TD4 - Traitement d'image - Détection de points d'intérêts

L'objectif de cette séance est de retrouver automatiquement la transformation géométrique qui a eu lieu entre 2 images d'une même scène. Nous allons d'abord travailler sur une transformation affine, puis sur une homographie. Pour cela, nous allons extraire des descripteurs (du type SIFT) de points d'intérêts et les mettre en correspondance entre les deux images étudiées.

Contents

2.1 Extraction descripteurs SIFT

```
% nom image
im1_name = 'bear1';
im2_name = 'bear2';

% extraction des descripteurs sift sur image 1
im1 = imread([im1_name,'.jpg']);
[im1_gray, d1, l1] = sift([im1_name,'.jpg']);
% nombre de descripteurs sift
n1=length(l1);

% extraction des descripteurs sift sur image 2
im2 = imread([im2_name,'.jpg']);
[im2_gray, d2, l2] = sift([im2_name,'.jpg']);
% nombre de descripteurs sift
n2=length(l2);

% visualisation des points d'intérêts
figure;
subplot(1,2,1);
imshow(im1); title('image 1'); hold on
plot(l1(:,2),l1(:,1),'+g');
subplot(1,2,2);
imshow(im2); title('image 2'); hold on
plot(l2(:,2),l2(:,1),'+g');
```

```
Finding keypoints...
52 keypoints found.
Finding keypoints...
63 keypoints found.
```

2.2 Comparaison des descripteurs SIFT

```
% distance entre descripteurs sift
tdist = zeros(n1,n2);
for i=1:n1
    for j=1:n2
        tdist(i,j) = norm(d1(i,:)-d2(j,:));
    end;
end;

% appariement des descripteurs sift
[dis, ind] = min(tdist);
```

2.3 Calcul de la transformation affine

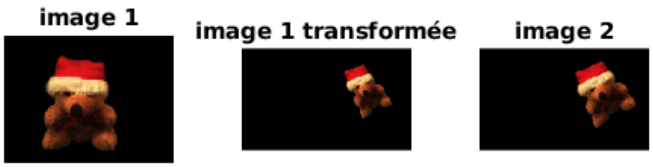
$x' = a\,x + b\,y + c$

$y' = d\,x + e\,y + f$

```
x2 = [];
x1 = [];
for i=1:n2
    x2 = [x2 [l2(i,1);l2(i,2);1]];
    x1 = [x1 [l1(ind(i),1);l1(ind(i),2);1]];
end

T1 = x2*pinv(x1);
```

2.4 Application de la transformation

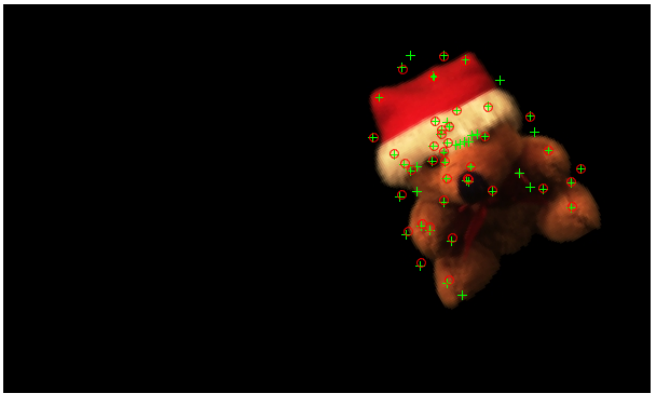
```
imt1 = applique_transfo(im1,T1,size(im2));

figure;
subplot(1,3,1); imshow(im1); title('image 1');
subplot(1,3,2); imshow(imt1); title('image 1 transformée');
subplot(1,3,3); imshow(im2); title('image 2');
```

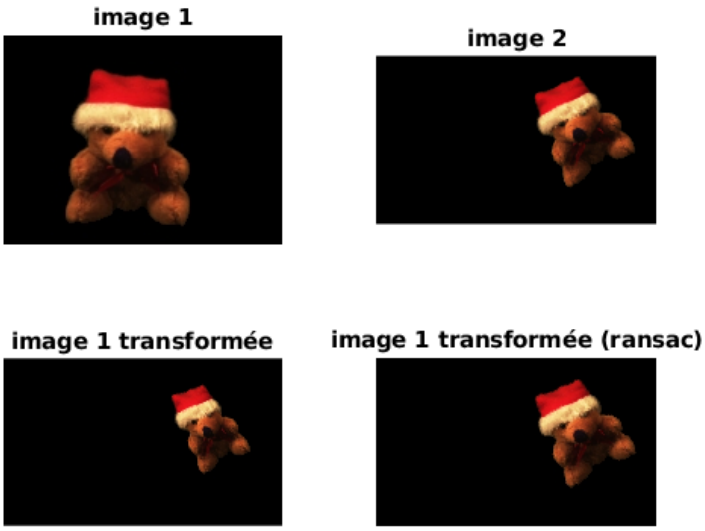


L'image 1 a bien subie la transformation affine translation + rotation représentée par T1.

