

# MC920 - Trabalho 4

Tobias Conran Zorzetto, RA: 166214

April 2023

## 1 Introdução

Esse trabalho tem como objetivo aplicar diferentes transformações espaciais e geométricas a uma imagem, aplicando técnicas de registro de imagem.

Assim, o relatório será dividido em 3 seções: **Introdução**, **Projeção Perspectiva**, onde será aplicada uma projeção perspectiva a uma imagem e analisado o seu resultado e método de resolução, e **Transformações Geométricas**, onde serão aplicadas diversas transformações geométricas, analisadas e comparados os seus resultados.

## 2 Projeção Perspectiva

A projeção perspectiva realiza a transformação de pontos para o plano de uma imagem. Essa técnica foi realizada no programa de nome `projecaoperspectiva.py`, na qual tem objetivo de aplicar uma projeção perspectiva sobre a imagem da figura 1.

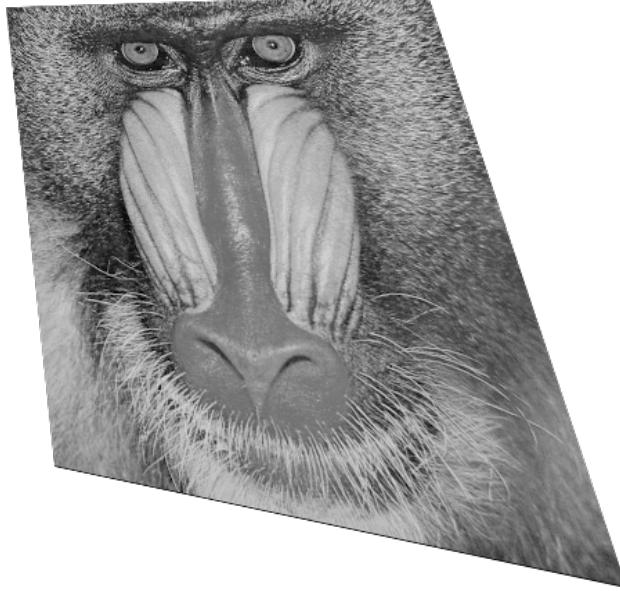


Figura 1: Imagem a ser transformada por Projeção Perspectiva

Nessa projeção, os pontos (37, 51), (342, 42), (485, 467), (73, 380) devem ser levados respectivamente para a (0, 0), (511, 0), (511, 511), (0, 511). Isso pode ser feito a partir da matriz da transformação afim que pode ser escrita como:

$$X' = \frac{aX + bY + c}{iX + jY + 1} \quad (1)$$

$$Y' = \frac{dX + eY + f}{iX + jY + 1} \quad (2)$$

Em que  $X'$  e  $Y'$  são as posições destino do pixel que antes estava em  $X$  e  $Y$ . Porém, como para o programa queremos obter a transformação indireta, para que possamos percorrer por cada pixel da nova imagem e achar seu pixel correspondente, no lugar de percorrer pelos pixels da imagem original, o que pode levar a falhas na imagem resultante por arredondamentos. Para fazer a transformação indireta, basta considerar  $X'$  e  $Y'$  como as posições originais e  $X$  e  $Y$  como as posições destino. Assim, como existem 8 variáveis e 8 equações possíveis, esse sistema é solucionado, chegando assim, às transformações:

$$X' = \frac{0,47X + 0,042Y + 37}{-3,6 \times 10^{-4}X - 3,8 \times 10^{-4}Y + 1} \quad (3)$$

$$Y' = \frac{-0,033X + 0,50Y + 51}{-3,6 \times 10^{-4}X - 3,8 \times 10^{-4}Y + 1} \quad (4)$$

Aplicando essa transformação para cada pixel da nova imagem, tem-se o resultado da figura 2, que mostra a imagem planificada sobre o espaço determinado, como na imagem conhecida utilizada nos outros projetos da matéria e em aula. Um fator que vale ser discutido é que a imagem transformada de volta a sua forma conhecida não parece ter perdido muita definição, pelo menos a olho humano.

