Assignment 2: Transforming Images

1. **transformImage.m**: This contains code for transforming images

function TransformedImage = transformImage(InputImage,TransformMatrix,TransformType)

%{

transformImage - Transform a grayscale image according to the given transformation.

Syntax:

TransformedImage = transformImage(InputImage, TransformMatrix, TransformType)

Description:

transformImage method applies a transformation to a grayscale image based on the given transformation matrix and type.

Inputs:

InputImage - The grayscale input image.

TransformMatrix - The transformation matrix defining the transformation to be applied.

TransformType - The type of transformation. It can be one of the following: 'scaling', 'rotation', 'translation', 'reflection', 'shear', 'affine', or 'homography'.

Outputs:

TransformedImage - The transformed image.

Example:

% Apply a scaling transformation to the input image

InputImage = imread('input.png');

TransformMatrix = [2, 0, 0; 0, 2, 0; 0, 0, 1]; % Example scaling matrix

TransformType = 'scaling';

TransformedImage = transformImage(InputImage, TransformMatrix, TransformType);

%}

% switch case to pick the right matrix depending on TransformType

switch lower(TransformType)

case 'homography'

A=TransformMatrix;

Ainv=inv(A);

case 'affine'

A=TransformMatrix;

Ainv=inv(A);

case 'scaling'

A=TransformMatrix;

Ainv=A;

Ainv(1,1)=1/Ainv(1,1);

Ainv(2,2)=1/Ainv(2,2);

case 'rotation'

A=TransformMatrix;

Ainv=A;

Ainv(2)=-Ainv(2);

Ainv(4)=-Ainv(4);

case 'reflection'

A=TransformMatrix;

Ainv=A.\*(-1);

case 'shear'

A=TransformMatrix;

Ainv=A;

Ainv(1,2) = -Ainv(1,2);

case 'composite'

A3=TransformMatrix{1};

A2=TransformMatrix{2};

A1=TransformMatrix{3};

A=A3\*A2\*A1;

% inverse of translation matrix A1

A1inv=A1;

A1inv(1,3)=-A1inv(1,3);

A1inv(2,3)=-A1inv(2,3);

% inverse of rotation matrix A2

A2inv=A2;

A2inv(2)=-A2inv(2);

A2inv(4)=-A2inv(4);

% inverse of scaling matrix A3

A3inv=A3;

A3inv(1,1)=1/A3inv(1,1);

A3inv(2,2)=1/A3inv(2,2);

Ainv=A1inv\*A2inv\*A3inv;

% Ainv=inv(A);

end

% reading input image and normalising it followed by grayscaling image

inputDouble = im2double(InputImage);

grayImage = rgb2gray(inputDouble);

[h,w] = size(grayImage);

% cordinates of input image frame

a = [1,1,1]';

b = [w,1,1]';

c = [1,h,1]';

d = [w,h,1]';

% finding new cordinates for transformed image via TransformMatrix

c1=A\*a;

c2=A\*b;

c3=A\*c;

c4=A\*d;

% 1/what is applied to all homogeneous coordinates

x1=c1(1)/c1(3);

y1=c1(2)/c1(3);

x2=c2(1)/c2(3);

y2=c2(2)/c2(3);

x3=c3(1)/c3(3);

y3=c3(2)/c3(3);

x4=c4(1)/c4(3);

y4=c4(2)/c4(3);

% bounding box dimensions

Aprime = [min([1,x1,x2,x3,x4]),min([1,y1,y2,y3,y4])];

Bprime = [min([1,x1,x2,x3,x4]),max([y1,y2,y3,y4])];

Cprime = [max([x1,x2,x3,x4]),min([1,y1,y2,y3,y4])];

Dprime = [max([x1,x2,x3,x4]),max([y1,y2,y3,y4])];

minx = Aprime(1);

miny = Aprime(2);

maxx = Dprime(1);

maxy = Dprime(2);

[Xprime,Yprime] = meshgrid( minx:maxx, miny:maxy );

% resulting image dimensions

[hprime,wprime] = size(Xprime);

pprime = [Xprime(:)';Yprime(:)';ones(1,wprime\*hprime)];

% disp(hprime);disp(wprime);

phat = Ainv \* pprime;

xhat = phat(1,:);

yhat = phat(2,:);

what = phat(3,:);

x = xhat ./ what;

y = yhat ./ what;

x = reshape( x', hprime, wprime );

y = reshape( y', hprime, wprime );

