تبدیل هندسی

مینی‌پروژه شماره ۲

مهدی صادقی ۹۶۱۳۶۶۶۱۷۵

موعد تحویل: ۹۶/۰۸/۱۷

تاریخ تحویل: ۹۶/۰۸/۱۴

خلاصه

هدف از انجام پروژه این است که با استفاده از تبدیل هندسی یک تصویر را از یک مکان به مکانی دیگر انتقال دهیم . روش کار بدین صورت می‌باشد که در ابتدا یک تصویر پایه را انتخاب می‌کنیم (در اینجا ما تصویر lm431.bmp را در پنجره Browse اول به عنوان تصویر پایه انتخاب می‌کنیم) که باید تصاویر رنگی که در پنجره‌های بعدی انتخاب می‌کنیم برروی مربعات مشکی این تصویر نگاشت پیدا کنند. برای انجام این کار ابتدا ۴ نقطه را به عنوان نقاط اولیه در دو تصویر انتخاب می‌کنیم سپس با دانستن این چهار نقطه از هر دو تصویر ضرایب معادله نگاشت خود را بدست می‌آوریم. برای انجام تبدیل هندسی از دو معادله نگاشت استفاده شده است معادله نگاشت اول به عنوان معادله نگاشت شماره سطر و دیگری به عنوان معادله نگاشت شماره ستون در نظر گرفته شده است پس از مشخص شدن ضرایب معادلات، تمامی پیکسل‌های تصویر تحت عمل نگاشت قرار می‌گیرند و در حین عمل نگاشت در صورت لزوم از یکی از دو درون‌یابی نزدیک‌ترین همسایه و یا دوخطی استفاده شده است. برای انجام این پروژه از نگاشت معکوس استفاده شده است.

شرح تکنیکال:

با توجه به توضیحات داده شده در ابتدا همانطور که بیان کردیم هدف از ایجاد این سامانه نگاشت تصویر به مکانی دیگر می‌باشد. در اینجا ابتدا با استفاده از دستور uigetfile تصویر پایه (که در اینجا همان تصویر Img431.bmp می‌باشد) را دریافت کردیم سپس مختصات نقاط چهار گوشه مربع‌های مشکی را با استفاده از فتوشاپ بدست آوردیم (همچنین می‌توانستیم با استفاده از دستور cpselect مختصات این چهار نقطه را بدست آورد) در ادامه با دستور uigetfile تصویری که قصد داریم نگاشت پیدا کند را دریافت می‌کنیم سپس ضرایب معادله نگاشت را بدست می‌آوریم و در نهایت تصویر را نگاشت می‌دهیم. در ادامه به شرح دقیق‌تر این مسائل می‌پردازیم.

همانطور که در قبل بیان کردیم ابتدا یک معادله نگاشت بدست می‌آوریم که این معادله نگاشت برای سطرها فرمول ۱ و برای ستون‌ها فرمول ۲ می‌باشد.

$$X = a_1x + a_2y + a_3xy + a_4 \quad (۱)$$

$$Y = a_5x + a_6y + a_7xy + a_8 \quad (۲)$$

که در این روابط $x،y$ سطر و ستون تصویر مبدا و $X،Y$ سطر و ستون تصویر مقصد می‌باشد.

معادلات ۱ و ۲ معادلات نگاشت مستقیم می‌باشند. در بسیاری از نگاشت‌ها باید از نگاشت معکوس استفاده کرد و حل بسیاری از نگاشت‌ها با استفاده از نگاشت مستقیم امکان پذیر نمی‌باشد بنابراین در این مسئله نیز باید از نگاشت معکوس استفاده کرد. بنابراین باید از فرمول‌های ۳ و ۴ استفاده کرد.

$$x = a_1X + a_2Y + a_3XY + a_4 \quad (۳)$$

$$y = a_5X + a_6Y + a_7XY + a_8 \quad (۴)$$

که در فرمول‌های ۳ و ۴ $x،y$ سطر و ستون تصویر مبدا و $X،Y$ سطر و ستون تصویر مقصد می‌باشند. نگاشت‌های ۳ و ۴ نگاشت معکوس می‌باشند.