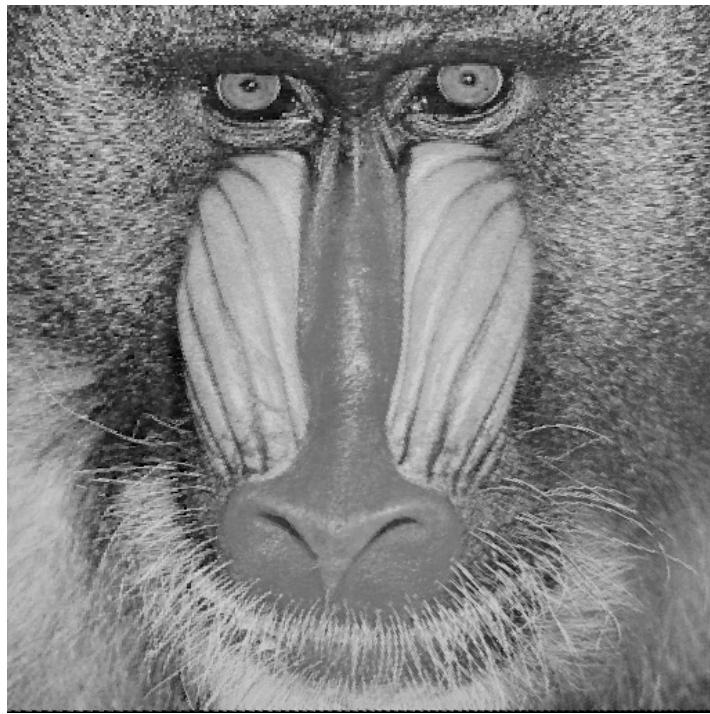


Figura 2: Projeção Perspectiva resultante da imagem da figura 1

### 3 Transformações Geométricas

Para a demonstração da aplicação de diferentes técnicas de transformações em uma imagem, foi realizado um programa, com nome `transformacoes_geometricas.py`. Esse programa recebe como argumentos obrigatórios:

- `--i` a url da imagem original que será utilizada para interpolação
- `--o` a url de destino da imagem de saída

Assim, Ele pode realizar uma transformação geométrica por execução. Essa transformação pode ser uma rotação ou a aplicação de uma redimensionalidade da imagem.

Para a rotação da imagem, o usuário deve apenas passar mais um argumento `--a` que define o ângulo dessa rotação.

Para a redimensionalidade da imagem o usuário pode escolher fazer isso através de um fator de escala (argumento passado como `--e`), que vai aumentar ou diminuir a imagem conforme esse fator, ou ele pode fazer isso passando as dimensões exatas, assim passará a altura e largura da imagem através do argumento `--d "largura altura"`. Nos dois casos também deve ser passado a técnica de interpolação desejada, com o argumento `--m`, podendo ser ela "vizinhoMaisProximo", "bilinear", "bicubica" e "lange".

Para essa seção, foi utilizada a mesma imagem, que pode ser vista na figura 7, como entrada em cada execução do programa para fim de melhor comparação entre as técnicas.



Figura 3: house.png

Dado isso, pode-se ser feita uma análise das técnicas utilizadas para rotação e interpolação e seus resultados aplicados em execuções do programa.

### 3.1 Interpolação

Para realização das técnicas de interpolação, cada função recebeu os fatores de escala para as dimensões X e Y. Com isso, pode ser feita a definição de uma nova imagem em preto com as dimensões requeridas por essas escalas dada as dimensões da imagem original.

Assim, basta a aplicação da expressão de interpolação na imagem. É importante dizer que essa expressão deve ser feita sendo a função definida o pixel em uma posição específica da nova imagem, com base nos pixels da imagem original. Assim, pode-se ser feito um loop em cada pixel da imagem de saída para que ela seja definida por inteira. Caso seja feito o contrário, ou seja, faça-se um loop nos

pixels da imagem original para projeção na nova imagem, a imagem de saída não terá todos os pixels definidos devido a arredondamentos que levam a projeções em mesmos pixels na imagem de saída.

Além disso, para evitar que se tente acessar um pixel fora do tamanho da imagem, como pode acontecer ao tentar acessar as vizinhanças de um pixel da imagem original que está perto da borda, as funções lidam com isso escolhendo a posição mínima entre a última posição naquela direção e a posição que a interpolação aponta que se deve ler a intensidade.

Assim, podemos apresentar cada uma das técnicas de interpolação utilizadas: Interpolação pelo vizinho mais próximo, bilinear, bicúbica e interpolação por polinômios de lagrange. Para fins de comparação, cada um desses métodos será executado para o mesmo objetivo: utilizando um fator de escala de 3 para altura e largura.

