le suivi se fait sur une séquence d'images prise par une caméra fixe ou une caméra en mouvement ou en multi caméra. Il existe de nombreux algorithmes permettant de suivre des objets dans des séquences vidéo, et c'est un domaine de recherche encore ouvert. Obtenir une méthode de suivi à la fois robuste, précise et efficace est un challenge.

### 1.3 Algorithmes de suivi

Les algorithmes de suivi ont deux composantes distinctes, une phase de prédiction connaissant les paramètres ou le modèle de mouvement de l'objet, et une phase de mise en correspondance d'informations locales. La plupart des algorithmes privilient l'une par rapport à l'autre. Une combinaison des deux phases est plus ou moins nécessaire si nous voulons avoir un système robuste et complet.

#### 1.3.1 filtre de Kalman [1]

Le filtre de Kalman est un filtre basé sur un schéma de prédiction-correction qui peut être adapté au suivi d'objet. Ce filtre suppose que le déplacement de l'objet d'une frame à une autre se fait suivant un modèle connu.

Le filtre de Kalman se déroule de manière cyclique et possède deux phases distinctes.

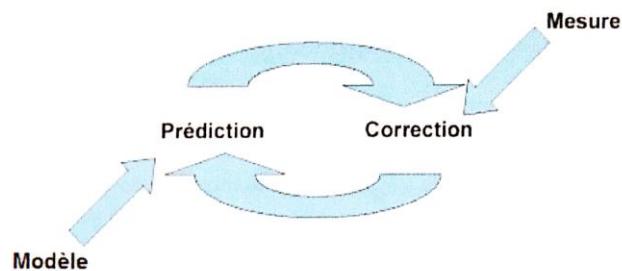


Figure 1.1 cycle du filtre de Kalman

- **Prédiction** : la nouvelle position de l'objet est prédite par une estimation Bayésienne par rapport au modèle de mouvement connu.
- **Correction** : les observations de l'instant courant sont utilisées pour corriger l'état prédit dans le but d'obtenir une estimation plus précise qui sera utilisée pour mettre à jour le modèle.

Dans l'étape de correction le filtre de Kalman utilise les points caractéristiques de l'objet qui peuvent être représentés de différentes manières, et peuvent être extraits par des différents algorithmes. Il est aussi possible de fusionner cette étape avec un autre algorithme de suivi qui privilégie la mise en correspondance

Le filtre de Kalman est efficace pour le suivi d'objets qui ont un modèle de mouvement connu et optimal pour les modèles linaires avec un bruit gaussien, ce qui les rend très efficace pour les applications de suivi du trafic routier.

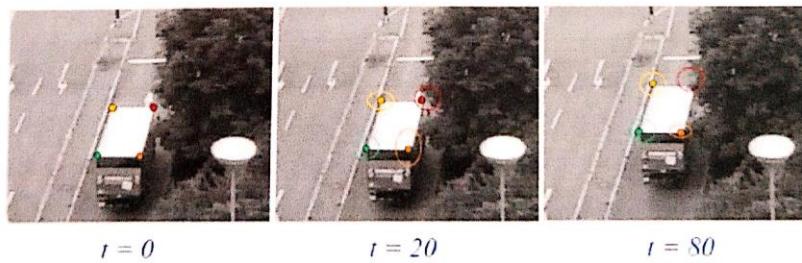


Figure 1.2 prediction des points représentant le bus

Cependant les performances de ce système se dégradent s'il y a un changement brusque de la direction de mouvement, et peuvent s'avérer inefficace pour les objets qui ne respectent pas une logique de déplacement c.à.d. qui ont une trajectoire complexe et non cyclique.



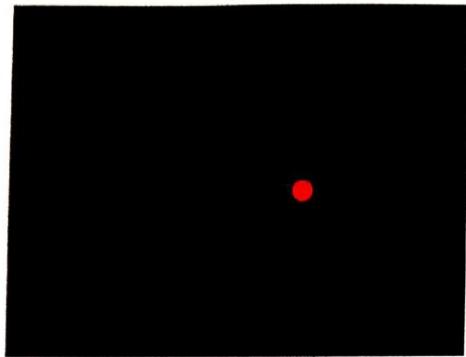


Figure 1.4. Barycentre de la région d'intérêt

Cette approche naïve est bien adaptée pour le suivi d'objets qui ont une couleur qui se détache fortement de leur environnement (sur l'image actuel). Mais à cause de la localisation qui est basé sur le centre de gravité, la localisation sera immédiatement erronée si la couleur de l'objet ne lui est pas propre et ne se démarque pas du reste de la scène.

### 1.3.3 Camshift

Le Camshift (Continuously Adaptive Mean Shift) est un algorithme de segmentation d'images couleur introduit par Gary Bradski en 1998 [Wikipédia], cette algorithme est basé sur l'algorithme Mean Shift, qui est une version très évolué de l'algorithme basé couleur.

#### Mean Shift [2]

L'une des limites de l'algorithme de suivi basé couleur c'est que généralement un objet à suivre n'a pas toujours une seule couleur. Pour remédier à cette contrainte le masque n'est pas calculé par rapport à la couleur dominante. Mais, il est construit par une retro propagation du modèle de couleur. Le masque ne sera pas une carte de point candidat mais des probabilités pour chaque point.

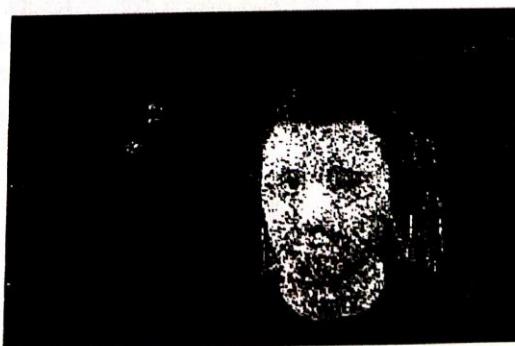


Figure 1.5: Rétro projection des probabilités

## Chapitre 1 : Etat de l'art

Le manque de fiabilité de la localisation due au centre de gravité de tous les candidats, est contourné en cherchant le maximum local des distributions de probabilité de l'objet par rapport au point initial.

Cette technique de localisation est très robuste pour trouver les maxima, l'algorithme calcule et met à jour le centre de gravité des distributions dans le voisinage de la taille de l'objet, ceci récursivement tant que le centre n'a pas convergé vers un point fixe, ou à un certain niveau de tolérance.

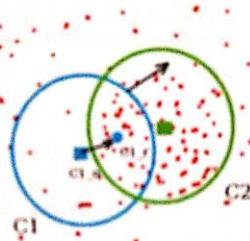


Figure 1.6: Convergence du maxima

### Camshift [3]

L'inconvénient avec Mean shift, c'est qu'une fois l'objet localisé sur la scène, l'algorithme ne prend pas en considération le changement d'échelle de l'objet, et cherchera toujours les maxima dans un voisinage fixe.

Camshift améliore Mean shift en l'adaptant au changement de taille ce qui lui permet de le rendre robuste au changement d'échelle et d'avoir des résultats plus précis. Un objet est représenté par une ellipse, ses paramètres sont déduits à partir des moments du second ordre de la distribution des probabilités de l'objet.

Camshift se déroule en deux phases, la première est la localisation qui est faite par Mean shift. La seconde est la mise à jour de l'échelle de l'objet. Cette étape change la taille de l'objet plus ou moins grande, et réapplique Mean shift, récursivement jusqu'à convergence vers la meilleure échelle.

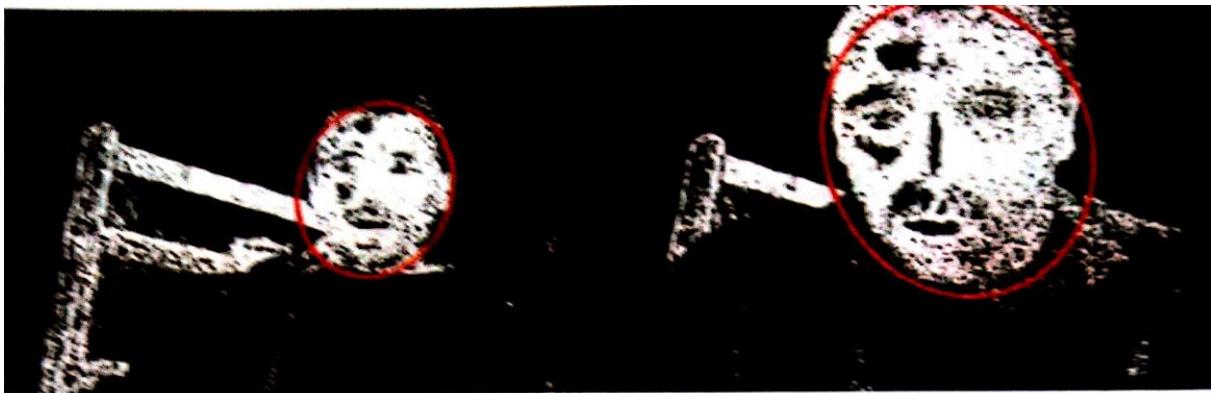


Figure 1.7 Camshift appliquée au suivi de visage

Camshift est un algorithme robuste et efficace pour le suivi d'objet. Le fait qu'il soit basé sur les couleurs lui permet d'être rapide, ce qui permet de faire un bon suivi et de faire face aux changements brusques de mouvement.

Bien que l'algorithme Camshift fournit des résultats satisfaisants, nous citerons quelques limites qui ne sont pas négligeables pour un système de suivi. En cas d'occlusion de l'objet avec une zone ou un objet qui possède une texture similaire Camshift aura tendance à converger vers toute la zone et perdra l'objet.



