MC940 – Processamento de Imagens
Trabalho 03

Rafael Timbó Matos
RA 106228

Execução

O programa pode ser chamado diretamente, em que a ordem dos parâmetros é importante, ou podemos utilizar a interface *shell*, que recebe os parâmetros no formato `optarg`. Parâmetros *default* são assumidos para quando estes não forem passados.

```
./lab04.m [angle scale width height method input output]
octave lab04.m [angle scale width height method input output]
./lab04.sh [-a angulo | -s escala | -e escala ]
           [-w largura]
           [-h altura]
           [-m metodo]
           [-i imagem_entrada]
           [-o imagem_saida]
```

As imagens deverão estar no mesmo diretório do programa. Utilizamos as imagens *house.ppm, monarch.ppm, baboon.ppm* para testes, que se encontram no diretório. Os testes utilizados são os comandos contidos no arquivo *batch.sh*.

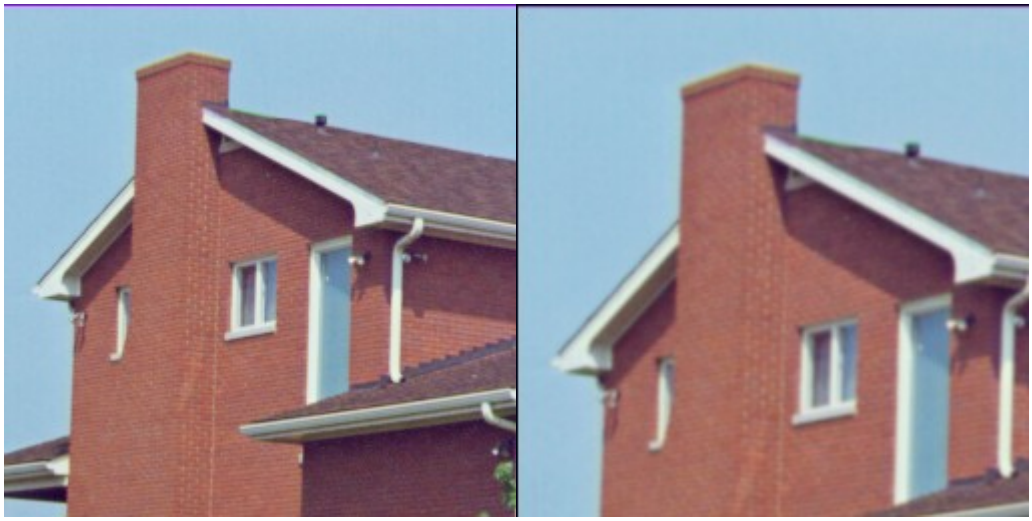
Algoritmo e Estrutura de Dados

O trabalho foi programado de forma não iterativa. Para tal, construimos vetores coluna que contem todos os indices (x,y) que usaremos para acessar os pixeis na matriz da imagem. Esses indices podem sofrer rotação ou mudança de escala, dependendo dos parametros passados. Em seguida, aplicamos o método de interpolação requerido, fazendo as operações em forma de matrizes para melhorar a performance. Verificamos quais indices ficarão fora dos limites da matriz de imagens, o que significa um pixel fora dos limites da imagem original, e os substituimos por um plano de fundo preto. Tais pixeis são indicados pela matriz booleana *outbound*. Antes de acessar a matriz da imagem por indice, trocamos todos os indices invalidos pelo limite mais proximo(1, nr ou nc), garantindo que não haverá erros ao tentar acessar a imagem, evitando verificações de invalidez. Como construímos os indíces para um canal só, replicamos o vetor coluna de indices 3 vezes, um para cada canal de cor. Finalmente, realizamos as operações requeridas por cada interpolação, considerando as imagens como vetores colunas para que as operações possam ser executadas utilizando os vetores coluna de indices. A função *sub2ind* retorna o indice linear do pixel que queremos da imagem original. Então, a imagem de saida é gerada por uma única operação com vetores colunas que representam os coeficientes que precisamos e os pixeis. Para terminar, reinterpretamos a matriz de saida com as dimensoes adequadas.