A **interpolação pelo vizinho mais próximo** consiste no cálculo da posição de cada pixel da nova saída e com isso o arredondamento para a posição inteira mais próxima. O resultado desse método nas condições estabelecidas pode ser encontrado na figura 4. A partir de um zoom (figura 5), vê-se que a imagem não parece ser mais definida que a original mesmo, tendo uma escala de pixels 3 vezes maior. As linhas diagonais, assim como qualquer detalhe na imagem parem estar pixelados, e quadrados, como se uma conjunto grande de pixels agisse como a mesma informação na imagem.

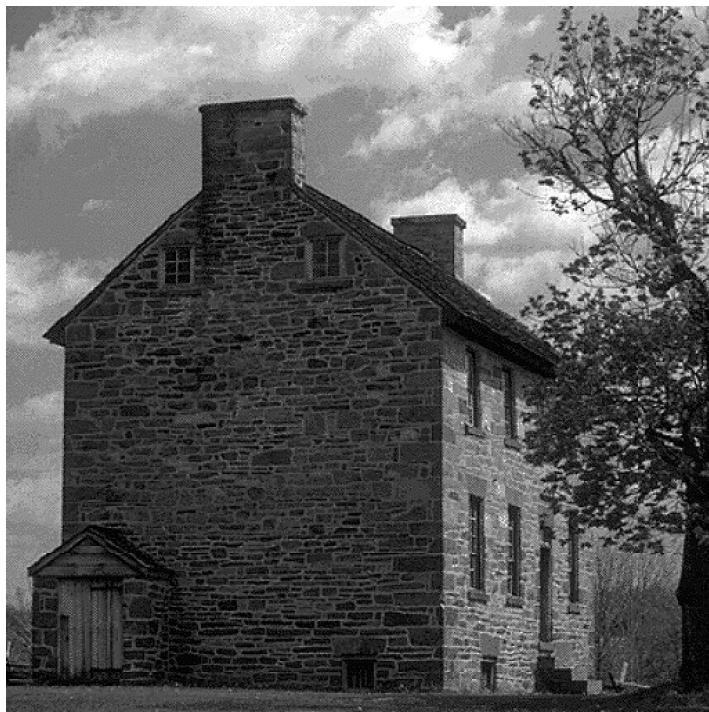


Figura 4: Aplicação do metodo de interpolação do vizinho mais próximo para um fator de escala de 3



Figura 5: zoom da figura 4

A **interpolação bilinear** consiste na média ponderada da distância dos quatro pixels vizinhos mais próximos do pixel calculado. Diferentemente do método anterior, essa técnica já cria valores de intensidade novos para pixels, no lugar de utilizar uma intensidade já existente em um dos pixels próximos, o que pode suavizar melhor a transição entre pixels e regiões da imagem. O resultado disso pode ser visto na figura 6 . Com o mesmo zoom dado à imagem anterior, na figura ?? podemos ver que algumas características como as linhas e bordas aparecem mais suavizadas e definidas. Apesar disso, o efeito negativo da interpolação é mais evidente em regiões em que a localidade dos 4 pixels vizinhos não é suficiente, como por exemplo na disposição das pedras que compõe a casa e seus preenchimentos, que perdem detalhes, parecem ter pixels que distoam da cor a sua volta e transições muito bruscas entre tons.