C'est pour cela qu'en traitement d'images, l'image est compressée et représentée par des primitives. Les primitives représentent des informations pertinentes sur l'image. Ils peuvent être sous la forme de points, de courbes continues, ou encore de régions connexes rectangulaires ou non, selon la méthode de détection utilisée.



Figure 1.9 Primitives sous la forme de point qui représenter les coins

Après la détection des primitives, le nombre d'entités à étudier est beaucoup moins important que le nombre des pixels. Pour comparer et associer des primitives, nous devons avoir un moyen pour les reconnaître et les identifier, on peut leur associer une signature sous la forme d'un descripteur. Les algorithmes qui calculent les descripteurs sont des extracteurs.

Les algorithmes de détection extraction peuvent garantir une certaine invariance à un ou plusieurs changements (éclairage, rotation, ...etc.) d'un point clé selon les combinaisons utilisées.

Dans ce qui suit nous présenterons quelques algorithmes d'extraction de primitives.

#### 1.4.1 SIFT: (Scale-Invariant feature transform)[4]

SIFT transformation des caractéristiques visuelles invariantes à l'échelle, développé par David G.Lowe. C'est un algorithme pour le calcul de similarité entre deux images, cet algorithme traduit chaque point-clé de l'image en un descripteur caractéristique, une image sera représentée par un ensemble de descripteurs SIFT qui ont la particularité d'être invariants au changement d'échelle, de rotation, de luminance et aux changements mineurs du point de vue.

La détection des points-clés passe par les étapes suivantes :

##### Espace des échelles

Une pyramide est une représentation multi échelle d'une image. Chaque niveau de la pyramide (octave) représente l'image à une certaine résolution, la base de la pyramide contient l'image originale.

SIFT représente un point par trois composantes, ses coordonnées cartésiennes ( $x, y$ ), et un facteur d'échelle  $\sigma$ .

## Chapitre 1 : Etat de l'art

Le gradient  $L$  de facteur d'échelle  $\sigma$ , est le résultat de la convolution d'une image  $I$  par un filtre gaussien  $G$  de paramètre  $\sigma$ , cette convolution va lisser l'image et supprimer les détails de rayons inférieurs à  $\sigma$ .

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$$

Une pyramide de gradients contient un nombre fixe de gradient de facteurs avec des échelles qui correspondent à une progression géométrique ( $\sigma, K\sigma, K^2\sigma\dots$ ). La première image de l'octave est obtenue en divisant l'image de la précédente octave en deux.

La détection des objets de dimension approximativement égale à  $\sigma$  se fait en utilisant l'image différences de gaussiennes DoG (différence de gaussienne).

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma)$$

La pyramide de DoG constitue l'espace des échelles utilisé par SIFT pour les futurs traitements.

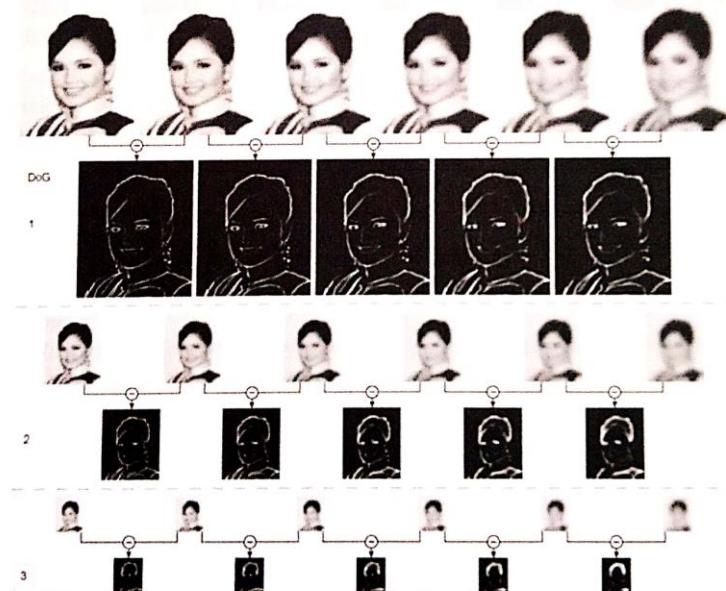


Figure 1.10: Construction de la pyramide de DoG à partir de la pyramide de gradients

### Détection des points-clés

Cette étape sert à détecter les points-clés candidat. Un point est considéré s'il constitue un extrémum local par rapport aux voisins de ses voisins directs.



*Figure 1.11: Exemple de détection d'extrem*

s clés candidats obtenue contient le procède alors a plusieurs filtrages pour améliorer les précisions

vergence de la position des points et l'obtention des coordonnées.

éliminer les points de faible contraste. Il faut également éliminer les points situés sur des arêtes "faibles". Cela peut se faire "assez" facilement.

## Chapitre 1 : Etat de l'art

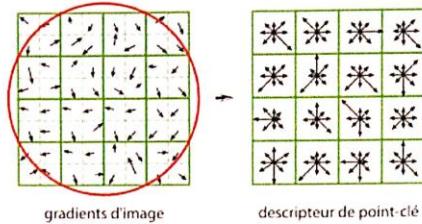


Figure 1.12: Construction d'un descripteur SIFT.

### 1.4.2 SURF: (Speeded Up Robust Features)[5]

Dans cette partie nous présenterons les aspects théoriques de l'algorithme SURF qui va être à la base de notre système de suivi d'objet. Les auteurs (**H. Bay, T. Tuytelaars et L. Van Gool**) se sont inspirés de l'algorithme SIFT pour créer un algorithme plus rapide et robuste.

#### Espace des échelles

SURF utilise tout comme SIFT une pyramide d'images pour représenter son espace des échelles, dans le même but de garantir une invariance à la mise à l'échelle pour ses descripteurs. La différence réside dans la méthode de calcul qui est l'un des points forts de SURF.

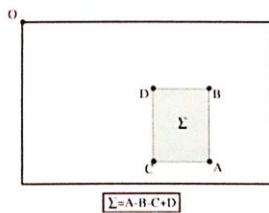


Figure 1.13 Principe de l'image intégrale.

Les images sont filtrées à plusieurs niveaux, au lieu de réappliquer le même filtre à la sortie de la convolution comme le fait SIFT, on utilise l'image intégrale de l'image originale filtre avec des filtres gaussiens progressifs.

La valeur d'une image intégrale en un point donné représente la somme de tous les pixels de l'image d'origine situés dans le rectangle formé par l'origine et le point considéré. Il suffit de trois additions pour calculer la somme des intensités des pixels de n'importe quelle région rectangulaire de l'image d'origine, quelle que soit sa taille.

## Chapitre 1 : Etat de l'art

Pour pouvoir utiliser les images intégrales, les dérives de second ordre des gaussiennes doivent être finies et discrétisées. Les auteurs de la méthode ont utilisé une approximation de type Box-Filter. Grâce aux images intégrales, le temps de calcul est indépendant de la taille du filtre.

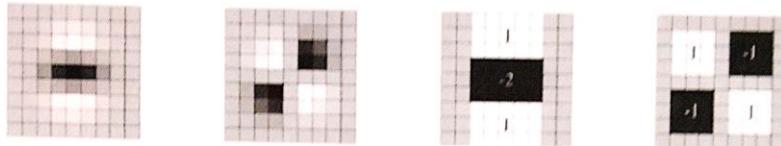


Figure 1.14: Dérivées partielles de second ordre de la gaussienne selon  $y$  et  $x, y$ . Finies à gauche et discrètes à droite

Les images des octaves supérieures sont obtenues non pas en réduisent l'image mais en agrandissant la taille du filtre qui est appliquée à l'image intégrale originale. Le filtre initial est de 9x9. Les filtres sont progressivement augmentés 9x9, 15x15, 21x21, 27x27, 39x39, 51x51, 75x75.

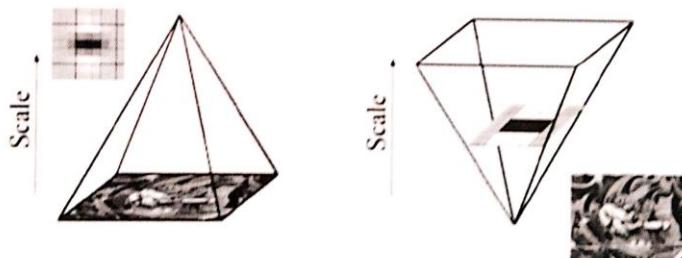


Figure 15: détections multi échelle : SIFT masque fixe, SURF image fixe

### Détection des points-clés

Les points-clés sont détectés à partir de la matrice Hessianne. Dans le contexte de détection de points, la matrice Hessianne d'un point  $X=(x, y)$  et à l'échelle  $\sigma$ , est définie comme suit :

$$\mathcal{H}(x, \sigma) = \begin{bmatrix} L_{xx}(x, \sigma) & L_{xy}(x, \sigma) \\ L_{xy}(x, \sigma) & L_{yy}(x, \sigma) \end{bmatrix},$$

Où  $L_{xx}(X, \sigma)$  qui est le résultat de la convolution de la dérivée seconde de la gaussienne selon  $x$  avec l'image au point  $x$ .

Un point  $(x, y, \sigma)$  est considéré comme point-clé si le déterminant de la matrice Hessianne en ce point atteint un maximum.

## Chapitre 1 : Etat de l'art

### Assignation d'orientation

L'orientation principale d'un point-clé  $(x, y, \sigma)$  est calculée à partir de l'image intégral en utilisant les ondelettes de Haar qui sont en fait des box-filter, ces ondelettes permettent de calculer les dérivées premières de l'image sur un voisinage carré.