در ادامه با بدست آوردن ضرایب مربوط به هر کدام معادله نگاشت بدست می‌آید و سپس تصویر را نگاشت می‌دهیم.

برای بدست آوردن ضرایب معادله با جایگذاری نقاط اولیه داریم:

$$x_1 = a_1X_1 + a_2Y_1 + a_3X_1Y_1 + a_4$$
$$x_2 = a_1X_2 + a_2Y_2 + a_3X_2Y_2 + a_4$$
$$x_3 = a_1X_3 + a_2Y_3 + a_3X_3Y_3 + a_4$$
$$x_4 = a_1X_4 + a_2Y_4 + a_3X_4Y_4 + a_4$$

با داشتن ۴ معادله و ۴ مجهول می‌توان این معادلات را به عنوان یک دستگاه در نظر گرفت و این دستگاه معادله را حل کرد که در ادامه به توضیح حل این دستگاه معادله می‌پردازیم.

برای حل این دستگاه معادله از روش ماتریس استفاده شده است. در ابتدا دستگاه را به صورت زیر بازنویسی می‌کنیم

$$\underbrace{\begin{bmatrix} X_1 & Y_1 & X_1Y_1 & 1 \\ X_2 & Y_2 & X_2Y_2 & 1 \\ X_3 & Y_3 & X_3Y_3 & 1 \\ X_4 & Y_4 & X_4Y_4 & 1 \end{bmatrix}}_{\text{A}} \underbrace{\begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}}_{\text{Z}} = \underbrace{\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}}_{\text{B}}$$

بنابراین داریم

$$AZ = B$$

طبق فرمول بالا برای بدست آوردن ماتریس ضرایب باید طرفین رابطه را در A^{-1} ضرب کنیم بنابراین داریم

$$A^{-1}A\,Z = A^{-1}\,B \;\rightarrow\; Z = A^{-1}B$$

بنابراین با ضرب معکوس A در ماتریس B ماتریس ضرایب بدست می‌آید.

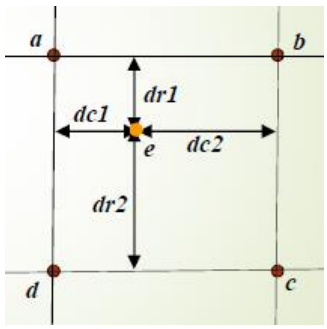
در نهایت بعد از این که ضرایب بدست آمدند معادله نگاشت بدست می‌آید و می‌توان تمام تصویر را با استفاده از این معادله نگاشت، تبدیل کرد.

توجه: در اینجا ما فقط روش بدست آوردن ضرایب معادله نگاشت X را شرح داده‌ایم ضرایب معادله Y نیز به همین روش بدست می‌آیند.

پس از مشخص شدن معادله نگاشت می‌بایست تک تک پیکسل‌ها نگاشت یابند در این عمل نگاشت به دلیل این که امکان دارد مقادیر بدست آمده دقیقا اعداد طبیعی نباشند (مثلا فرض کنید مقدار $x = 1.3$ بدست بیاید) بنابراین باید از درون یابی استفاده کرد در این مسئله در دو نوع درون یابی نزدیک‌ترین همسایه و دو خطی استفاده شده است.

در درون یابی نزدیک‌ترین همسایه اگر مختصات یک پیکسل عددی طبیعی نباشد باید آن را برابر با نزدیک‌ترین مختصات در نظر گرفت فرض کنید x=1.2 و y=1.8 از فرمول‌های نگاشت بدست آورده ایم در درون یابی نزدیک‌ترین همسایه باید این مختصات را برابر x=1 و y=2 در نظر بگیریم. برای پیاده سازی این درون‌یابی از دستور round استفاده کرده ایم که مقادیر را گرد می‌کند.