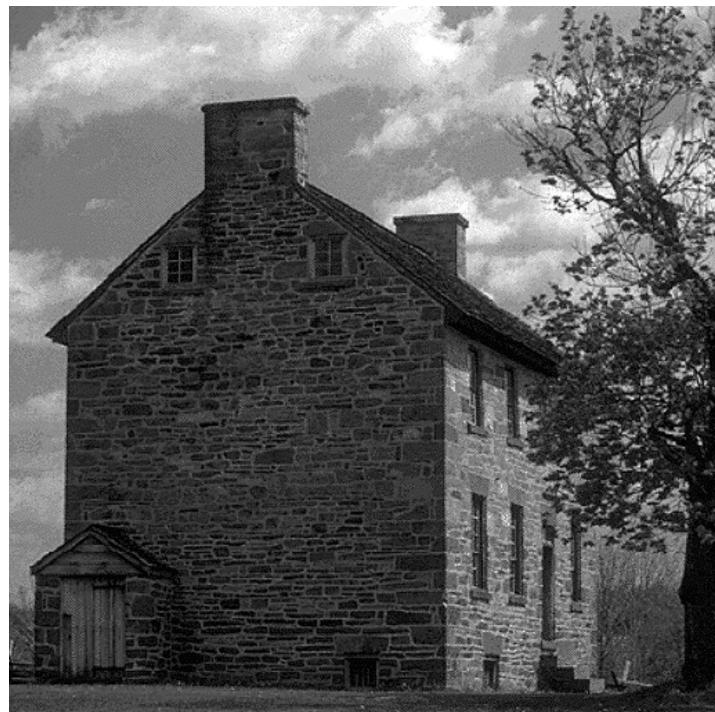


Figura 6: Aplicação do metodo de interpolação bilinear para um fator de escala de 3



Figura 7: zoom da figura 6

A **interpolação bicúbica** utiliza uma vizinhança muito maior que a bilinear, uma vizinhança 4x4. Além disso além de uma simples média ponderada, ela utiliza uma função B-spline cúbica para suavização da imagem. O resultado disso pode ser visto na figura 8. O mesmo zoom dado nas outras figuras pode ser visto na figura 9. A partir da análise das imagens, vê-se que esse método lida muito melhor com localidades e com a suavização da imagem no geral, removendo de forma muito efetiva as mudanças bruscas em tonalidades que outros métodos tinham, o que ao olho humano parece tornar a imagem muito melhor definida. Apesar disso, essa suavização pode levar a perda de detalhes onde essas mudanças em tonalidade realmente acontecem.

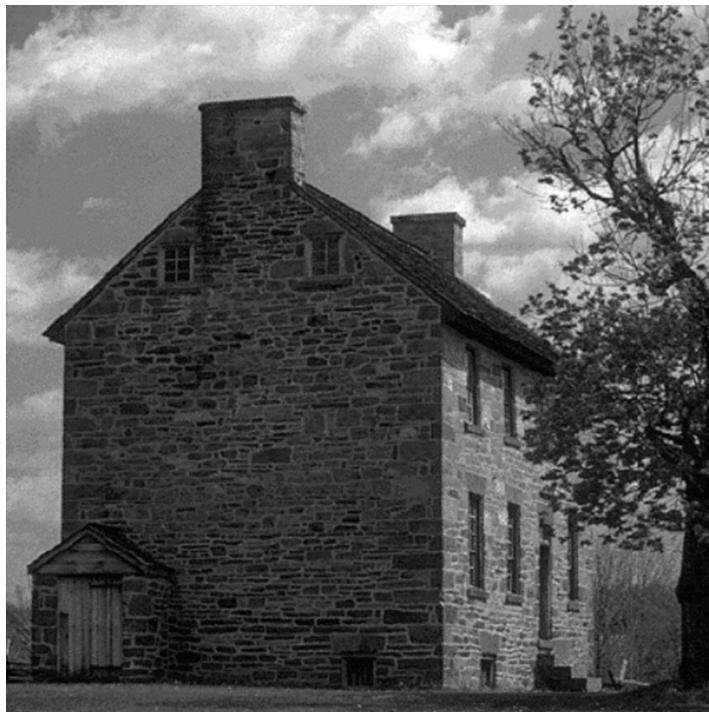


Figura 8: Aplicação do metodo de interpolação bicúbica para um fator de escala de 3



Figura 9: zoom da figura 8

Por fim, a **interpolação por polinômios de Lagrange** também utiliza uma vizinhança  $4 \times 4$ . Porém a função utilizada para o cálculo de cada uma das itensidades dos pixels da nova imagem é a função dos polinômios de Lagrange. O resultado da aplicação dessa função em cada pixel da imagem de saída pode ser visto na figura 10. O mesmo zoom feito para as outras figuras pode ser visto na figura 11. Pode-se analisar que o resultado dessa interpolação traz uma grande definição à imagem, com linhas e bordas bem definidas. Esse método parece lidar bem com localidades e definição de detalhes combinado com uma definição e suavização da imagem, sem perda aparente perda de informação. Sendo assim, o método que obteve o melhor resultado, apesar de ser o mais custoso computacionalmente.