SURF calcule la somme de toutes les réponses des ondelettes situées dans une fenêtre de taille  $\pi/3$  tournant autour du point-clé avec un rayon de  $6\sigma$ , l'orientation du point clé est l'orientation de la somme dominante.

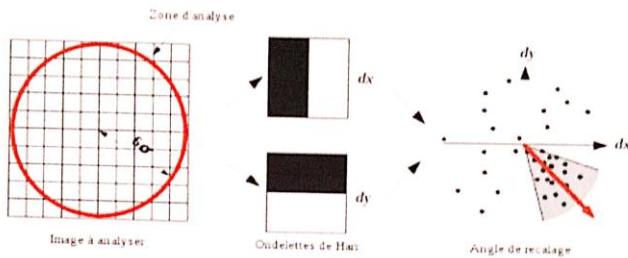


Figure 1.16: assignation d'orientation d'un point-clé SURF

### Calcul du(descripteur)

En premier lieu, une région rectangulaire centrée autour du point d'intérêt et orientée suivant la direction principale du point est définie. La taille de cette fenêtre est de  $20\sigma$ .

La fenêtre est subdivisée en  $4 \times 4$  régions de  $5 \times 5$  sous-régions. Pour chaque région, un descripteur est calculé et contient :

- la somme des réponses des ondelettes verticales.
- la somme absolue des réponses des ondelettes verticales.
- la somme des réponses des ondelettes horizontales.
- la somme absolue des réponses des ondelettes horizontales.

Les ondelettes ont une taille de  $2\sigma$

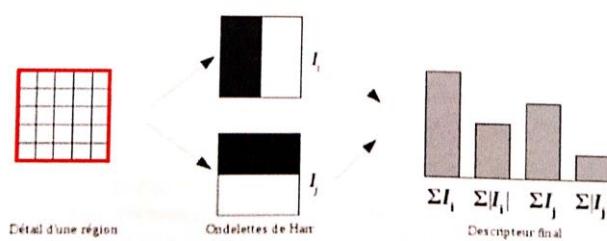
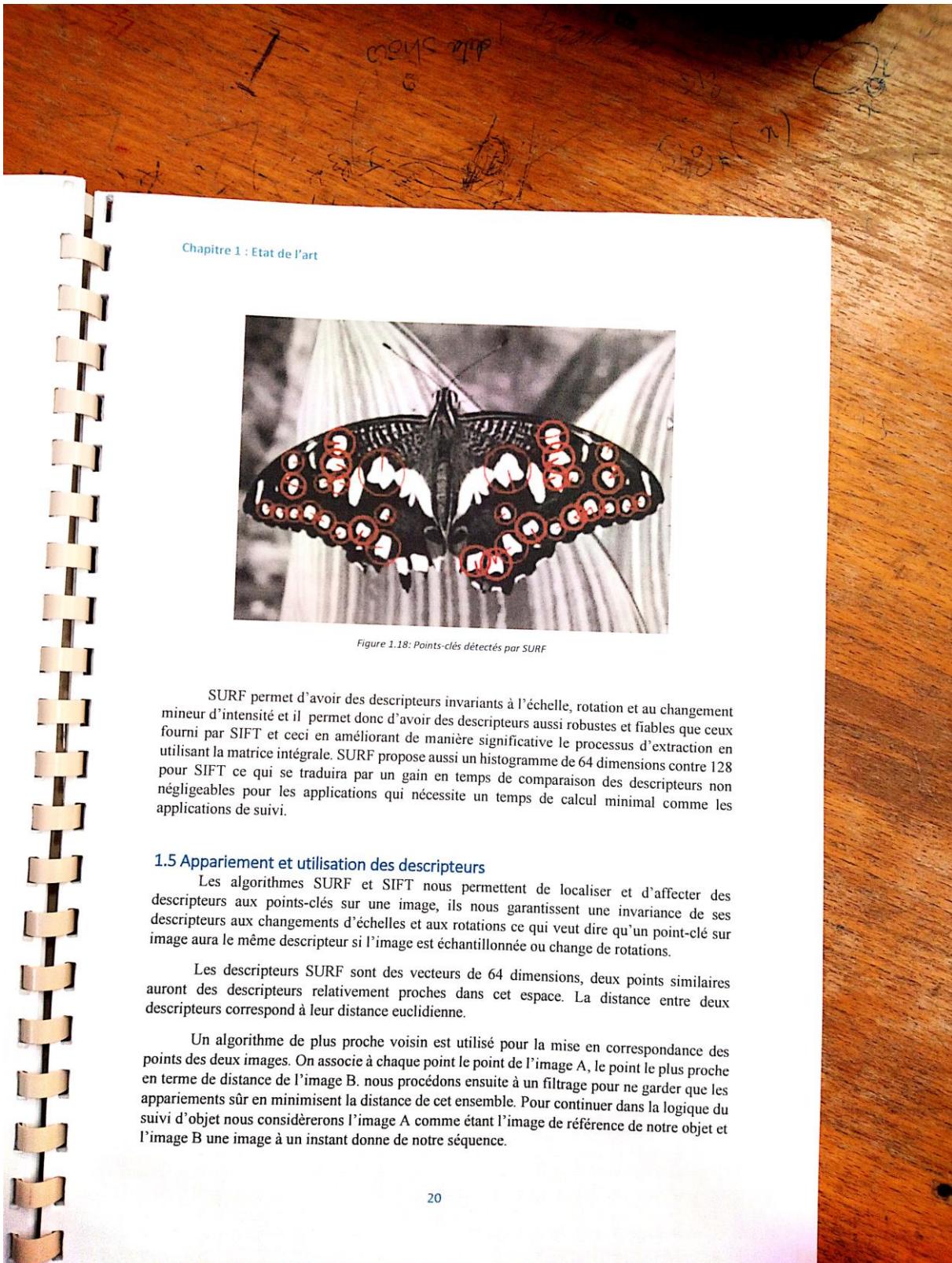


Figure 1.17: descripteur d'une zone

Le descripteur SURF du point est alors obtenu en concaténant les descripteurs de toutes les régions  $16 \times 4 = 64$  niveaux.



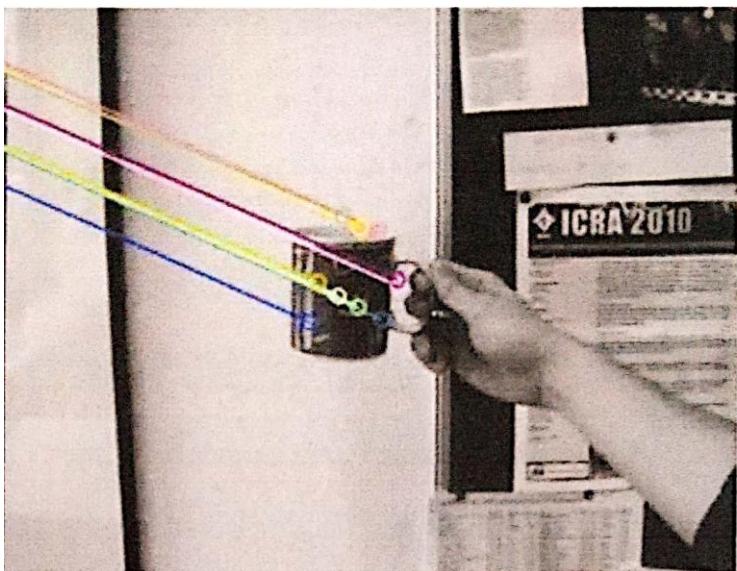


Figure 1.19: appariement des points-clés entre deux images

Ces points peuvent être utilisés pour estimer la projection affine ou la transformation linéaire non singulière dans la scène.

une transformation linéaire non singulière pour des coordonnées espace projectif.

dans ce chapitre quelques notions sur le suivi d'objets. Nous avons

## 1.1 Introduction

Nous avons vu dans le chapitre précédent qu'un système de suivi d'objet dans une séquence vidéo avait besoin d'un algorithme détection et de mise en correspondance des points-clés. Les algorithmes de suivi que nous allons développer et présenter de ce chapitre vont privilégier l'aspect de mise en correspondance des informations locales, les primitives utilisées seront des points-clés avec leurs descripteurs détectés et extraits avec l'algorithme SURF.

Le but de notre contribution est de voir quel sont les avantages et faiblesses de SURF, dans des applications de suivi d'objet dans une séquence vidéo, le suivi doit être réalisé sur des séquences avec un arrière-plan dynamique sans contrôle de luminosité.

## 1.2 Représentation de l'objet et des scènes

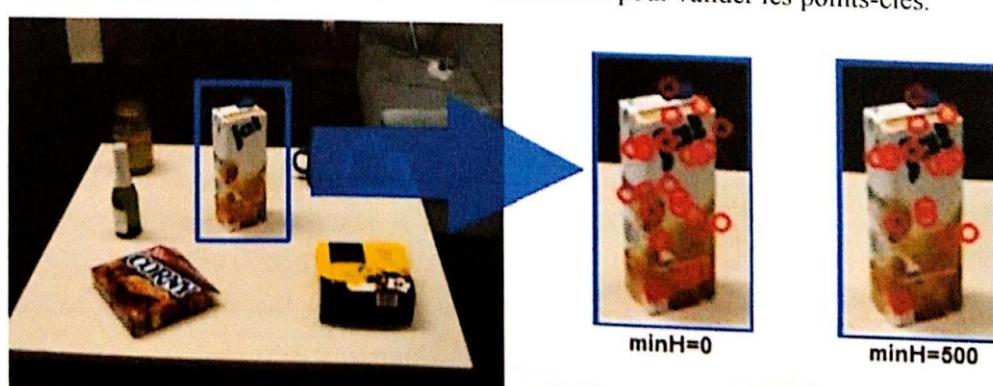
Avant toutes choses nous devons définir qu'est-ce qu'un objet à suivre, et comment représenter cette entité pour le reste de notre travail.