2.5 Calcul robuste de la transformation affine par l'algorithme RANSAC

```
% ransac
nb_step=1000;

for k=1:nb_step
    i2 = ceil(rand(1,3)*n2);
    t2 = [];
    t1 = [];
    for i=1:3
        t2 = [t2 [l2(i2(i),1);l2(i2(i),2);1]];
        t1 = [t1 [l1(ind(i2(i)),1);l1(ind(i2(i)),2);1]];
    end
    T = t2*pinv(t1);
    nbr(k) = length(find(sqrt(sum((x2-T*x1).^2))<6));
    transf{k} = T;
end
[y,index] = max(nbr);
T = transf{index};

% application de la transformation sur l'image 1
imt = applique_transfo(im1,T,size(im2));
x2t = T*x1;

% comparaison des résultats
figure;
imshow(im2); hold on
plot(l2(:,2),l2(:,1),'+g');
plot(x2t(2,:),x2t(1,:),'or');

figure;
subplot(2,2,1); imshow(im1); title('image 1');
subplot(2,2,2); imshow(im2); title('image 2');
subplot(2,2,3); imshow(imt1); title('image 1 transformée');
subplot(2,2,4); imshow(imt); title('image 1 transformée (ransac)');
```





Transformation estimée plus stable avec l'algorithme RANSAC.

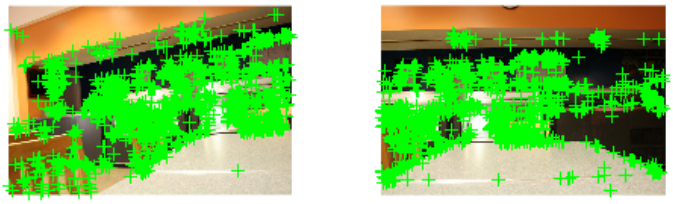
2.6 Modélisation par une transformation affine et mosaïquage

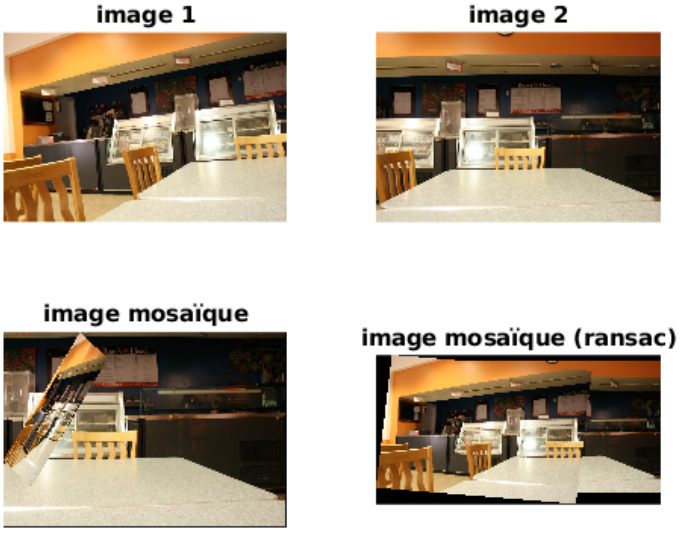
```
clc;
close all;
clear all;

% nom image
im1_name = 'bar1';
im2_name = 'bar2';

% extraction des descripteurs sift sur image 1
im1 = imread([im1_name,'.jpg']);
[im1_gray, d1, l1] = sift([im1_name,'.jpg']);
% nombre de descripteurs sift
n1=length(l1);

% extraction des descripteurs sift sur image 2
im2 = imread([im2_name,'.jpg']);
[im2_gray, d2, l2] = sift([im2_name,'.jpg']);
```

```
% nombre de descripteurs sift
n2=length(l2);

% visualisation des points d'intérêts
figure;
subplot(1,2,1);
imshow(im1); hold on
plot(l1(:,2),l1(:,1),'+g');
subplot(1,2,2);
imshow(im2); hold on
plot(l2(:,2),l2(:,1),'+g');

% distance entre descripteurs sift
tdist = zeros(n1,n2);
for i=1:n1
    for j=1:n2
        tdist(i,j) = norm(d1(i,:)-d2(j,:));
    end;
end;

% appariement des descripteurs sift
[dis, ind] = min(tdist);

% calcul de la transformation affine
x2 = [];
x1 = [];
for i=1:n2
    x2 = [x2 [l2(i,1);l2(i,2);1]];
    x1 = [x1 [l1(ind(i),1);l1(ind(i),2);1]];
end

T1 = x2*pinv(x1);

% mosaïquage entre image 1 et image 2
imt1 = mosaic_affine(im1,T1,im2);

figure;
subplot(1,3,1); imshow(im1); title('image 1');
subplot(1,3,2); imshow(im2); title('image 2');
subplot(1,3,3); imshow(imt1); title('image mosaïque');

%  calcul robuste de la transformation affine par l'algorithme RANSAC
nb_step=1000;

for k=1:nb_step
    i2 = ceil(rand(1,3)*n2);
    t2 = [];
    t1 = [];
    for i=1:3
        t2 = [t2 [l2(i2(i),1);l2(i2(i),2);1]];
        t1 = [t1 [l1(ind(i2(i)),1);l1(ind(i2(i)),2);1]];
    end
    T = t2*pinv(t1);
    nbr(k) = length(find(sqrt(sum((x2-T*x1).^2))<6));
    transf{k} = T;
end
[y,index] = max(nbr);
T = transf{index};
```

```
% mosaïquage entre image 1 et image 2 avec la nouvelle transformation affine
imt = mosaic_affine(im1,T,im2);

% % comparaison résultats
% x2t = T*x1;
% figure;
% imshow(im2); hold on
% plot(l2(:,2),l2(:,1),'+g');
% plot(x2t(2,:),x2t(1,:),'or');

figure;
subplot(2,2,1); imshow(im1); title('image 1');
subplot(2,2,2); imshow(im2); title('image 2');
subplot(2,2,3); imshow(imt1); title('image mosaïque');
subplot(2,2,4); imshow(imt); title('image mosaïque (ransac)');
```

```
Finding keypoints...
1707 keypoints found.
Finding keypoints...
1365 keypoints found.
```