% output image

TransformedImage = interp2( grayImage, x, y );

end

1. transforms\_script.m: This contains code running transforms by calling transformImage method

clear all; close all; clc;

% reading input image and normalising it followed by grayscaling image

inputImage=imread('Image1.png');

inputDouble = im2double(inputImage);

grayImage = rgb2gray(inputDouble);

[h,w] = size(grayImage);

target\_width = 1920;

target\_height = 1080;

Sx = target\_width / w;

Sy = target\_height / h;

% matrices for transformations

scalingMatrix=[Sx 0 0;0 Sy 0;0 0 1];

rotationMatrix=[cosd(30),-sind(30),0;sind(30),cosd(30),0;0 0 1];

reflectionMatrix=[1 0 0; 0 -1 0; 0 0 1];

shearMatrix=[1 0.5 0;0 1 0;0 0 1];

affineMatrix1=[1 0.4 0.4;0.1 1 0.3;0 0 1];

affineMatrix2=[2.1 -0.35 -0.1;-0.3 0.7 0.3;0 0 1];

homographyMatrix1=[0.8 0.2 0.3;-0.1 0.9 -0.1;0.0005 -0.0005 1];

homographyMatrix2=[29.25 13.95 20.25;4.95 35.55 9.45;0.045 0.09 45];

%matrices for 5th part of question

A1=[1 0 300;0 1 500;0 0 1];

A2=[cosd(-20),-sind(-20),0;sind(-20),cosd(-20),0;0 0 1];

A3=[0.5 0 0;0 0.5 0;0 0 1];

compositeMatrix={A3,A2,A1};

%Output of images after transformations

outputImageOfScaling=transformImage(inputImage,scalingMatrix,'scaling');

outputImageOfReflection=transformImage(inputImage,reflectionMatrix,'reflection');

outputImageOfRotation=transformImage(inputImage,rotationMatrix,'rotation');

outputImageOfShear=transformImage(inputImage,shearMatrix,'shear');

outputImageOfAffine1=transformImage(inputImage,affineMatrix1,'affine');

outputImageOfAffine2=transformImage(inputImage,affineMatrix2,'affine');

outputImageOfHomography1=transformImage(inputImage,homographyMatrix1,'homography');

outputImageOfHomography2=transformImage(inputImage,homographyMatrix2,'homography');

outputImageOfCompositeOperations=transformImage(inputImage,compositeMatrix,'composite');

%plotting of output images

figure;imshow(grayImage);title('Input Image in Gray Scale');

figure;imshow(outputImageOfScaling);title('Output Image 1');

figure;imshow(outputImageOfReflection);title('Output Image 2');

figure;imshow(outputImageOfRotation);title('Output Image 3');

figure;imshow(outputImageOfShear);title('Output Image 4');

figure;imshow(outputImageOfCompositeOperations);title('Output Image 5');

figure;imshow(outputImageOfAffine1);title('Output Image 6.1');

figure;imshow(outputImageOfAffine2);title('Output Image 6.2');

figure;imshow(outputImageOfHomography1);title('Output Image 7.1');

figure;imshow(outputImageOfHomography2);title('Output Image 7.2');

1. Output of Transforms on Image1

Each Image contains titles representing the answer to questions in same order

A plant next to train tracks

Description automatically generated

Close-up of a plant next to a train track

Description automatically generated

A close-up of a train track

Description automatically generated

A close-up of a plant growing on a train track

Description automatically generated

A close-up of a rail road

Description automatically generated

A close-up of a train track

Description automatically generated

A black and white image of a planet

Description automatically generated

A plant growing on a train track

Description automatically generated

A close-up of a plant

Description automatically generated

1. Output of Transforms on Image2

Each Image contains titles representing the answer to questions in same order

A room with a piano and chairs

Description automatically generated

A room with a round bathtub

Description automatically generated

A room with a table and chairs

Description automatically generated

A room with a table and chairs

Description automatically generated

A room with a table and chairs

Description automatically generated

A room with a rug and chairs

Description automatically generated

A building with a few windows

Description automatically generated with medium confidence

A room with a table and chairs

Description automatically generated

A room with a table and chairs

Description automatically generated