Un objet a une forme géométrique 3D qui peut être rigide ou déformable, sur un plan image nous ne disposent que d'une vue 2D. Nous considérons que nous n'avons pas de connaissance a priori sur la forme 3D de cet objet et la seul information dont nous disposent est une vue 2D contenue sur une image de la séquence.

Dans ce qui suit nous considérons une image comme étant un ensemble de points-clés SURF avec leurs descripteurs associés. Cette représentation sera appliquée pour la vue de l'objet à suivre ainsi que toutes les images de la séquence.

Les points-clés sont détectés à partir d'une pyramide qui constitue l'espace des échelles. Cette pyramide a deux paramètres sa hauteur et sa largeur, la hauteur et le nombre d'octaves, et la largeur est le nombre de convolution par octave. Ces paramètres seront fixes dans un premier temps à 4 octaves et 3 convolutions par octave.

Nous avons vu qu'un point-clé est accepté si la réponse du déterminant de sa matrice Hessienne constitue un maximum local. Pour avoir un ensemble de points plus stable et plus fiable nous ajoutons un seuil de réponse minimal  $\text{minH}$  pour valider les points-clés.



## 2.3 Détection de l'objet dans la scène

### 2.3.1 Appariement des points-clés

Dans cette étape nous allons faire correspondre à chaque point-clé de l'objet un point-clé de la scène. Nous savons qu'un point-clé de l'objet lui est associé le point-clé le plus proche de la scène (distance entre les descripteurs), mais la recherche des plus proches voisins est une opération couteuse si elle est appliquée de manière exhaustive. Pour remédier à cela nous utiliserons l'algorithme FLANN (Fast Approximate Nearest Neighbor) [6]. FLANN contient une collection d'algorithmes d'optimisation pour la recherche des plus proches voisins et va selon la taille de nos données automatiquement choisir le meilleur algorithme avec ses paramètres optimums.

Pour des descripteurs SURF, FLANN utilisera les arbres KD aléatoires [7] pour permettre de structurer l'espace de recherche afin d'accélérer la comparaison d'un élément avec les autres. Les performances de recherche d'un arbre KD se rapprochent de celles d'une recherche linéaire lorsque le nombre de dimensions est grand. Un arbre KD ne serait donc pas performant pour SURF puisqu'il y a 64 dimensions. Le principe d'arbres KD aléatoires, dans ce cas sera d'utiliser N arbres KD en parallèle, chacun utilisant uniquement 5 dimensions tirées aléatoirement.

Après la mise en correspondance on souhaite éliminer le maximum de fausses mises en correspondance, ce qui est primordial pour le rester de notre travail puisque les algorithmes de suivi vont dépendre entièrement de la précision de ces mises en correspondance.

Nous procéderons d'abord par filtrage par unicité qui permet d'éliminer les mises en correspondances ambiguës, pour cela nous associons à chaque point  $\mathbf{P}$  ses deux plus proches voisins  $\mathbf{v1}$  et  $\mathbf{v2}$ , une mise en correspondance  $(\mathbf{P}, \mathbf{v1})$  est acceptée si

$$\text{Distance}(\mathbf{P}, \mathbf{v1}) < \text{ratio} * \text{Distance}(\mathbf{P}, \mathbf{v2}).$$

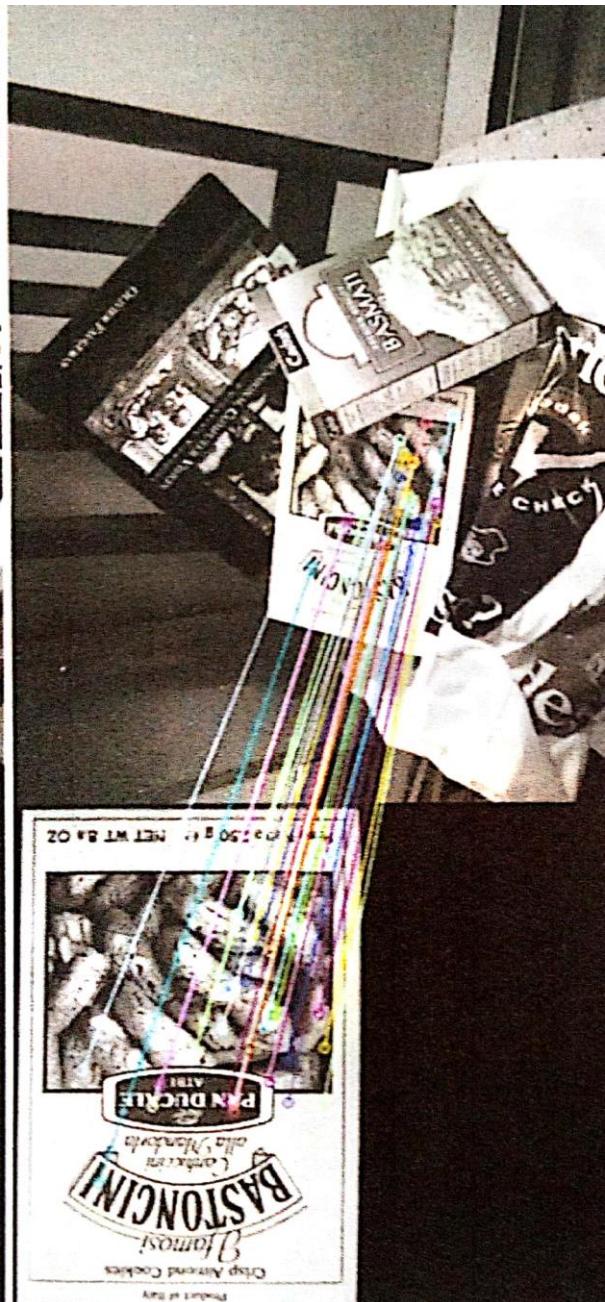
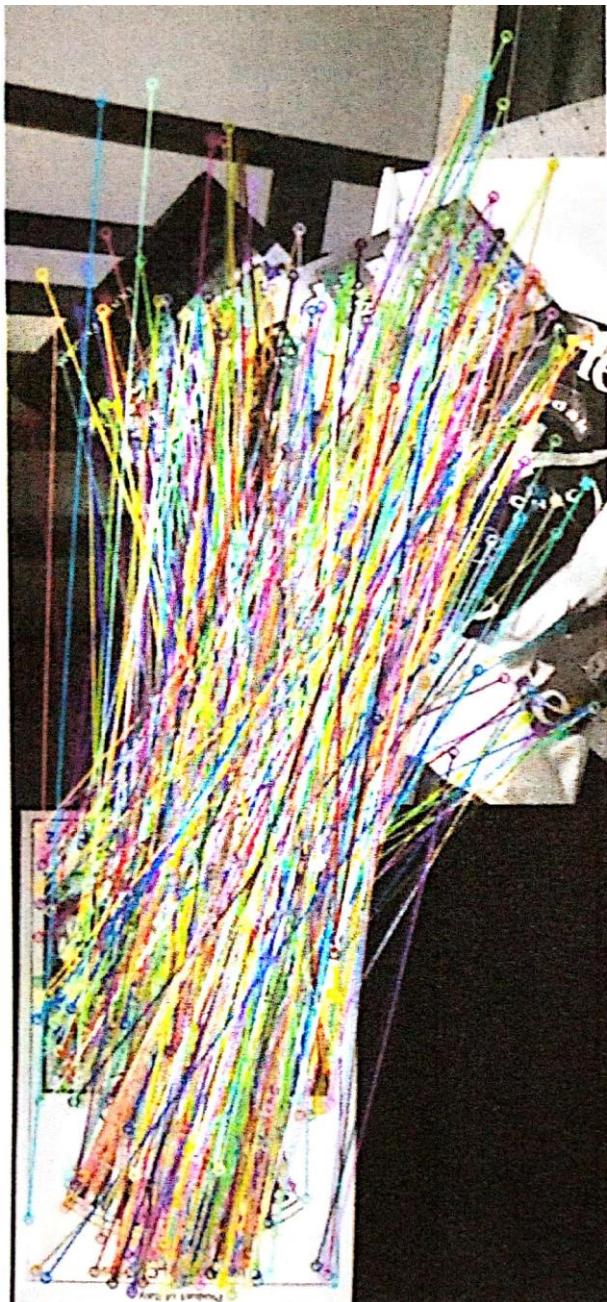
Avec un  $\text{ratio} < 1$ , ce qui va nous permettre d'éliminer les voisins trop proches.

Ensuite, un second filtre va réduire l'impact des faux positifs les auteurs [5] proposent d'ajouter des contraintes géométriques sur les points-clés, l'espace des angles est découpé en tranches de 20 degrés et les facteurs d'échelles en tranches de 1.5. En pratique nous n'utiliserons que la contrainte d'angle, la contrainte d'échelle de facteur ne sera pas utilisée pour avoir plus de tolérance par rapport au changement de taille de l'objet dans une séquence.

Nous ajoutons une autre contrainte pour valider les mises en correspondance celle-ci par rapport à leur distance. Nous remarquons que plus la distance grandit plus les appariements deviennent erronés, pour cela nous ajoutons un seuil de distance minimal.

