Introdução ao Processamento de Imagem – Turma A

Projeto 1

*David Gonçalves Mendes  
Departamento de Ciência da Computação - UnB*

*Universidade de Brasília (UnB)  
Brasília, Brasil  
davidgmendes@hotmail.com*

***Será apresentado a seguir o passo-a-passo da resolução do primeiro trabalho da matéria de Introdução ao processamento de Imagem – UnB (2020.1), o qual envolve redimensionamento de imagens e a alteração da quantidade de pixeis de uma imagem.***

# Introdução

O projeto proposto tem o objetico de fixar os conceitos quanto ao domínio espacial e filtragem do domínio da frequência e de morfologia matemática. O trabalho foi realizado utilizando-se da ferramenta open source *Octave*.

Na maior parte do projeto foi realizado o acesso pixel a pixel de matriz afim de manipular a imagem passada. Foi utilizado o conceito da média dos vizinhos de 4 no intuito de criar novos pixels no caso de redimensionar uma imagem para uma imagem maior. É importante ressaltar que todas as imagens manipuladas e utilizadas durantes os testes foram imagens de 8bits, que consistem em imagens com 256 níveis de brilho variando de 0 (mais escuro) até 255 (mais claro). Também foi utilizado do conceito da quantização para alterar o número de níveis de brilho de imagens sem realmente alterar o número de bits da imagem.

Nas próximas seções serão detalhados os algoritimos utilizados para a resolução dos exercícios propostos(2), serão apresentados os resultados de testes realizados(3) além das conclusões finais(4).

# Metodologia

## Alteração da quantidade de níveis de brilho

Para esse algorítimo, foi gerado, a partir de parâmetro recebido (indicando a nova quantidade de bits da imagem, variando de 1 a 8), um array de tamanho 256, que seria usado para quantizar o valor de cada nível de brilho de um pixel da seguinte forma: cada posição representa o nível de brilho do pixel original e, o valor do array naquela posição, representará o novo nível de brilho do pixel após a quantização. Para gerar o array quantizado, calculou-se o número de valores iguais que deveriam haver para cada quantidade de bits(2^(8 – n) e o intervalo entre os difertentes numero de bits, valendo 255/2^n – 1, sendo n o novo número de bits a ser assumido pela imagem. Por fim comparou-se esse array com cada pixel da matriz da imagem.

## *Redimensionamento de Imagem*

Para esse algorítimo, foi criada uma Nova matriz para armazenar os valores dos pixeis da nova imagem redimensionada, e foi verificado o número de linhas e colunas da nova imagem. Em seguida, foi verificado se o novo tamanho da imagem seria menor ou maior do que a original.

