ESIR 3ème Année Informatique
Image Numérique
Module SFX

TP Warping & Morphing

Auteur : Rémi Cozot (Septembre 2012)

Présentation : Le but du TP est la mise en œuvre et l'étude d'algorithmes de « warping » et « morphing » d'images.

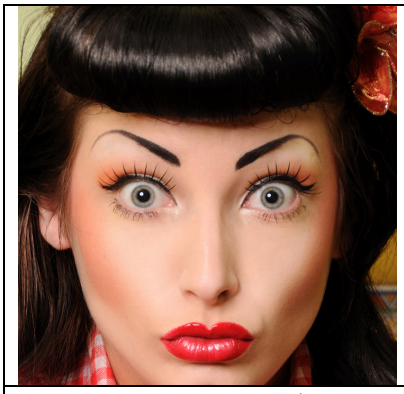
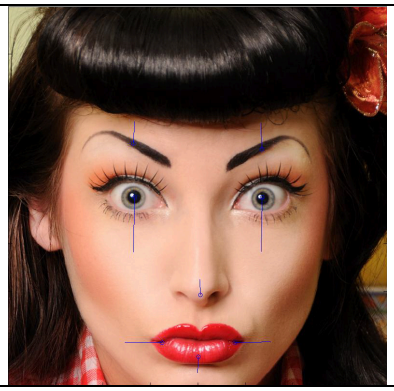
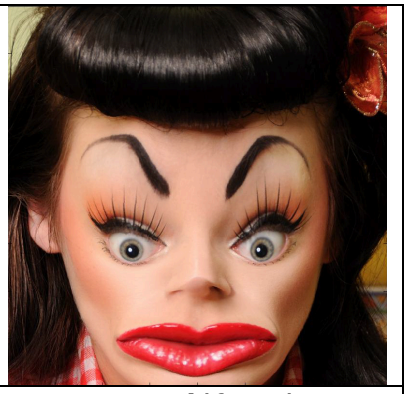

Image source[1]	Champ de déplacement	Image déformée

Figure 1 : déformation (warping) d'image basée points

Warping basé points

Dans cette première partie, le code d'un warping simple (basé points) est foruni.La définition du champ de déplacement se fait en spécifiant quelques vecteurs de déplacements sur l'image source.

Le champ de déplacement continu (pour tous les pixels) est donné par interpolation de fonctions gaussiennes. L'algorithme fonctionne en mode direct.

Question 1 :
Faites quelques tests à partir du logiciel fourni. Quels sont les artefacts qui apparaissent ?

Question 2 :
Proposez au moins deux approches pour remplier les trous qui apparaissent lors de la déformation de l'image. Etudiez les limites des approches que vous proposer. Mettez en œuvre l'une des approches proposées.

[1] Crédit photo : armor-photo.com, tous droits réservés, reproduction interdite.

Morphing basé « features »

Dans cette deuxième partie, vous développez un algorithme de morphing basé « features » en mode inverse.

Question 3

Rappelez le principe de morphing basé « features » (vous utiliserez ici des vecteurs), décrivez l'algorithme.

Question 4

Mettez en œuvre (en MATLAB) l'algorithme. Testez l'aalgorithme avec différentes fonctions de pondérations (2 fonctions de pondération au minimum) et analysez l'influence des fonctions de pondérations sur le résultat.

Travail à Rendre
Code matlab de l'algorithme de warping basé point sans artefact
Dossier d'analyse de l'algorithme de morphing basé « features »
Code matlab de l'algorithme de morphing basé « features »
Dossier d'analyse des résultats de l'algorithme de morphing basé « features » , discussion sur le choix des fonctions de pondérations.

Annexe 1 : Code MATLAB

```
function [ output_args ] = pointwarpp( input_args )
%-------------------------------------------------------------------------
% POINT BASED WARPING
% ------------------------------------------------------------------------
% Draft version
%
% example of point base warping
%
% displacement field kernel : gaussian function
% hole filling : To Be Done
% ------------------------------------------------------------------------
% Author: Remi Cozot
% Date: June 2012
% ------------------------------------------------------------------------

% select and display image
[filename,pathname]=uigetfile('*.jpg', 'select input image');
img = imread(strcat(pathname,filename));
image(img); axis image ; hold on ;
[sy sx sc] = size(img);

% warping : point based

% define displacement vectors
cont = 1 ;
cl = 0 ;
pt = 0 ;
while cont
    [x,y,b] = ginput(1);
    % enter the point
    if cl==0
        % first point
        vect(pt+1,:) = [x y 0 0];
    else
        % end point
        vect(pt+1,:) = vect(pt+1,:)+ [0 0 x y];
        plot([vect(pt+1,1) vect(pt+1,3)],[vect(pt+1,2) vect(pt+1,4)],'-');
        plot(vect(pt+1,1), vect(pt+1,2),'o');
        pt = pt+1;
    end
    cl = mod(cl+1,2);
    if b==3
        cont = 0 ;
    end
end
% computing the warp
% new image
newimg = uint8(zeros(sy,sx,sc));

% for all pixel
for yi=1:sy
    for xi=1:sx
        % for all displacementvectors
        dp =[0 0];
        for k=1:pt
            % displacement length gives sigma2
            dx = vect(k,3)-vect(k,1);
            dy = vect(k,4)-vect(k,2);
            sigma2 = dx*dx' + dy*dy';
```

```
            % add displacement
            d = [xi-vect(k,1), yi-vect(k,2)];
            dk = [dx dy]*exp(-2*(d*d')/(2*sigma2));
            dp = dp +dk;
        end
    % end for all vectors
    xx =clamp(1,xi+round(dp(1)),sx);
    yy = clamp(1,yi+round(dp(2)),sy);
    % compute new image
    % draw pixel
    newimg(yy,xx,:)=img(yi,xi,:);
    end
end
% end for all pixel

% post processing

figure;image(newimg); axis image ;

end
% function
function res = clamp(mi,v,ma)
res = min(ma,max(mi,v));
end
```