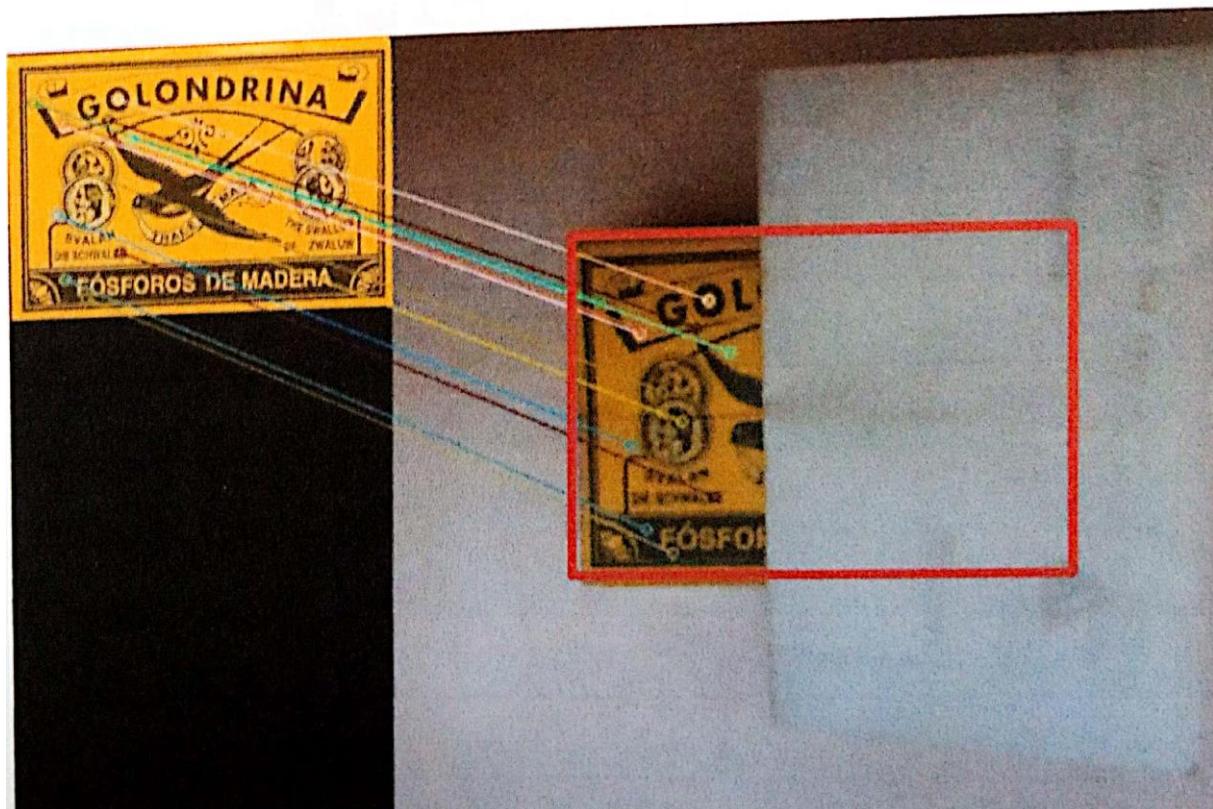
Dans la figure 2.2 ci-dessous nous pouvons voir les appariements avant et après filtrage, voici les résultats obtenus progressivement au cours des différents filtrages :

- 803 appariements sans filtrage.
- 46 appariements après filtrage par unicité (ratio=0.7).
- 41 appariements après filtrage par contrainte géométrique.
- 29 appariements après filtrage par seuil de distance minimal (seuil=0.2).



ce rectangle utilise le barycentre de l'rectangle. On calcule la position calculée précédemment au rectangle en appliquant une mise à l'échelle rapport à son point de référence. Nous projetons ensuite ce rectangle et nous interposons son point de référence avec le barycentre des points de la scène.

Cette méthode est efficace pour localiser un objet sur une scène qui est rigide, et peut aussi se montrer robuste face aux occultations partielles.



Cette méthode considère que les mises en correspondance sont valides et ne contiennent pas de faux appariement ce qui n'est pas toujours le cas, ce qui va se répercuter directement sur la fiabilité du résultat.

Mais cette technique de localisation à une limite non négligeable elle ne permet pas de prendre en considération les changements de point de perspective de la vue de l'objet, ce qui est une information très importante pour un système de suivi d'objet qui doit généralement faire face à ce type de changement.

### 2.3.2.2 Transformation affine

Une transformation affine est une transformation non rigide qui conserve le parallélisme, cette transformation est représentée par une matrice qui définit la relation entre les deux images.

La projection affine d'un point 2D peut être représentée par une matrice  $M$  2x3.

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix}_{2 \times 2}, B = \begin{bmatrix} b_{00} \\ b_{10} \end{bmatrix}_{2 \times 1}$$
$$M = [A \ B] = \begin{bmatrix} a_{00} & a_{01} & b_{00} \\ a_{10} & a_{11} & b_{10} \end{bmatrix}_{2 \times 3}$$

La projection  $T$  d'un point  $[x, y]$  est obtenue par

$$T = A \cdot \begin{bmatrix} x \\ y \end{bmatrix} + B \text{ ou } T = M \cdot [x, y, 1]^T$$
$$T = \begin{bmatrix} a_{00}x + a_{01}y + b_{00} \\ a_{10}x + a_{11}y + b_{10} \end{bmatrix}$$

Pour localiser notre objet sur la scène nous devons calculer la matrice correspondant à la projection affine des points de l'objet sur la scène.

Une fois la matrice calculée nous projetons les quatre coins de l'objet sur la scène ainsi l'objet sera localisé par le rectangle inscrit par ses quatre points.

Une projection affine est une projection rigide à laquelle s'ajoutent le changement d'échelle anisotrope et le cisaillement, un objet rigide ne subit que des changements rigides (changement d'échelle isotrope et rotation) par conséquent une transformation affine va hériter des limites de la projection rigide.

### 2.3.2.3 Transformation Homographique

Une transformation Homographique est une transformation perspective qui conserve l'alignement des points des droites parallèles. Cette projection prend en considération les changements de perspective de la vue de l'objet. Il s'agit d'une transformation linéaire entre deux plans projectifs. Ce type de transformation est utilisé pour modéliser le passage de l'espace tridimensionnel à l'espace bidimensionnel.

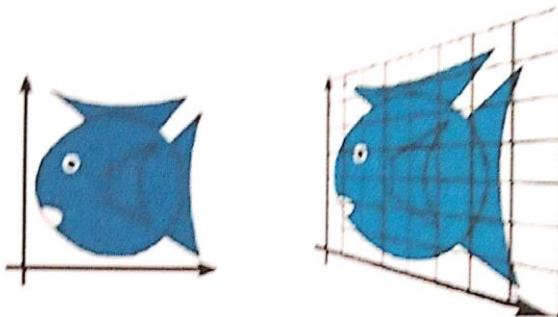


Figure 2.4 projection homographique

La projection Homographique peut être représentée par une matrice  $H$   $9 \times 9$ , pour avoir la projection d'un point  $(X_D, Y_D)$  nous passons au coordonnées homogenes

$$\begin{pmatrix} X_2 \\ Y_2 \\ 1 \end{pmatrix} = \begin{bmatrix} h_1 & h_2 & h_3 \\ h_4 & h_5 & h_6 \\ h_7 & h_8 & h_9 \end{bmatrix} \begin{pmatrix} X_1 \\ Y_1 \\ 1 \end{pmatrix}$$

Pour calculer une matrice de projection homographique nous avons besoin d'au moins 4 points, nous pouvons utiliser l'algorithme DLT (Direct Linear Transform)[8] qui consiste à déterminer les inconnues de la matrice en utilisant un système les systèmes d'équations obtenues par la projection de ces points.

Pour estimer la matrice homographique correspondante à nos mises en correspondance nous utilisons une méthode de vote qui vise à réaliser un certain nombre d'estimations de la matrice d'homographie (retournant chacune un vote), ceci va permettre de récupérer la projection optimale et d'éliminer l'impact des fausses correspondances. La méthode de vote utilisé est basé sur l'algorithme RANSAC [9] qui est une méthode itérative robuste. RANSAC nous permettre de récupérer la projection optimale et aussi de détecter les pairs qui appartiennent ou non à cette projection.

Après le calculer de la matrice la localisation se fait de la même que précédemment, on projet les coins de l'objet pour récupérer la nouvelle position



Figure 2.5 localisations avec les trois types de projection : rouge (rigide), bleu (affine), vert (homographique)

La projection homographique et la projection la plus réaliste, elle va nous permettre de localiser l'objet avec grand facteur de certitude. Si RANSAC échoue nous utiliserons la projection affine ou rigide pour faire une localisation approximative de l'objet. La projection rigide étant un cas particulier de projection affine qui elle aussi est un cas de projection homographique, l'homographie sera capable de faire face aux occultations partielles de l'objet et le localiser

#### 2.4 Prétraitement

Nous venons de proposer un moyen de localisation d'un objet sur une scène avant de la généraliser au suivi dans une séquence vidéo nous devons préparer nos images pour améliorer nos performances.

En général, les objets à suivre ont une basse résolution sur l'image, le problème est que face aux convolutions gaussiennes et aux ré-échantillonnages consécutifs qu'applique SURF à l'image les détails caractérisant l'objet vont rapidement disparaître, ce qui va considérablement influer sur le nombre de points-clés obtenus.

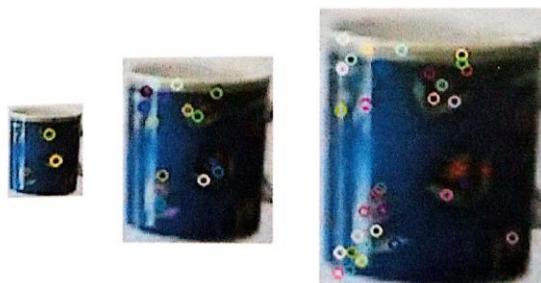


Figure 2.6 points-clés obtenue à différents agrandissements

On peut voir sur la figure ci-dessus le nombre de points-clés obtenus avec un seuil de min Hessienne=400 pour un objet de taille 48x59.