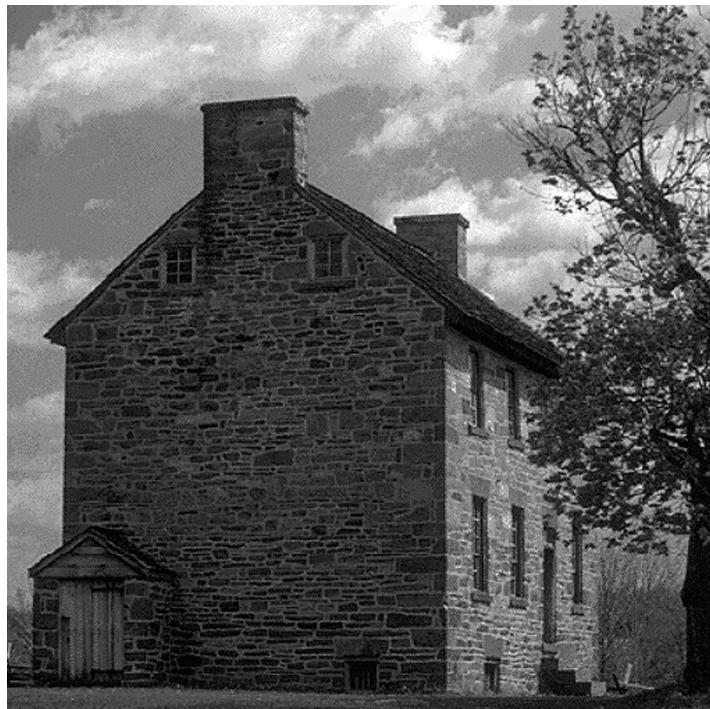


Figura 10: Aplicação do metodo de interpolação por polinômios de Lagrange para um fator de escala de 3

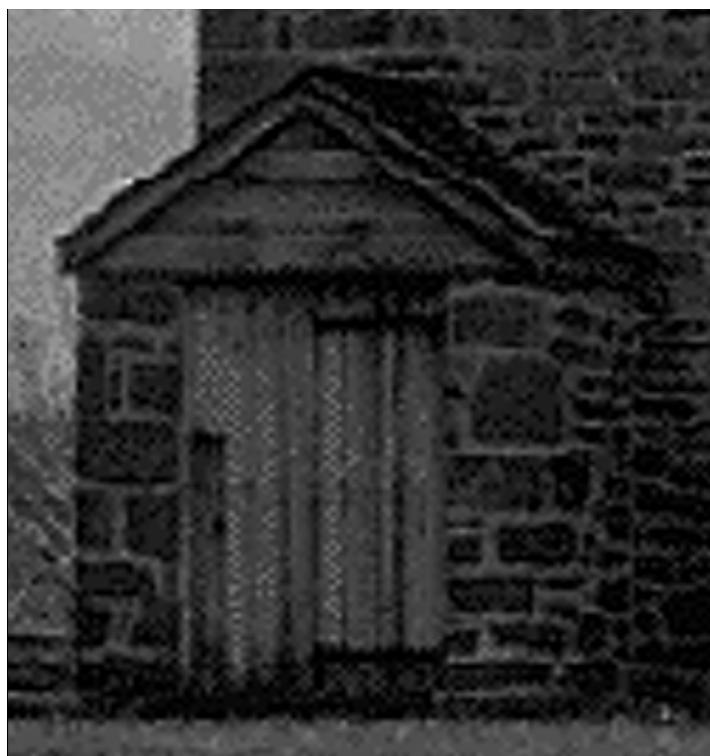


Figura 11: zoom da figura 10

### 3.2 Redimensionalidade

Além da mudança no fator de escala como um todo, que permite uma mudança nas dimensões de forma congruente, o programa, como citado anteriormente, permite a escolha das dimensões de altura e largura da imagens de forma arbitrária. Assim, os métodos de interpolação funcionam para a alteração dos aspectos da imagem, sem que o código precise ser alterado, já que as funções já recebem separadamente as escalas devidas para as direções  $x$  e  $y$ . Com isso, na figura 12 abaixo podemos ver a utilização da interpolação por polinômios de lagrange para gerar a mesma imagem original porém com aspecto 1920x1080 (widescreen HD).

