

Aplicando projeções perspectiva e geométrica (trabalho 4)

Introdução ao Processamento de Imagem Digital

Randerson A. Lemos (103897) 2022-1S

1 Introdução

Encontrar a correspondência entre pixels de diferentes imagens que capturam a mesma informação de ângulos, posições e perspectivas diferentes é uma problema muito comum abordado pelo processamento de imagens. Esses problemas são genericamente denominados de problemas de Registro e sua resolução normalmente se desenvolve pelo uso dos conhecimento de álgebra linear e suas ideias de espaços vetoriais e transformações entre tais espaços por meio de matrizes de transformação. Para exemplificar em termos práticos os problemas de registro, observe a Figura 1 a seguir retirada do material de aula.

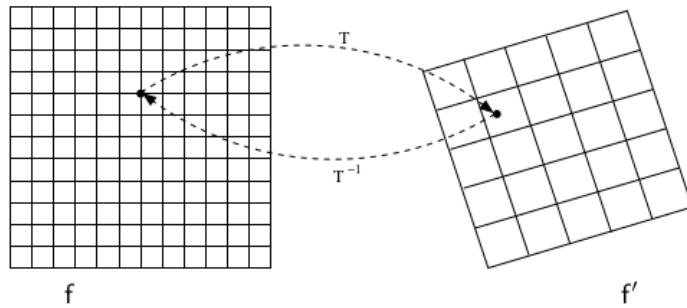


Figura 1: Aplicação da matriz de transformação \mathbf{T} entre as imagens \mathbf{f} e \mathbf{f}' (**fonte**: material de aula).

Nesse figura, temos duas imagens \mathbf{f} e \mathbf{f}' cujo pixels estão relacionados pela matriz de transformação \mathbf{T} , ou seja

$$\mathbf{f}' = \mathbf{T}\mathbf{f}. \quad (1)$$

Na Equação 1, a matriz de transformação \mathbf{T} está fazendo o mapeamento da imagem \mathbf{f} para a imagem \mathbf{f}' . Tal mapeamento pode ser representado como $\mathbf{T} : \mathbf{f} \mapsto \mathbf{f}'$. Um aspecto importante que precisa ser levado em conta nos problemas de registro é da natureza discreta das informações presentes, isto é: das imagens digitais. Devido a essa natureza, o mapeamento direto $\mathbf{T} : \mathbf{f} \mapsto \mathbf{f}'$ nem sempre vai conduzir um pixel válido da imagem de origem \mathbf{f} para um pixel válido da imagem de destino \mathbf{f}' , uma vez que depois de aplicada a matriz de transformação não há garantias que o pixel transformado seja discreto e esteja contido dentro do domínio da imagem de destino \mathbf{f}' . Para contornar esse aspecto prático da aplicação das matrizes de transformação em domínios discretos, a solução encontrada foi fazer uso da transformação inversa e de métodos de interpolação, isto é:

$$\mathbf{f} = \mathbf{T}^{-1}\mathbf{f}'. \quad (2)$$

A aplicação da transformação inversa garante que todos os pixels da imagem de destino \mathbf{f}' sejam considerados no problema de encontrar seu correspondente na imagem de origem \mathbf{f} e a aplicação das técnicas de interpolação garante que todos os pixels transformados estejam associados a algum pixel válido da imagem de origem \mathbf{f} . Existem diversas técnicas de interpolação, sendo algumas das mais conhecidas as seguintes: Interpolação pelo Vizinho Mais Próximo, Interpolação Bilinear, Interpolação Bicúbica e Interpolação por Polinômios de Lagrange. Essas interpolações são amplamente discutidas no material da disciplina como também no documento de instruções desse trabalho.

Os problemas de registro são principalmente organizados em problemas de transformações geométricas e problemas de transformações projetivas. Ambos esses problemas serão discutidos em mais detalhes a seguir.