- Image x1 =2 points-clés
- Image x2 =16 points-clés
- Image x3 =38 points-clés

Nous remarquerons que le nombre de points-clés obtenus va converger à partir d'un certain niveau. Il n'est pas nécessaire d'avoir un nombre maximal de points nous nous contenterons d'un certain nombre. Nous fixerons une limite d'agrandissements pour ne pas avoir beaucoup de calculer.

## 2.5 Application au suivi d'objet premier approche

La première approche que nous étudierons va faire une prédiction approximative de l'objet sur la scène. Puis la localisation se fera à partir d'une vue de l'objet en calculant sa projection.

La détection dans les séquences vidéo se fera par un module de prédictions-localisations auquel nous ajoutons un module de mise à jour pour prendre en charge les changements de l'objet. Cette approche est étudiée pour des objets rigides.

La position d'un objet est représentée par son centre. Après chaque localisation nous calculerons le vecteur de déplacement de ce point par rapport à la position de l'image précédente si elle exister.

### 2.5.1 Prédiction et localisation de l'objet

La localisation de l'objet sur la scène se fait en calculant les projections comme vue précédemment. Nous utiliserons la dernière position valide de l'objet pour définir autour une région dans laquelle nous chercherons à localiser l'objet. L'objet a de fortes chances d'être localisé dans cette région parce que le mouvement de l'objet est supposé relativement faible entre deux images consécutives.

Cette limitation de zone de recherche va nous permettre de gagner en temps de calcul et va surtout diminuer la probabilité de fausse mise en corrépondance des points.

Si l'objet disparaît de la scène ou si nous n'arrivons pas à le localiser, après un certain nombre d'images la zone prédite n'est plus valide et l'objet peu se trouver plus loin sur la scène. Pour cela la limitation de zone est désactivée tant que l'objet n'a pas été relocalisé.

La région est définie par rapport à la dernière localisation de l'objet qui est augmenté par un  $\Delta x$  et  $\Delta y$ . Si nous supposons que l'objet n'effectue pas de changement de mouvement brusque nous pouvons déplacer cette zone avec le vecteur de mouvement précédent.

### 2.5.2 MAJ

Au cours d'une séquence vidéo un objet ne subit pas que des changements de perspective, mais il subit également des changements d'intensité lumineuse, ces changements peuvent faire apparaître ou estomper des détails de l'objet à suivre.

Au cours d'une séquence l'objet peut même changer complètement et ne plus correspondre à l'objet initial. En pratique cela se confirme l'objet est perdu si les changements sont importants. Les points-clés détectés après les changements ne correspondent plus aux points-clés initiaux de l'objet.

Pour cela nous mettons à jour l'objet que nous cherchons à localiser en prenant en considération les changements appliqués.

### 2.5.2.1 MAJ locale

Cette méthode consiste à garder la vue initiale de l'objet avec ses points-clés en mettant à jour uniquement ses descripteurs et les paramètres des points-clés. Comme suit

Une fois que l'objet est détecté nous remplaçons les descripteurs de l'objet par leurs correspondants sur la scène, l'angle et le facteur d'échelle des points-clés sont également remplacés.

Cette mise à jour continue permet de prendre en considération les changements progressifs d'intensité lumineuse.

Quand l'objet subit des grands changements les points-clés ne seront plus détectés aux mêmes positions cette méthode va continuer à remplacer les descripteurs et peut converger vers des descripteurs qui ne correspondent pas aux points-clés. Un autre inconvénient est qu'à chaque mise à jour nous ne remplaçons qu'un sous ensemble de points-clés de l'objet, nous aurons alors des descripteurs calculés sur différentes images ce qui n'est pas cohérent.

### 2.5.2.2 MAJ globale

Pour être dans une logique de suivi plus cohérente nous devons mettre à jour le changement de perspective de la vue de l'objet.

L'objet est mis à jour juste après sa localisation sur la scène, on utilise les résultats de la projection homographique pour définir le nouvel objet.

Cette mise à jour va permettre d'adapter l'objet aux variations progressives de changements de perspective et de luminosité.

Cependant, nous avons vu que le module de localisation peut localiser un objet même si ce dernier est partiellement recouvert par un autre objet *figure 22*, en se basant sur la projection l'objet sera mal mis à jour et nous l'égarerons comme on peut le voir sur la figure ci-dessous.



Figure 2.7 : échec de la MAJ de l'objet en se basant sur la projection

Pour ne pas faire de fausse mise à jour nous pouvons diviser notre objet en  $N$  parties, la mise à jour est effectuée seulement si ces parties sont localisées et en respectant la même configuration géométrique. Mais en pratique cela ne garantit pas un meilleur résultat cela dépend de la vitesse à laquelle l'objet est recouvert, et une partie peut ne pas contenir de point-clé.

### 2.5.3 Algorithmme de la première approche

#### Debut

- . Ouvrir la séquence vidéo
- . Sélectionner l'objet à suivre
- . Extraire les points clés surf de l'objet
- . Calculer les descripteurs surf des points clés
- . Pour chaque frame de la séquence
  - . Faire
    - . Si (dernière position connue)
    - . . Alors définir une sous-région de recherche sur la frame par rapport à la position précédente
    - . . Sinon la région de recherche est toute la frame
    - . . FinSi
    - . . Extraire les points clés surf de la région de recherche
    - . . Calculer les descripteurs surf des points clés
    - . . Mise en correspondance des points clés
    - . . Filtrer la mise en correspondance
    - . . Si (nombre de mise en correspondance < 4 )
    - . . . Alors "impossible de localiser l'objet"
    - . . . Sinon
      - . . . Calculer de la projection homographique
      - . . . Si la projection est correctement calculée
      - . . . . Alors récupérer la nouvelle position de l'objet
      - . . . . Sinon
        - . . . . Calculer la projection affine
        - . . . . Si la projection est correctement calculée

```
    . . . Alors    récupérer la nouvelle position de l'objet
    . . . Sinon    récupérer la nouvelle position en utilisant le barycentre
    . . . FinSi
    . . . FinSi

    . . . Si (mise à jour locale)
    . . . Alors
    . . . . Remplacer les descripteurs de l'objet par leur correspondant sur la frame
    . . . . Remplacer les paramètres des points clés de l'objet par leur correspondant sur la frame
    . . . FinSi

    . . . Si (mise à jour globale)
    . . . Alors
    . . . . Si objet est localisé avec la projection homographique
    . . . . Alors
    . . . . . Remplacer l'objet par celui de la frame
    . . . . . Calculer les points clés et descripteurs du nouvel objet
    . . . . Sinon    "impossible à mettre à jour"
    . . . . FinSi
    . . . FinSi

    . . FinSi
. Fait
Fin.
```

## 2.6 Application au suivi d'objet second approche

La seconde approche explore est une approximation du flux optique de l'arrière-plan avec lequel on déduira le mouvement de l'objet à suivre.

Le flux optique sera utilisé pour estimer le mouvement de la caméra face à la scène, en utilisant la projection homographique pour calculer avec l'algorithme RANSAC. Nous avons vu que cet algorithme calcule la projection homographique en utilisant un système de vote, si nous avons plusieurs modèles de projections (projection scène + projection objet) RANSAC va nous permettre de récupérer la projection la plus dominante ainsi que les paires de points qui n'appartiennent pas à cette projection.

Nous calculons la projection homographique entre la scène précédente et la scène actuelle à l'aide de RANSAC.

La seule information que nous avons sur l'objet à suivre est sa position qui est définie par une zone, nous parcourant les mises en correspondances des points-clés se trouvant sur cette zone en gardant que les paires qui n'appartiennent pas à la dernière projection calculée. Ceci va nous permettre d'éliminer les points-clés se trouvant sur l'arrière-plan. Nous utilisons la méthode de localisation précédente pour calculer la nouvelle position de l'objet.

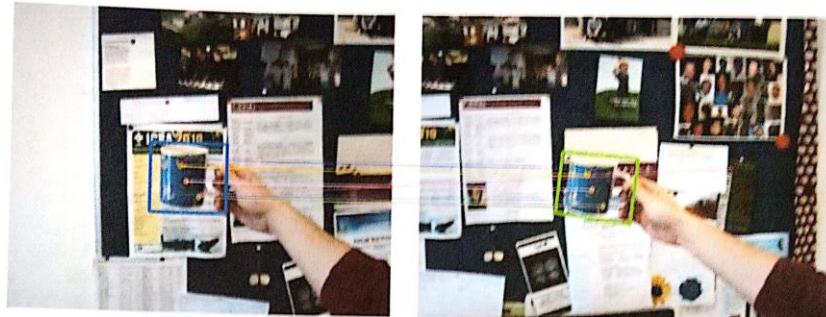


Figure 2.8: localisation en utilisant le flux optique de l'arrière-plan

Cette approche peut être robuste face aux changements brusques d'intensités lumineuses de l'objet.

Si nous n'avons pas de points-clés sur l'objet cela peut être dû s'il n'y a pas un mouvement réel de l'objet ou à cause d'une mauvaise détection. Pour gérer cela nous avons deux propositions. Soit on suppose que l'objet se déplace avec la scène et nous le projetons avec, ou soit nous laissons la position tel quel en espérant qu'elle sera corrigé dans les prochains calculer.

Cette approche suppose que sur la scène nous n'avons que l'objet à suivre qui est en mouvement réel et que l'arrière-plan est riche en détail pour pouvoir estimer sa projection. Un autre inconvénient, si l'objet disparaît et réapparaît dans une autre position on ne pourra pas le localiser.

