Joanne Kwon

PID #: A15359545

**Part I**

**QUESTION #1**

* If the value of *i* inside the body of the loop is changed to 5:10, then the program outputs i=5, i=6, i=7, i=8, i=9, i=10. In this example, the 5 correlates with the start of the loop and the 10 correlates with where the loop ends. This case was similar in different experiments.

**QUESTION #2**

* The effect that was created is a gradual change from image2 (dayman.jpg) to image1 (nightman.jpg). The image changes from left to right until the full image of image2 replaces image1.

Code:

for col = 0: size (image1, 1)

imshow([image1(:,1:col,:) image2(:,col+1:end,:)]);

drawnow;

end

**QUESTION #3**

Code:

for row = 0: size (image1, 1)

imshow([image1(1:row,:,:); image2(row+1:end,:,:)]);

drawnow;

end

**QUESTION #4**

Code:

function [newmatrix] = dissolve(image1, image2)

% Name: Joane Kwon

% This function takes two images and transitions over different "blendings"

% of the images.

% The dissolve function uses a for loop to repeat the various blends of image1

% and image2.

for p = 0:100

imshow(((100-p)/100)\*image1 + ((100-p)/100)\*image2);

drawnow;

end

end

**QUESTION #5**

Code:

function [newmatrix] = dissolve(image1, image2)

% Name: Joane Kwon

% This function takes two images and transitions over different "blendings"

% of the images.

% The dissolve function uses a for loop to repeat the various blends of image1

% and image2.

for p = 0:100

imshow(((p)/100)\*image1 + ((p)/100)\*image2);

drawnow;

end

end

**QUESTION #6**

* By changing the value, x, in 0:x:100, the speed of dissolve is changed. When the value of the numeric number is closer to zero (ex: 0.1) the images dissolve at a slower speed. On the other hand, when the value of the numeric value is higher (ex: 5) the images dissolve at a faster pace. This is because the value from 0 is being increased by the value of x all the way to 100.

**QUESTION #7**

Code:

function [newmatrix] = dissolve(image1, image2)

% Name: Joane Kwon

% This function takes two images and transitions over different "blendings"

% of the images.

% The dissolve function uses a for loop to repeat the various blends of image1

% and image2.

for p = 100:0

imshow(((100-p)/100)\*image1 + ((100-p)/100)\*image2);

drawnow;

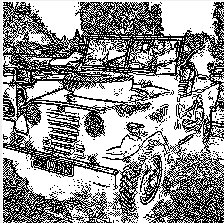
end

end

**Part II**

** **

[threshold = 30]

** **

[threshold = 30]

**IsEdge**

Code:

function [edge] = IsEdge(bw, row, col, threshold)

% Name: Joanne Kwon

% PID: A15359545

% The function 'IsEdge' takes in four arguments to find the edges of an image. This function returns true if the pixel at (row, col) is considered an edge pixel and false otherwise.

% This function takes in the arguments bw, row, col, and threshold. bw is

% a matrix that represents a black and white image. The arguments row and

% col are the row and column of the pixel that the function checks for an

% edge. The argument threshold is a value between 0 and 225 that is used

% to tune the sensitivity of the criteria for what counts as an edge. A

% higher threshold value indicates that fewer edges are detected and a

% threshold of 0 indicates everything as an edge.

if ((row-1 > 0 && row+1 < size(bw,1)) && (col-1 > 0 && col+1 < size(bw,2))) %checks if row and column in range

%check boundary

if (threshold <= (bw(row-1, col) - bw(row, col)))

edge = 1;

elseif(threshold <= (bw(row+1, col) - bw(row, col)))

edge = 1;

elseif (threshold <= (bw(row, col+1) - bw(row, col)))

edge = 1;

elseif (threshold <= (bw(row, col-1) - bw(row, col)))

edge = 1;

else

edge = 0;

end

else

edge = 0;

end

end

**FindEdges**

Code:

function [output] = FindEdges(image, threshold)

% Joanne Kwon

% PID: A15359545

% The function 'FindEdges' takes two arguments and returns a black and

% white image that shows the edges of the input image. The image is the

% same size as the input image but only with one layer.

% This function takes in the arguments image and threshold. The argument

% image is a matrix that represents an RGB image in the usual RGB image

% format. The argument image is a value between 0 and 255 for tuning the

% sensitivity of out criteria for what counts as an edge.

bw = rgb2gray(image);

blank = zeros(size(bw));

%imshow(blank);

for i = 1:size(image,1)

for j= 1:size(image,2)

if IsEdge(bw, i, j, threshold) == 1

blank (i,j,1) = 0;

else

blank (i,j) = 255;

end

end

end

output = blank;

size(blank)

imshow(output);

end