1.1 Transformações geométricas

Os problemas de transformação geométricas consistem de uma transformação espacial e uma interpolação. A transformação espacial está associada as transformações afins de rotação, escala, translações, espelhamento e

cisalhamento. Nessas transformações, o paralelismo das linhas e curvas da imagem transformada é preservado, mas dimensões lineares, de área ou volumétricas não. Em coordenadas homogêneas e considerando o caso bidimensional, essas transformações podem ser generalizadas por

$$\begin{bmatrix} X' \\ Y' \\ W \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} X \\ Y \\ W \end{bmatrix}. \quad (3)$$

Cada uma das transformações de rotação, escala, translação, espelhamento e cisalhamento terão uma matriz de transformação específica (a matriz com os coeficientes a, b, \dots, i). A seguir, vamos conhecer a forma das matrizes das transformações de escala, translação e rotação.

1.1.1 Transformação de Escala

A transformação de escala para o caso 2D pode ser realizada a partir da matriz de transformação em coordenadas homogêneas apresenta a seguir.

$$\begin{bmatrix} X' \\ Y' \\ W \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 \\ 0 & S_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ W \end{bmatrix}. \quad (4)$$

Então, para aplicar um aumento ou redução de escala da imagem, o usuário precisa definir o valor de escala S_x a ser aplicado ao longo do eixo-x e o valor de escala S_y a ser aplicado ao longo do eixo-y.

1.1.2 Transformação de Translação

A transformação de translação para o caso 2D pode ser realizada a partir da matriz de transformação em coordenadas homogêneas apresenta a seguir.

$$\begin{bmatrix} X' \\ Y' \\ W \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ W \end{bmatrix}. \quad (5)$$

Então, para aplicar uma translação na imagem, o usuário precisa definir o valor de translação t_x a ser apicado ao longo do eixo-x e o valor de translação t_y a ser aplicado ao longo do eixo-y.

1.1.3 Transformação de Rotação

A transformação de rotação para o caso 2D pode ser realizada a partir da matriz de transformação em coordenadas homogêneas apresenta a seguir.

$$\begin{bmatrix} X' \\ Y' \\ W \end{bmatrix} = \begin{bmatrix} \cos(\alpha) & -\sin(\alpha) & 0 \\ \sin(\alpha) & \cos(\alpha) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ W \end{bmatrix}. \quad (6)$$

Então, para aplicar uma rotação na imagem, o usuário precisa definir o valor de rotação α a ser aplicado na imagem.

1.2 Transformações projetivas

As transformações projetivas estão associadas ao processo de projeção de pontos dispostos tridimensionalmente no espaço em um plano bidimensional. Esses transformações são principalmente do tipo ortográfica ou perspectiva. A projeção ortográfica é uma transformação que realiza o mapeamento de pontos tridimensionais sobre o plano da imagem, tal que os pontos são projetados ao longo de linhas paralelas na imagem (informação extraída do material de aula). Já as projeções perspectivas, apesar de haver o mesmo mapeamento de um espaço tridimensional para um espaço bidimensional (um plano), o tamanho do objecto projeto assim como algumas razões de suas medidas lineares e de área são dependentes do aumento da distância do centro de projeção. Essa dependência está associada a percepção de profundida do sistema visual humano. A ideia das transformações projetivas está apresentada na Figura 2.

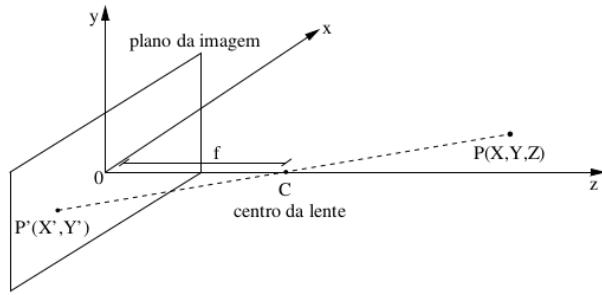


Figura 2: Modelagem geométrica de transformação projetiva (**fonte**: material de aula).

Existem algumas maneiras de se calcular a matriz de transformação de uma transformação projetiva. No caso particular das projeções perspectivas, uma das formas de obter tal matriz é a partir de quatro pontos não colineares e a resolução das seguintes equações

$$X' = \frac{aX + bY + c}{iX + jY + 1} \quad Y' = \frac{fX + eY + f}{iX + jY + 1} \quad (7)$$

considerando-se os quatro pontos não colineares em questão. Os valores obtidos da resolução do sistema de equações para as variáveis a, b, c, d, e, i, j são os coeficientes da matriz de transformação

$$\begin{bmatrix} a & b & c \\ d & e & f \\ i & j & 1 \end{bmatrix}$$

da projeção perspectiva em questão.