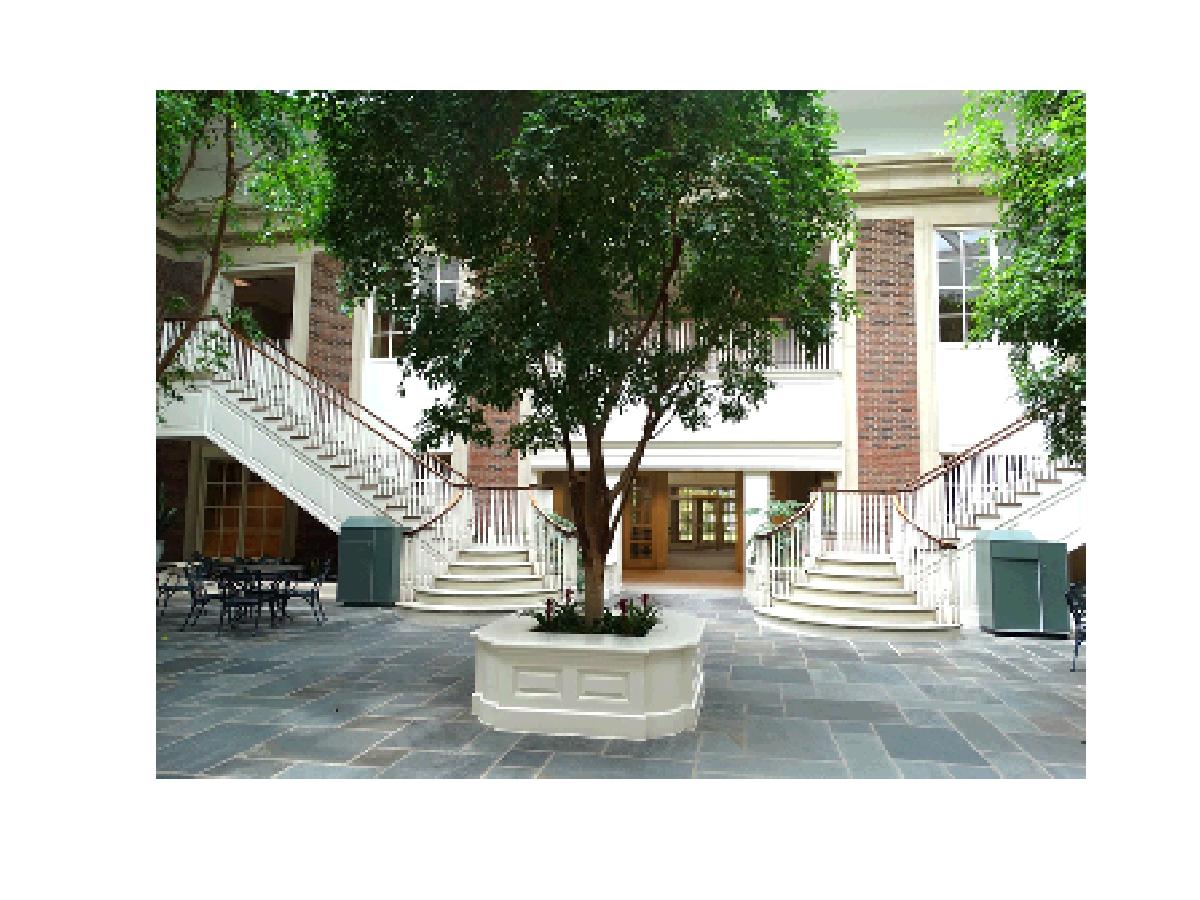
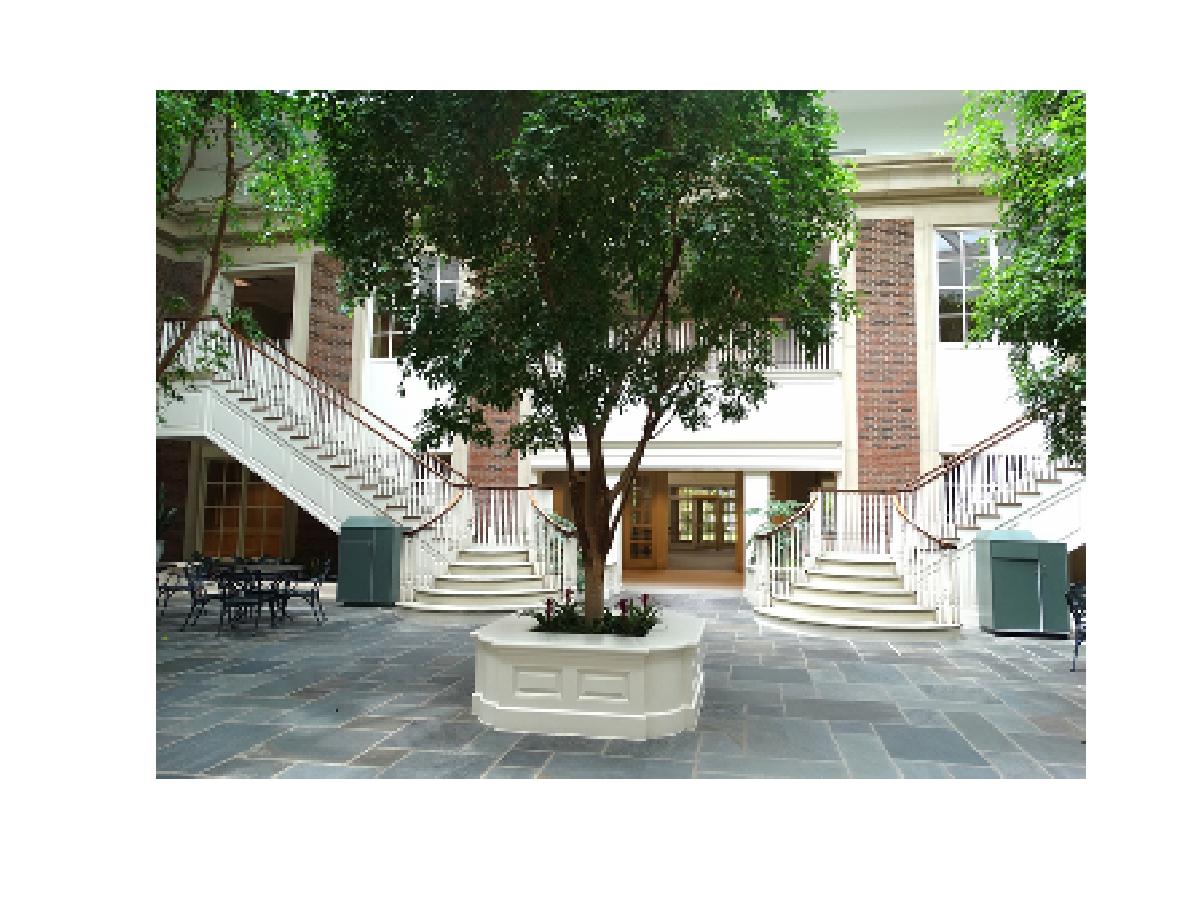
Imagem menor (tamanho < 1): O valor de cada pixel(x, y) da nova imagem deveria ser igual ao valor do pixel (x – 1, y – 1) multiplicado pelo inverso do tamanho e somado 1.