### 2.6.1 Algorithme de la seconde approche

#### Début

- . . Ouvrir la séquence vidéo
- . . Sélectionner l'objet à suivre sur la premier frame
- . . Pour chaque frame de la séquence
- . . Faire
  - . . . Extraire les points clés SURF de la frame et de la frame précédente
  - . . . Calculer les descripteur SURF associé aux points clés
  - . . . Mise en correspondance des points clés
  - . . . Filtrer les mises en correspondances
  - . . . Si(nombre mise en correspondances <4 )
  - . . . Alors "impossible de localiser l'objet"
  - . . . Sinon
    - . . . . Calculer la projection homographique avec RANSAC
    - . . . . Si la projection est correctement calculée
      - . . . . Alors

Chapitre 2 : Extraction et suivi des régions d'intérêt dans une séquence vidéo

```
    . . . . Récupérer les appariements qui n'appartiennent pas à la projection
    . . . . Garder seulement les appariements de l'objet sur la frame précédente
    . . . . Si(nombre de points clés >4 )
    . . . . Alors
    . . . .   . Calculer la position de l'objet en utilisent la projection homographique
    . . . . Sinon
    . . . .   # deux cas possibles
    . . . .   1. Garder la position telle quelle, sans rien faire
    . . . .   2. Calculer la nouvelle position de l'objet en utilisant la dernière projection
    . . . . Finsi
    . . . . Sinon "impossible de localiser l'objet"
    . . Finsi
. Fait
Fin
```

## 2.7 Conclusion

Nous avons pu voir dans ce chapitre comment mettre en correspondance deux images en utilisant les descripteurs SURF. Nous avons ensuite proposé deux approches pour le suivi d'objets dans une séquence idéo.

## Chapitre 3

### *Suivi d'objet dans une séquence vidéo : Validation des algorithmes proposés sur des bases de données*

Thierry Boult

Université de Guelph  
Centre de recherche en informatique et en génierage  
Guelph, Ontario, N1G 2W1, Canada

http://www.cs.guelph.ca/~boult/research.html  
E-mail: thierry.boult@cs.guelph.ca

http://www.cs.guelph.ca/~boult/research.html  
E-mail: thierry.boult@cs.guelph.ca

### 3.1 Introduction

Dans ce chapitre nous allons faire une évaluation des approches que nous avons proposées dans le chapitre précédent sur des bases de données, nous évaluerons chaque séquence par les deux approches proposer et nous discuterons des résultats obtenus.

### 3.2 Environnement de travaille

#### 3.2.1 Environnement Matérielle

Les évaluations ont été réalisées sur un micro-ordinateur, caractérisé par les éléments suivants :

- Processeur : Intel (R) core (TM) i5 CPU M2410M @ 2.30GHz
- Mémoire installée(RAM) 4,00 Go
- système d'exploitation 64 bits (Windows 7)
- indices de performance Windows 5,9

#### 3.2.2 Environnement logiciel



Les algorithmes ont été développés avec le langage C++ et la bibliothèque référence dans la vision par ordinateur OpenCv qui la particularité d'être open source, ce qui lui permet d'être constamment mis à jour par une grande communauté de spécialistes et de chercheurs.

Grâce à cette communauté active nous avons une garantie d'avoir des algorithmes optimaux et fiables.

### 3.3 Séquence d'évaluation

Les séquences que nous utiliserons pour notre évaluation sont celles utilisées pour le workshop iccv (The Visual Object Tracking VOT2013 Challenge). La base de données contient différentes séquences prises dans des conditions différentes. Chacune des séquences contient des changements spécifiques (changement d'échelle, luminosité ...etc.) sur lesquels les systèmes seront évalués.

### Chapitre 3 : Validation des algorithmes proposés sur des bases de données

Name	number of frames	Description	Notes
bicycle [8]	271	bike, occlusion, scale change	
bolt [10]	350	body, articulation, scale change	
car	374	car	Based on PETS 2001 challenge data
cup [3]	303	cluttered background	sequence was shortened for the challenge
david [6]	770	head, illumination and scale change	
diving [4]	231	body, articulated, rotation	
face [1]	415	head, occlusion	sequence was shortened for the challenge
gymnastics [7]	207	body, articulated, scale change	
hand [7]	244	hand, articulated	
iceskater [9]	500	body, articulated	sequence was shortened for the challenge
juice [3]	404	box	
jump [2]	228	bike, scale change	
singer [5]	351	body, illumination and scale change	
sunshade	172	head, illumination change	
torus [7]	264	interesting geometry	
woman [1]	597	body, occlusion, scale change	

Figure 3.1 description de la base de données [réf]

#### 3.4 Evaluations des algorithmes

Notre évaluation consiste à voir si la position prédictive correspond à la position réelle sur la scène. Dans ce qui suit, nous appliquerons pour chaque séquence les deux approches avec leurs variantes, puis nous comparerons leurs résultats et essaierons de les expliquer.

Les différents algorithmes sont :

- Algorithme 1.1 : approche 1 sans mise à jour de l'objet.
- Algorithme 1.2 : approche 1 avec mise à jour locale de l'objet.
- Algorithme 1.3 : approche 1 avec mise à jour globale de l'objet.
- Algorithme 2.1 : approche 2 sans projection de l'objet avec la scène.
- Algorithme 2.2 : approche 2 avec projection de l'objet avec la scène.

Les algorithmes sont présentés dans le chapitre 2 aux pages 32 et 34.

##### 3.4.1 Séquence Cup.

Objectif : suivre une tasse.

Difficulté : l'objet passe par un arrière-plan encombré et contient des couleurs similaires.



Figure 3.2 : Objet à suivre dans la séquence Cup.

**Résultats :**

L'algorithme 1.3 réussit à suivre l'objet sur toute la séquence en faisant une mise à jour entre chaque frame. Les deux autres algorithmes de la première approche ne réussissent pas à localiser l'objet quand il se confond avec l'arrière-plan mais l'objet est relocalisé dès qu'il est visible.

La seconde approche perd l'objet dès le début de la séquence, cela est dû à un mouvement lent de l'objet que l'algorithme ne parvient pas à les différencier avec celui de la scène.

**3.4.2 Séquence Juice**

**Objectif :** suivre la boîte de jus.

**Difficulté :** mouvement de caméra pas de mouvement réel de l'objet plus changement d'échelle.



Figure 3.3 : Objet à suivre dans la séquence Juice.

**Résultats :**

Les algorithmes 1.3 et 2.2 réussissent à suivre l'objet sur toute la séquence sans le perdre.

Les algorithmes 1.1 et 1.2 effectue le suivi avec quelque difficulté au moment du changement d'échelle.

Pas de suivi pour l'algorithme 2.1 parce qu'il y a pas de mouvement réel de l'objet.

#### 3.4.3 Séquence bycicle

**Objectif :** suivre la personne sur bicyclette

**Difficulté :** changement d'échelle plus occultation de l'objet avec la scène.



Figure 3.4 Objets à suivre dans la séquence bycycle

#### Résultats :

Les algorithmes 1.3, 2.1 et 2.2 réalise le suivi mais perd l'objet à la fin de la séquence.

Les deux autres algorithmes de la première approche sont inefficaces dès le début de la séquence, l'objet subi des changements assez importants.

#### 3.4.4 Séquence face

**Objectif :** suivre le visage de la personne.

**Difficulté :** occlusion de l'objet par un objet en mouvement

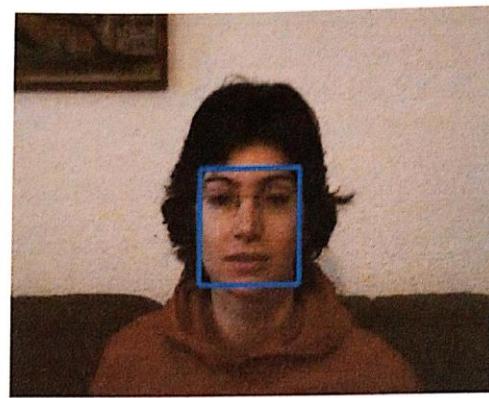


Figure 3.5 Objets à suivre dans la séquence face

#### Résultats :

Les algorithmes 1.2 et 1.1 effectue le suivi sur toute la séquence avec quelques difficultés de localisation au moment de l'occlusion mais sans incident.

Les autres algorithmes échouent face à l'occlusion en convergeant vers l'objet en mouvement ou des zones erronées.

#### 3.4.5 Séquence Jump

Objectif : suivre la moto.

Difficulté : changement d'échelle



Figure 3.6 Objets à suivre dans la séquence Jump

#### Résultats :

Les algorithmes de la seconde approche parviennent à effectuer un suivi sur toute la séquence.

L'algorithme 1.3 réussit à suivre l'objet au début de la séquence mais le perd l'objet en le confondant avec la scène, les changements subis à l'objet étant trop importants les autres algorithmes sont inutilisables.

#### 3.4.6 Séquence Sunshade

**Objectif :** suivre la personne.

**Difficulté :** changement de luminosité

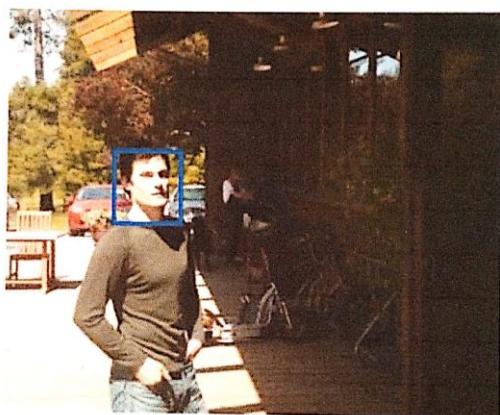


Figure 3.7 Objets à suivre dans la séquence Sunshade

#### Résultats :

L'algorithme 2.2 et 2.1 réussit à faire une approximation moyenne de l'objet.