نوع دیگری از درون‌یابی درون یابی دوخطی می‌باشد. فرض کنید مختصات نگاشت یافته شده برابر مختصات نقطه e در شکل زیر باشد در این صورت داریم:



$$\mathbf{e = a \times dr2 \times dc2 + b \times dc1 \times dr2 + c \times dr1 \times dc2 + d \times dr1 \times dc2}$$

که در آن داریم:

$$\mathbf{dc1 = 1\text{-}dc2}$$

$$\mathbf{dr1 = 1\text{-}dr2}$$

بدین صورت مقدار روشنایی پیکسل e از طریق درون‌یابی دوخطی بدست می‌آید.

شرح نتایج:

نتایج بدست آمده با استفاده از درون‌یابی **نزدیک ترین همسایه** به شکل زیر می‌باشد:

همانطور که مشاهده می‌شود تصویرها به صورت بلاک بلاک در آمده‌اند

و همچنین نتایج بدست آمده از درون‌یابی **دوخطی** به صورت زیر می‌باشد:

در این تصویر همانطور که مشاهده می‌شود پدیده بلاک بلاک رخ نداده است اما تصویر حالت تیزی خود را از دست داده است و لبه‌های تصویر از بین رفته است.

پیوست:

برای پیاده سازی این پروژه دو نوع کد نوشته شده است در کد اول نقاط چهار گوشه مربع‌های تصویر پایه به صورت ثابت تعرف شده است و کد دوم تنها از تابع cpselect برای بدست آوردن مختصات چهار گوشه استفاده شده است. طریقه استفاده این تابع بدین صورت هست که پس از اجرای این دستور تصاویر را نمایش می‌دهد و کاربر با کلیک کردن در یک نقطه تنها مختصات نقطه در تصویر را برمی‌گرداند و برنامه دوم جامع تر و بهتر از برنامه اول می‌باشد.

