FIGURE 1 – exemple d'images successive

# Exemple d'application en traitement d'images

## Contexte

Les deux images de la figure 1 correspondent à deux images successives d'une vidéo acquise par un drone.

L'exploitation de ce type de vidéo (par exemple la détection de personnes) passe généralement par l'application de traitement relativement lent alors qu'une nouvelle image est acquise toutes les 40 millisecondes. Un pré traitement souvent utilisé pour réduire le temps de calcul consiste à essayer de recaler les images les unes sur les autres pour ne pas à avoir à traiter une même zone géographique dans plusieurs images.

Compte tenue du faible décalage entre deux images successives on peut obtenir des résultats satisfaisant en faisant simplement l'hypothèse que la transformation entre les 2 images est proche d'une transformation affine : c'est à dire qu'un point $p$ de l'ancienne image devient $q = Tp + t$ dans la nouvelle ($p, q, t \in \mathbb{R}^2$, $T \in M_{2,2}(\mathbb{R})$).

Pour estimer $T, t$, la méthode généralement privilégier consiste à extraire des 2 images des motifs suffisamment saillant pour qu'on puisse les appareiller deux à deux. Un exemple d'extraction sur les deux images est donnée en figure 2 (les motifs sont simplement extraits dans des zones de forte variation de couleur).

On obtient ainsi un ensemble de couples de points $p_1, q_1, ..., p_N, q_N$ supposés se correspondre entre les deux images (puisque supposés correspondre au même motif), à partir desquels, on peut estimer $T, t$.

## Estimation de T et t

Dans les questions de cette partie, vous pouvez écrire des formulations formelles en fonction des $(p_n, q_n)_{n\in\{1,...,N\}}$ (notamment à l'aide du quantificateur $\forall$) et/ou écrire la formulation spécifiquement pour les 8 couples ($p$,$q$) de points suivants (un sous ensemble de ceux de la figure 2) :

— $(10, 18)$ $(15, 19)$
— $(83, 174)$ $(88, 179)$

FIGURE 2 – exemple d'extraction de points
Les motifs extraits des deux images sont localisés sur les points bleus. Le décalage entre les deux images est mis en évidence en représentant en jaune dans la deuxième image les points obtenus dans la première image : on voit qu'ils sont proches des motifs correspondants mais décalés.

— $(49, 373)$ $(50, 376)$
— $(167, 49)$ $(162, 55)$
— $(183, 123)$ $(178, 129)$
— $(344, 241)$ $(339, 246)$
— $(369, 397)$ $(368, 403)$
— $(603, 4)$ $(609, 6)$

Des codes sont attachés au document pour vous aider (lisez l'intégralité du sujet avant de commencer).

### Appareillement stricte

Écrivez le système d'équation (attention $T_{1,1}, T_{1,2}, T_{2,1}, T_{2,2}, t_1, t_2$ sont des inconnues à ce niveau) correspondant simplement à la transformation des points $p_n$ en $q_n$ par la relation $q = Tp + t$.

Ce système est généralement infaisable : les appareillements sont bruitées et la transformation n'est pas réellement affine. Il est nécessaire d'utiliser une estimation robuste au bruit : on va chercher $T, t$ tel que $q_n \approx Tp_n + t$.

### Minimisation de la somme

Une façon de faire est de chercher $T, t$ tel que l'écart (en norme 1) entre $q_n$ et $Tp_n + t$ soit en **moyenne** le plus petit possible. Écrivez le programme linéaire correspondant.

Aide : si $v$ est une variable alors en introduisant une variable auxilière $a$ et en écrivant $a + v \geq 0$ $a - v \geq 0$, on peut s'assurer que $a$ est supérieur (pas forcément égale) à la valeur absolue de $v$ !

Rappel : la norme 1 d'un point $p = (p_x, p_y)$ est le nombre $|p_x| + |p_y|$.

### Minimisation de l'erreur maximale

Une autre façon de faire est de minimiser **le plus grand** des écarts (en norme 1) entre les $q_n$ et les $Tp_n + t$ correspondants. Écrivez le programme linéaire correspondant.

Cette deuxième formulation est meilleure quand le bruit est faible car on y sacrifie aucun point. En effet, dans la première on préfère avoir une erreur de 1 sur 7 points et une erreur de 8 sur le 8ème plutôt que d'avoir une erreur de 2 sur tous, ce qui signifie que sur une importante portion de l'image la transformation est très mal estimée. Mais dans le cas il y a des couples $p_n, q_n$ erronnés (il y a eu une erreur lors de l'extraction des motifs), l'erreur maximale sera nécessairement haute et chercher à la minimiser aboutira à mal estimer la transformation (Ce problème existe aussi avec la minimisation de la somme mais est encore plus significatif avec l'erreur maximale).