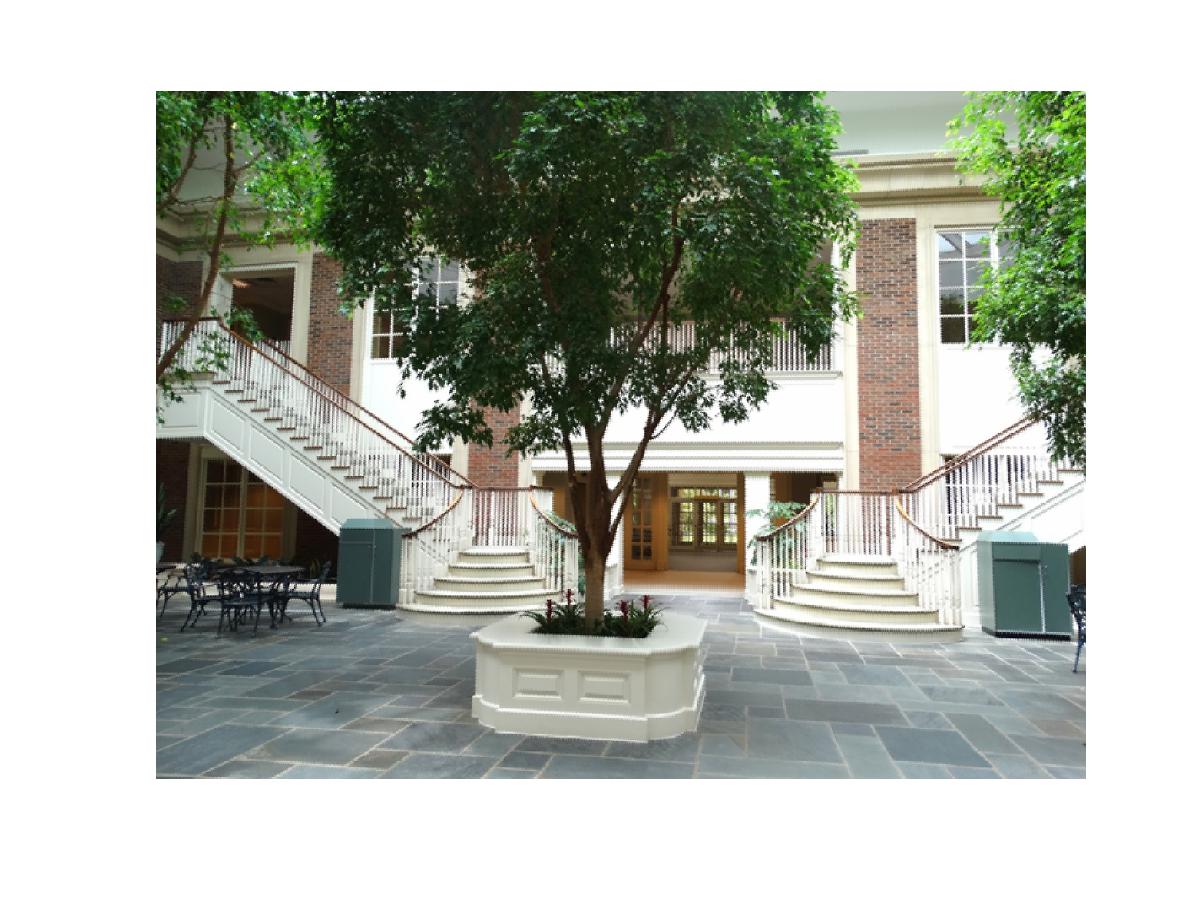
Imagem maior (tamanho > 1) : atribuiu-se -1 ao valor de cada pixel da nova imagem, em seguida, foi notado que o valor na posicao (x, y) da posicao original iria ser o valor da posicao (tamanho\*(x -1) + 1, tamanho\*(y – 1) + 1) da nova imagem e portanto esses valores foram alterados na nova matriz. Por fim foi verificado que enquanto houvesse algum pixel assumindo o valor de -1 na matriz da nova imagem, o valor deste seria substituído pelo valor da média seus vizinhos de 4, caso existissem e fossem diferentes de -1.

# Resultados

Os algorítimos citados acima foram testados em conjunto com a seguinte imagem de base:

As seguintes imagens foram obtidas na saída, para os valores de, respectivamente (8, 0.5), (5, 0.5), (8, 1.75) e (3, 1.75).







# Conclusões

Pelos resultados é possível perceber que para valores menores de tamanhos, a matriz fica mais pixelada, porém para valores maiores ela fica com uma quantidade maior de pixeis, sendo a diferença ao olho quase imperceptível.

Além do mais, é possível perceber que para valores menores de bits, a imagem perde um pouco de contraste, uma vez que a quantidade de cores diminui.