Les algorithmes de la première approche ne supportent pas le changement brusque de luminosité et ne détectent l'objet que quand il est au soleil.

#### 3.5 Évaluation générale des algorithmes :

L'algorithme 1.1 est robuste aux occlusions partielles de l'objet et dans le cas où l'objet est perdu ce dernier peut être relocalisé s'il réapparaît n'importe où sur la scène. Par contre cet algorithme se limite pour le suivi d'objet qui ne subisse pas de grands changements au cours de la séquence.

L'algorithme 1.2 améliore le précédent en lui permettant de suivre l'objet s'il subit des changements progressifs de luminosité. Si les changements sont brusques nous obtiendrons les mêmes limites

L'algorithme 1.3 permet de suivre un objet même s'il subit des grands changements dans la séquence. Mais échoue face aux occlusions.

### Chapitre 3 : Validation des algorithmes proposés sur des bases de données

L'algorithme 2.1 permet de suivre un objet en mouvement par rapport à la scène, mais pas si nous avons un mouvement de camera. Si l'objet est perdu on ne pourra plus le relocaliser.

L'algorithme 2.2 améliore l'algorithme précédent en prenant en charge les mouvements de la camera. Si nous ne détectons pas le mouvement de l'objet nous projetons avec la scène, mais l'absence de mouvement peut-être dû à une erreur de calcul ce qui va nous éloigner de notre objet et le perdre.

#### 3.6 Conclusion

Nous avons pu dans ce chapitre d'évaluer les performances réelles de nos algorithmes appliqués au dernier benchmark de cette discipline. Ceci nous a permis de voir quelle sont les capacités de nos algorithmes ainsi que leur véritables faiblesses.





### Conclusion générale et perspectives

Au cours de ce stage de fin d'étude où nous avions comme objectif initial d'utiliser l'algorithme SURF pour faire un suivi d'objet dans une séquence vidéo, de nous plonger dans ce vaste domaine et d'acquérir les bases théoriques essentielles.

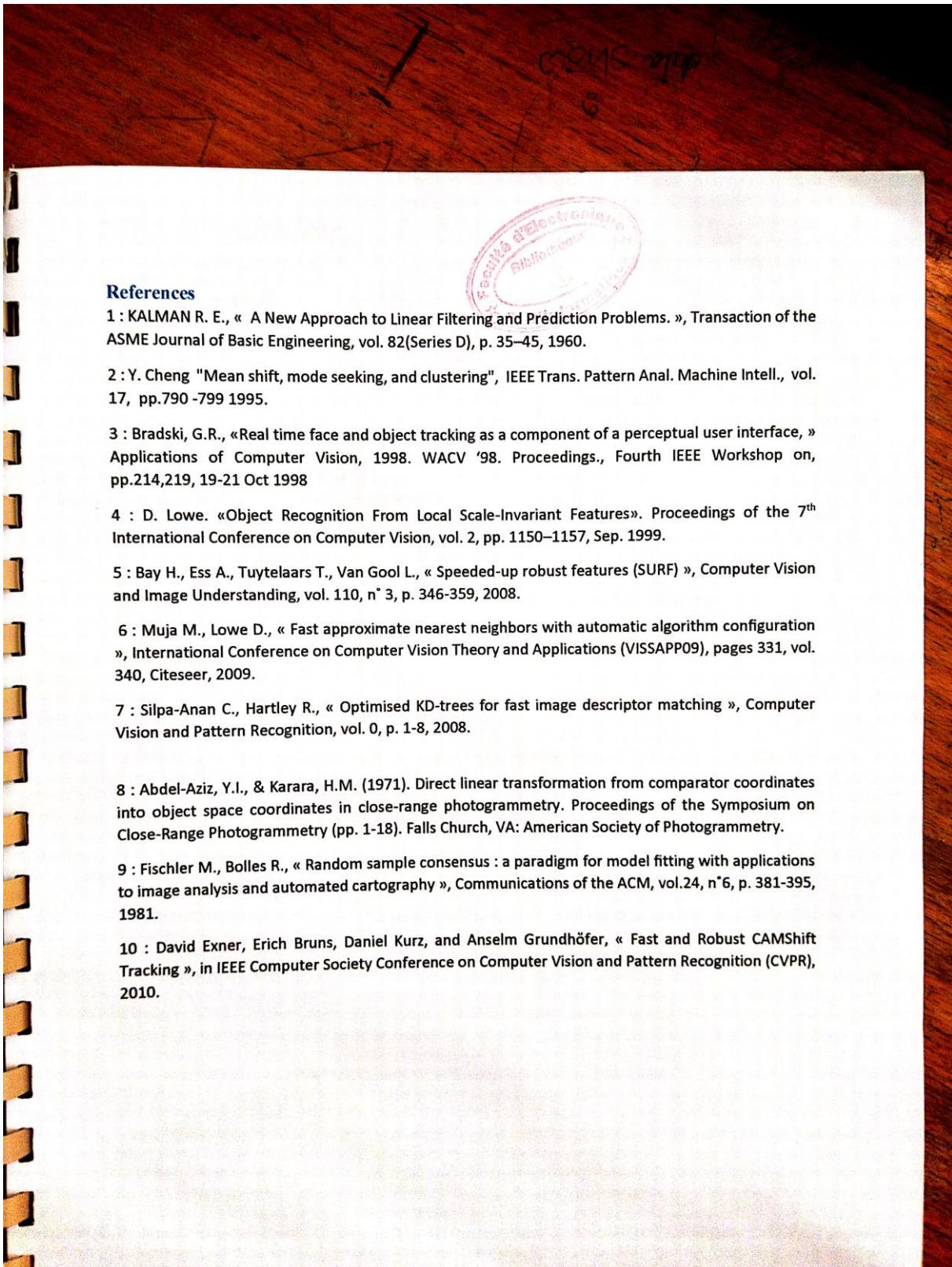
Nous avons étudié de près l'algorithme SURF et d'utiliser ses subtilités pour voir comment elles peuvent être intégrées dans des algorithmes de suivi d'objet.

En pratique, nous avons proposé différentes méthodes pour le suivi d'objet que nous avons évalué sur des récents benchmarks qui nous ont permis de voir les avantages et faiblesses de chaque méthodes.

Nous avons développé une application pour chaque approche que nous pouvons utiliser sur des séquences vidéo ou en temps réel avec une webcam.

Nous citons comme perspectives les idées suivantes :

- un algorithme de suivi est aussi évalué par rapport à sa rapidité d'exécution. L'optimisation des temps de calcul est l'étape suivante pour nos algorithmes.
- On peut fusionner les points forts de chaque algorithme pour essayer d'avoir un algorithme plus robuste.
- Les descripteurs SURF ne garantissent pas une invariance face au changement de luminosité. On peut essayer d'ajouter d'autres paramètres prenant en considération ce type de changement.



## References

- 1 : KALMAN R. E., « A New Approach to Linear Filtering and Prediction Problems. », Transaction of the ASME Journal of Basic Engineering, vol. 82(Series D), p. 35–45, 1960.
- 2 : Y. Cheng "Mean shift, mode seeking, and clustering", IEEE Trans. Pattern Anal. Machine Intell., vol. 17, pp.790-799 1995.
- 3 : Bradski, G.R., «Real time face and object tracking as a component of a perceptual user interface, » Applications of Computer Vision, 1998. WACV '98. Proceedings., Fourth IEEE Workshop on, pp.214,219, 19-21 Oct 1998
- 4 : D. Lowe. «Object Recognition From Local Scale-Invariant Features». Proceedings of the 7<sup>th</sup> International Conference on Computer Vision, vol. 2, pp. 1150–1157, Sep. 1999.
- 5 : Bay H., Ess A., Tuytelaars T., Van Gool L., « Speeded-up robust features (SURF) », Computer Vision and Image Understanding, vol. 110, n° 3, p. 346-359, 2008.
- 6 : Muja M., Lowe D., « Fast approximate nearest neighbors with automatic algorithm configuration », International Conference on Computer Vision Theory and Applications (VISSAPP09), pages 331, vol. 340, Citeseer, 2009.
- 7 : Silpa-Anan C., Hartley R., « Optimised KD-trees for fast image descriptor matching », Computer Vision and Pattern Recognition, vol. 0, p. 1-8, 2008.
- 8 : Abdel-Aziz, Y.I., & Karara, H.M. (1971). Direct linear transformation from comparator coordinates into object space coordinates in close-range photogrammetry. Proceedings of the Symposium on Close-Range Photogrammetry (pp. 1-18). Falls Church, VA: American Society of Photogrammetry.
- 9 : Fischler M., Bolles R., « Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography », Communications of the ACM, vol.24, n°6, p. 381-395, 1981.
- 10 : David Exner, Erich Bruns, Daniel Kurz, and Anselm Grundhöfer, « Fast and Robust CAMShift Tracking », in IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2010.

## RESUME

Le suivi d'objet dans une séquence vidéo est un processus de base pour plusieurs applications telles que la réalité augmenter, suivi de cible par un robot mobile, reconnaissance de mouvement ...etc. un system de suivi d'objet robuste doit parvenir à localiser la position de l'objet tous au long de la séquence de manier fiable et doit faire face aux changements d'angle de vue, d'intensités lumineuses, déformation de l'objet, l'occultation...etc. rencontré par l'objet.

Dans ce document nous allons étudier et utiliser l'algorithme SURF pour voir quel sont ces performances pour le suivi d'objet. Nous avons développé deux approches avec plusieurs variantes que nous avons évaluées par la suite sur les dernier benchmark d'iccv. Les résultats obtenues sont présentés et discuter.

Mots clefs : Suivi d'objet, SURF, Flux optique, Homographie.