### Avec exclusion

Une solution consiste à détecter des couples faux.

Reprendre les 2 estimations précédente en autorisant la minimisation à considérer que $k$ (prennez $k = 2$ avec les 8 points) des couples sont faux et ne doivent donc pas être pris en compte dans la minimisation. (cela correspondrait à résoudre $k$ parmis $N$ problèmes où en enlève $k$ des points - si $k$ est une fraction de $N$ il est bien plus simple et efficace de laisser le solver exploré efficacement la combinatoire résultante).

## Matériel associé et questions bonus

### Visualisation

Dans le cas où vous avez écrit les modèles pour les 8 points, vous devriez pouvoir les résoudre rapidement avec des solvers simples (excel, openoffice, lpsolve). La résolution devrait mettre quelques secondes pour les minimisation sans exclusion et une petite minute maximum pour les minimisation avec exclusion.

Vous devriez aussi pouvoir visualiser la transformation (et ainsi vérifier qu'il n'y a pas d'erreur grossière dans votre modèle) à l'aide du code fourni en C++ ou java. Dans les 2 cas ça va produire un svg (pour des raisons de portabilité) normalement lisible nativement par l'ordinateur (essayer avec un navigateur sinon par exemple firefox ou installer inkscape si c'est votre machine). Ce svg est similaire à celui de la figure 2 avec des points verts supplémentaires qui correspondent à la position estimée dans l'image 2 des points de l'image 1 après transformation. L'objectif est que les points verts soient le plus près possibles des points bleus (le résultat n'est pas forcément très probant surtout sans exclusion mais vous devriez avoir des points verts meilleurs que les points jaunes) Cela n'est là que pour vous aider à visualiser la transformation obtenue. Ce n'est pas nécessaire à la réalisation du mini projet.

**En C++ :** Ouvrez le fichier main.cpp . Remplacer les valeurs de Txx, Txy, Tyx, Tyy, tx, ty de la fonction main par les valeurs que vous avez obtenu. Compilez le dans le dossier et lancez le programme.

**En java :** Ouvrez le fichier Main.java . Remplacer les valeurs de Txx, Txy, Tyx, Tyy, tx, ty de la fonction main par les valeurs que vous avez obtenu. Soit lancer le programme avec un IDE (attention aux chemins relatifs) soit compiler le et lancer le en ligne de commande.

### Avec plus de points

Écrivez un programme C++ ou Java qui qui écrit le modèle linéaire (nommé *model.lp*) qu'on cherche à résoudre dans le format lp-solve pour un ensemble quelconque de couples de points lu par le programme (vous n'avez pas à écrire de fonction pour lire les fichiers de points : il vous suffit de récupérer le code correspondant dans main.cpp/main.java).

Vous pouvez alors tester avec les 4 formulations avec les deux fichiers de points disponibles dans l'archive (Ne le faites pas à la main).

Il n'est pas possible de lancer la résolution du programme linéaire que vous venez d'écrire directement depuis votre code (sauf en c++ sous linux avec la commande *system("lp_solve model.txt");*) mais ce n'est pas de toute façon ce qui est demandé.

### L'état de l'art

Les solutions obtenues avec ce type de méthode sont généralement de très bonne qualité ! Mais, malheureusement, ces méthodes sont trop lentes : dans ce contexte, il convient de ne pas mettre plus de 5ms pour estimer la transformation (on a une nouvelle image toutes les 40ms et il faut laisser au moins 35ms par image pour les traitements lourds comme la détection de personnes).

La méthode couramment utilisée en pratique consiste à résoudre la minimisation de l'écart quadratique moyen (c'est à dire $\|q - Tp - t\|^2$) sans exclusion : on peut alors ramener le problème à deux simples systèmes linéaire à 3 inconnues (les voyez vous ?) très facilement résoluble. Afin d'être robuste au bruit, cette méthode est couramment couplée avec la méthode stochastique RANSAC permettant de tester un certain nombres d'exclusions (excellent résultat en moyenne - mais possibilité d'un très mauvais résultat).

Au passage si ce genre de problématique vous intéresse, je vous encourage à consulter la liste des stages/postes proposés par l'ONERA (équivalent de la NASA en France - http ://www.onera.fr/fr/emploi-et-formation/offres-emploi) ou à m'envoyer une candidature spontanée (adrien.chan_hon_tong@onera.fr)