همچنین تابع درون یابی خطی به صورت یک تابع جداگانه پیاده سازی شده است که به عنوان آخرین کد پیوست شده است.

```
%% This Script read Your image and transform to one image
%The First Image is a Base Image and another Images are Transform Image

clc
clear all
close all
check=1;
%%Read Base Images
[Name_1 path_1] = uigetfile({'*.*'},'Select Your Base Image');
Image_1 = imread([path_1 Name_1]);
Image=zeros(size(Image_1 , 1) , size(Image_1 , 2) , 3);
Final=zeros(4,2,4);
Final(:,:,1)=[244 145 ; 727 145 ; 335 351 ; 635 351];
Final(:,:,2)=[131 742 ; 131 259 ; 335 649 ; 335 351];
Final(:,:,3)=[728 854 ; 244 854 ; 635 649 ; 335 649];
Final(:,:,4)=[840 259 ; 840 742 ; 635 351 ; 635 649];
Orginal=zeros(4,2,4);
Orginal(:,:,1)=[1 1 ; 64 1 ; 1 64 ; 64 64];
Orginal(:,:,2)=[1 1 ; 64 1 ; 1 64 ; 64 64];
Orginal(:,:,3)=[1 1 ; 64 1 ; 1 64 ; 64 64];
Orginal(:,:,4)=[1 1 ; 64 1 ; 1 64 ; 64 64];
for req=1:4

    if check

    [Name_2 path_2] = uigetfile({'*.*'}, 'Select Your Image to Transform');
    Image_2 = imread([path_2 Name_2]);
    txt=['Wiche Kind of Interpolation for Image_' num2str(req) '?'];
    choice = questdlg(txt, ...
```

```
        'choice Interpolation', ...
        'Bilinear','Nearest neighbor','Nearest neighbor');
    switch choice
        case 'Bilinear'
            disp([choice ' Interpolation Selected'])
            Int = 1;
        case 'Nearest neighbor'
            disp([choice ' Interpolation Selected'])
            Int = 0;
    end


    x=Final(:,2,req);
    y=Final(:,1,req);
    % AZ_X=X;
    Z_X=zeros(1,4);
    F=[x y x.*y ones(4,1)];
    Z_X= F\Orginal(:,2,req);
    %AZ_Y=Y
    Z_Y=zeros(1,4);
    F=[x y x.*y ones(4,1)];
    Z_Y= F\Orginal(:,1,req);

    if Int==1
        for i = 1 : size(Image_1,1)
            for j = 1 : size(Image_1,2)
                xs = (Z_X(1)*i + Z_X(2)*j + Z_X(3)*i*j + Z_X(4)) ;
                ys = (Z_Y(1)*i + Z_Y(2)*j + Z_Y(3)*i*j + Z_Y(4)) ;

                if xs<1 || xs>size(Image_2,2) || ys<1 || ys>size(Image_2,1)
                    continue;
                end
                Image(i,j,1)  = Bilinearinter( [floor(xs) ceil(xs)] ,
[floor(ys) ceil(ys)] ...
                    , [Image_2(floor(xs),floor(ys),1)
Image_2(floor(xs),ceil(ys),1)...
                    Image_2(ceil(xs),floor(ys),1)
Image_2(ceil(xs),ceil(ys),1)] , [xs ys]);

                Image(i,j,2)  = Bilinearinter( [floor(xs)
ceil(xs)],[floor(ys) ceil(ys)]...
                    ,[Image_2(floor(xs),floor(ys),2)
Image_2(floor(xs),ceil(ys),2)...
                    Image_2(ceil(xs),floor(ys),2)
Image_2(ceil(xs),ceil(ys),2)],[xs ys]);

                Image(i,j,3)  = Bilinearinter( [floor(xs)
ceil(xs)],[floor(ys) ceil(ys)]...
                    ,[Image_2(floor(xs),floor(ys),3)
Image_2(floor(xs),ceil(ys),3)...
                    Image_2(ceil(xs),floor(ys),3)
Image_2(ceil(xs),ceil(ys),3)],[xs ys]);
            end
        end
    else
```

```
        for i = 1 : size(Image_1,1)
            for j = 1 : size(Image_1,2)
                xs = round(Z_X(1)*i + Z_X(2)*j + Z_X(3)*i*j + Z_X(4)) ;
                ys = round(Z_Y(1)*i + Z_Y(2)*j + Z_Y(3)*i*j + Z_Y(4)) ;

                if xs<1 || xs>size(Image_2,2) || ys<1 || ys>size(Image_2,1)
                    continue;
                end
                Image(i,j,:)=Image_2(xs,ys,:);
            end
        end

    end
    Image = uint8(Image);
    imshow(Image)


        choice = questdlg('Do You Want to Continue?', ...
        'Check', ...
        'Yes','No thank you','Yes');
    switch choice
        case 'Yes'
            disp([choice ' coming right up.'])
            check = 1;
        case 'No thank you'
            disp('Thank you for your trust :)')