Le résultat du mosaïquage n'est pas optimal même lorsque l'algorithme RANSAC est utilisé pour estimer de façon plus robuste la transformation affine liant les deux images. En effet, l'image 2 ne correspond pas à une transformation affine de l'image 1 (changement de plan).

2.7 Modélisation par une transformation homographique et mosaïquage

[x' y' z']^T = [H11 H12 H13 ; H21 H22 H23 ; H31 H32 H33] * [x y z]^T

x" = x'/z' et y" = y'/z' et z = 1 mènent à:

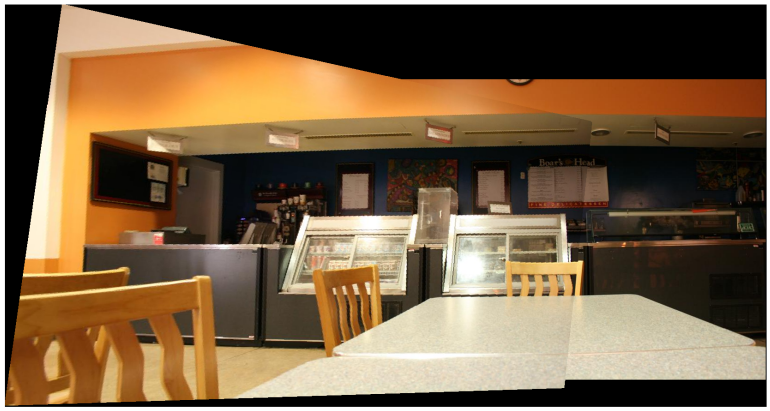
H11*x + H12*y + H13 - x"(H31*x + H32*y + H33) = 0

H21*x + H22*y + H23 - y"(H31*x + H32*y + H33) = 0

résolu par SVD

```
% normalisation
[x1n, T1] = normalise2dpts(x1);
[x2n, T2] = normalise2dpts(x2);
```

```
% calcul robuste de la transformation homographique par l'algorithme RANSAC
for k=1:nb_step
    i2 = ceil(rand(1,5)*n2);
    A = zeros(10,9);
    for i=1:5
        A(1+2*(i-1),:) = [x1n(1,i2(i)) x1n(2,i2(i)) 1 0 0 0 -x2n(1,i2(i))*[x1n(1,i2(i)) x1n(2,i2(i))
        A(2+2*(i-1),:) = [0 0 0 x1n(1,i2(i)) x1n(2,i2(i)) 1 -x2n(2,i2(i))*[x1n(1,i2(i)) x1n(2,i2(i))
    end;
    [u,s,v] = svd(A);
    H = reshape(v(:,9),3,3)';
    % suppression normalisation
    H = T2\H*T1;
    x1hat = H*x1;
    x1hat = x1hat./repmat(x1hat(3,:),3,1);
    nbr(k) = length(find(sqrt(sum((x2-x1hat).^2))<6));
    transf{k} = H;
end;
[y, index] = max(nbr);
H = transf{index};

% mosaïquage entre image 1 et image 2 avec la nouvelle transformation homographique
imt = mosaic_homographie(im1,H,im2);
figure;
imshow(imt);
```



Meilleur résultat en prenant en compte une transformation homographique (correspond davantage à la transformation observée entre les deux images).

Published with MATLAB® R2017b