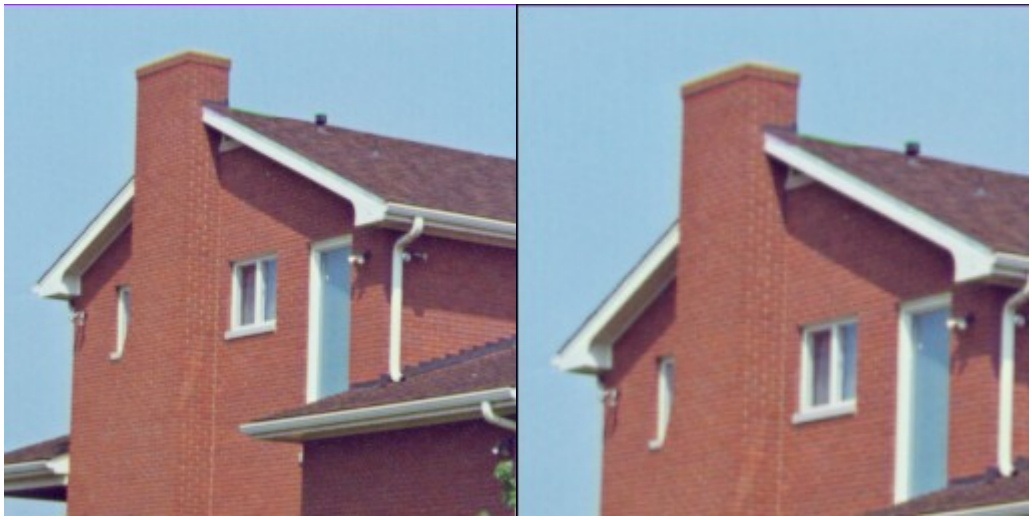
Saídas

Notamos que todos os métodos apresentam resultados satisfatórios, com exceção do método de Lagrange para casos muito pequenos. Os métodos de interpolação bilinear e bicubica suavizam(blur) levemente a imagem, enquanto o método do vizinho mais próximo produz um efeito serrilhado nas bordas. O método de Lagrange produz linhas verticais e horizontais formando um padrão xadrez na imagem. Os traços se tornam desprezíveis com uma imagem suficientemente grande. O programa considera as dimensões de entrada como de saída caso esta não seja fornecida. Os comandos a seguir produziram as respectivas imagens, com a entrada à esquerda e a saída a direita:

```
./lab04.sh -s 1.25 -i house.ppm -o house1.25bicubica.ppm  -m bicubica
```

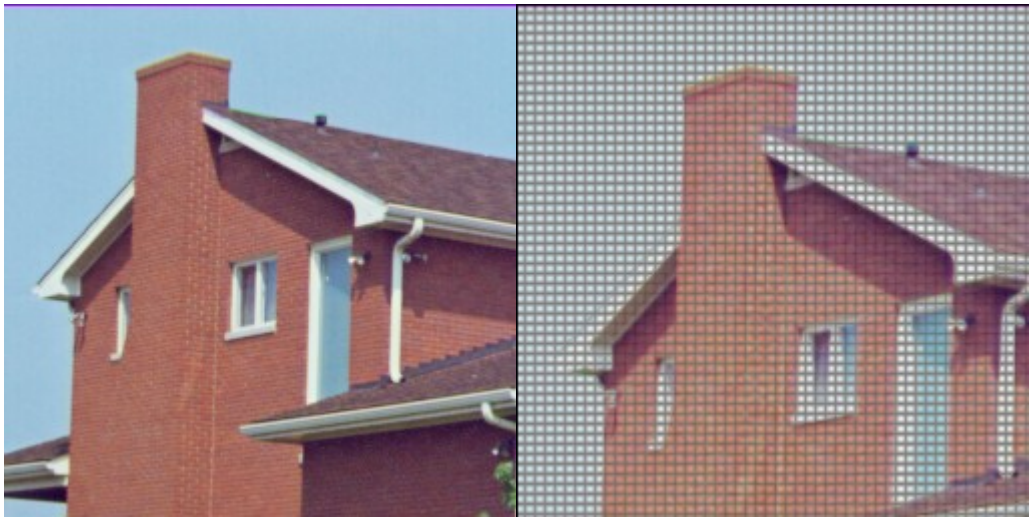


```
./lab04.sh -s 1.25 -i house.ppm -o house1.25bilinear.ppm  -m bilinear
```



```
./lab04.sh -s 1.25 -i house.ppm -o house1.25vizinho.ppm  -m vizinho
```

```
./lab04.sh -s 1.25 -i house.ppm -o house1.25lagrange.ppm  -m lagrange
```



```
./lab04.sh -a 100 -i baboon.ppm -o baboon100anglevizinho.ppm  -m vizinho
```

Observe o leve serrilhamento no olho do baboon, devido ao método dos vizinhos.

```
./lab04.sh –a 45 –i baboon.ppm –o baboon45anglebilinear.ppm  –m bilinear
```

```
./lab04.sh -m lagrange -s 4 -w 3072 -h 2048 -i monarch.ppm -o monarch4x.png
```

Com uma dimensão de 2048x3072, os traços xadrez são quase imperceptíveis.