            check = 0;
    end

    end

end

**********************************************************************

%% This Script read Your image and transform to one image
%The First Image is a Base Image and another Image are Transform Image

clc
clear all
close all
%%Read Base Images
[Name_1 path_1] = uigetfile({'*.*'},'Select Your Base Image');
Image_1 = imread([path_1 Name_1]);
Image=zeros(size(Image_1 , 1) , size(Image_1 , 2) , 3);
req=1;
while(req)

    [Name_2 path_2] = uigetfile({'*.*'}, 'Select Your Image to Transform');
    Image_2 = imread([path_2 Name_2]);
    [Orginal,Final] = cpselect(Image_2,Image_1,'Wait', true);
    choice = questdlg('Wiche Kind of Interpolation?', ...
        'choice Interpolation', ...
```

```
        'Bilinear','Nearest neighbor','Nearest neighbor');
    switch choice
        case 'Bilinear'
            disp([choice ' Interpolation Selected'])
            Int = 1;
        case 'Nearest neighbor'
            disp([choice ' Interpolation Selected'])
            Int = 0;
    end

    x=Final(:,2);
    y=Final(:,1);
    % AZ_X=X;
    Z_X=zeros(1,4);
    F=[x y x.*y ones(4,1)];
    Z_X= F\Orginal(:,2);
    %AZ_Y=Y
    Z_Y=zeros(1,4);
    F=[x y x.*y ones(4,1)];
    Z_Y= F\Orginal(:,1);

    if Int==1
        for i = 1 : size(Image_1,1)
            for j = 1 : size(Image_1,2)
                xs = (Z_X(1)*i + Z_X(2)*j + Z_X(3)*i*j + Z_X(4)) ;
                ys = (Z_Y(1)*i + Z_Y(2)*j + Z_Y(3)*i*j + Z_Y(4)) ;

                if xs<1 || xs>size(Image_2,2) || ys<1 || ys>size(Image_2,1)
                    continue;
                end
                Image(i,j,1)  = Bilinearinter( [floor(xs) ceil(xs)] ,
[floor(ys) ceil(ys)] ...
                    , [Image_2(floor(xs),floor(ys),1)
Image_2(floor(xs),ceil(ys),1)...
                    Image_2(ceil(xs),floor(ys),1)
Image_2(ceil(xs),ceil(ys),1)] , [xs ys]);

                Image(i,j,2)  = Bilinearinter( [floor(xs)
ceil(xs)],[floor(ys) ceil(ys)]...
                    ,[Image_2(floor(xs),floor(ys),2)
Image_2(floor(xs),ceil(ys),2)...
                    Image_2(ceil(xs),floor(ys),2)
Image_2(ceil(xs),ceil(ys),2)],[xs ys]);

                Image(i,j,3)  = Bilinearinter( [floor(xs)
ceil(xs)],[floor(ys) ceil(ys)]...
                    ,[Image_2(floor(xs),floor(ys),3)
Image_2(floor(xs),ceil(ys),3)...
                    Image_2(ceil(xs),floor(ys),3)
Image_2(ceil(xs),ceil(ys),3)],[xs ys]);
            end
        end
    else

        for i = 1 : size(Image_1,1)
            for j = 1 : size(Image_1,2)
```

```
                xs = round(Z_X(1)*i + Z_X(2)*j + Z_X(3)*i*j + Z_X(4)) ;
                ys = round(Z_Y(1)*i + Z_Y(2)*j + Z_Y(3)*i*j + Z_Y(4)) ;

                if xs<1 || xs>size(Image_2,2) || ys<1 || ys>size(Image_2,1)
                    continue;
                end
                Image(i,j,:)=Image_2(xs,ys,:);
            end
        end

    end
    Image = uint8(Image);
    imshow(Image)


    choice = questdlg('Do You Want to Continue?', ...
        'Check', ...
        'Yes','No thank you','Yes');
    % Handle response
    switch choice
        case 'Yes'
            disp([choice ' coming right up.'])
            req = 1;
        case 'No thank you'
            disp('Thank you for your trust :)')
            req = 0;
    end
end
```

تابع درون یابی دو خطی

```
function [ out ] = Bilinearinter( x,y,f,o )
% Example :
% out  = Bilinearinter( [1 5],[3 8],[10 15 17 2],[4 6])
Q11 = f(1);
Q12 = f(2);
Q21 = f(3);
Q22 = f(4);

xl = x(1);
xh = x(2);
yl = y(1);
yh = y(2);
x = o(1);
y=o(2);

out  = (Q11 * (xh-x) *(yh-y)) + (Q12 * (xh-x) * (y-yl)) + (Q21 * (x-xl) *
(yh-y)) + (Q22 * (x-xl) * (y-yl)) ;

end
```