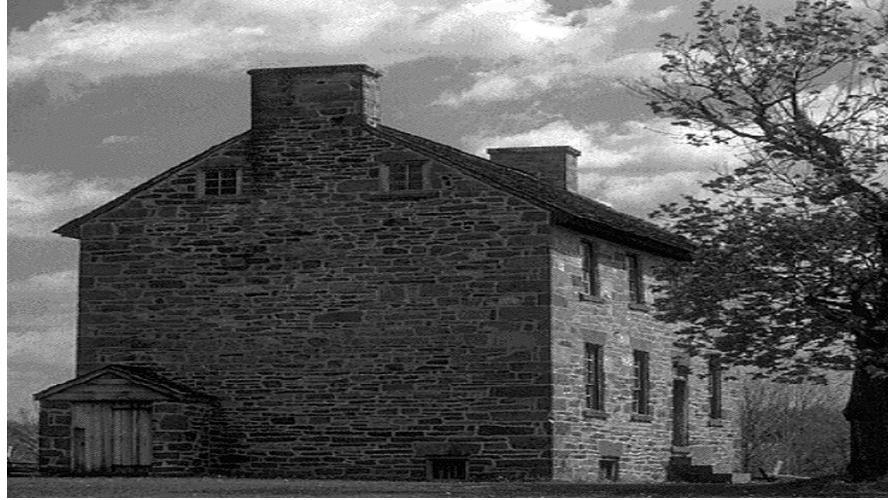


Figura 12: utilização da interpolação por polinômios de Lagrange para redimensionalidade 1920x1080

### 3.3 Rotação

Por fim, também pode ser feita a rotação de uma imagem dentro do programa. Para a produção de uma imagem de saída sendo ela rotacionada em uma certa quantidade de graus, basta fazer uma função parecida com as funções de interpolação. A função recebe como parâmetros a imagem e o grau de rotação. Assim, deve ser criada uma nova imagem em preto com as mesmas dimensões da imagem original.

Com isso faz-se um loop em cada ponto da imagem antiga, utilizando a inversa da matriz de rotação para calcular de que posições de  $x$  e  $y$  da imagem original será copiada a intensidade para a posição de pixel observada da nova imagem. Um ponto diferente das interpolações é que como a matriz de rotação utiliza apenas a posição do pixel calculado, no lugar de uma vizinhança, caso a posição calculada para um pixel seja fora do seu domínio, o pixel respectivo da nova imagem apenas será preto.

Assim, podemos ver na figura 13 a aplicação dessa função de rotação para um ângulo de 45.5 graus. Analisando a imagem, pode-se ver que a imagem tem uma definição parecida com a original, com algumas perdas devido a arredondamentos no cálculo utilizando a matriz de rotação. Além disso, vê-se a perda de alguns cantos da imagem visto que a dimensão utilizada igual a da imagem original não é capaz de conter a imagem toda. Por final, pode-se ver os pixels pretos, que como explicados anteriormente, não representam espaço para representação da imagem original rotacionada.

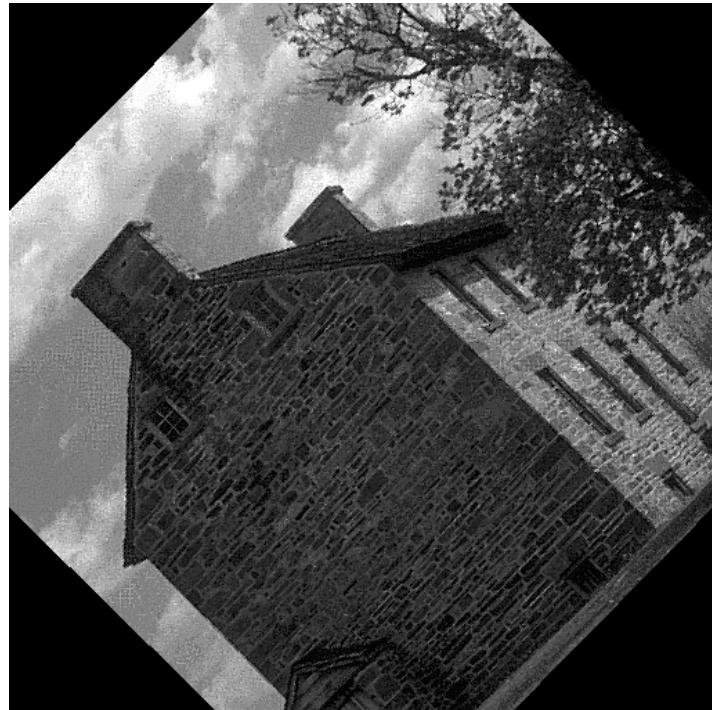


Figura 13: rotação de 45.5 graus da imagem original 7