2 Objetivo

O objetivo deste documento é o de fornecer uma relatório sobre as técnicas de registro solicitadas no documento de instrução do trabalho 4. De acordo com as instruções, é solicitado que seja desenvolvido um programa capaz de realizar as transformações espaciais de escala e rotação. Também é solicitado a implementação de um programa que realize uma projeção perspectiva sobre a imagem **baboon_perspectiva_result.png**.

3 Solução

A solução utiliza a linguagem de programação Python e conta com o auxílio do gerenciador de projetos e pacotes Conda. Assumindo que o usuário tenha o Conda instalado em sua máquina, a configuração do projeto pode ser feita pela execução do comando `conda env create -f environment.yml` a partir da pasta do projeto **trab4**. Esse comando cria o ambiente de trabalho **mc920-trab4** e instala os seguintes módulos: opencv, numpy, scipy, pandas, matplotlib. Finalizada a configuração do ambiente de trabalho em questão, o usuário deve executar o comando `source source.sh`¹ para carregar as variáveis de ambiente adequadas e, assim, poder usar os programas do projeto dentro do próprio ambiente de trabalho recém configurado.

Dos arquivos presentes na pasta do projeto **trab4**, destacam-se as pastas **assets** e **tex** e os programas **apply_perspective.py** e **main.py**. A pasta **assets** contém imagens no formato png que podem ser utilizadas para a aplicação das técnicas de registro consideradas. A pasta **tex** contém os arquivos Latex deste relatório. O programa **main.py** contém as implementações necessárias para aplicar as técnicas de registro implementadas na imagem de entrada fornecida pelo usuário. Informações pertinentes das implementações serão apresentadas a seguir.

3.1 Apply_perspective.py

O programa **apply_perspective.py** é responsável pela aplicação da projeção perspectiva requerida no trabalho. Para rodar tal programa, o usuário deve executar o seguinte comando:

```
1 python3 apply_perspective.py
```

Depois de executado o programa **apply_perspective.py**, a imagem transformada é salva na pasta **out** com nome adequado e alto explicativo.

¹O comando que configura o ambiente de trabalho mc920-trab4 precisa ser executado apenas uma vez. Assim sendo, depois que este ambiente está configurado, o usuário precisa apenas executar o comando `source source.sh`

3.1.1 Detalhes de implementação

A aplicação da transformação perspectiva é realizada com o auxílio da função `getPerspectiveTransform` disponibilizada pelo módulo do **opencv**.

3.2 Main.py

O programa **main.py** é responsável pela aplicação de todas as transformações espaciais implementadas. Este programa recebe diferentes parâmetros de entrada e pode ser executado de diferentes maneiras. Para executar transformações de escala com a escolha de uma determinada forma de interpolação, o usuário pode usar alguns dos comandos de exemplo a seguir:

```
1 python3 main.py -imagem_entrada assets/baboon.png -e 1.25 -m lagrange
2 python3 main.py -imagem_entrada assets/baboon.png -e 1.25 -m bicubic
3 python3 main.py -imagem_entrada assets/baboon.png -e 1.25 -m bilinear
4 python3 main.py -imagem_entrada assets/baboon.png -e 1.25 -m nearneighbours
```

Para executar a transformação de escala por meio da definição dos tamanhos de largura e comprimento, o usuário pode usar alguns dos comandos exemplo a seguir:

```
1 python3 main.py -imagem_entrada assets/baboon.png -d 300x400 -m lagrange
2 python3 main.py -imagem_entrada assets/baboon.png -d 300x400 -m bicubic
3 python3 main.py -imagem_entrada assets/baboon.png -d 300x400 -m bilinear
4 python3 main.py -imagem_entrada assets/baboon.png -d 300x400 -m nearneighbours
```

Para executar transformações de rotação com a escolha de uma determinada forma de interpolação, o usuário pode usar alguns dos coandos de exemplo a seguir (a unidade considerada para o angulo de rotação é graus):

```
1 python3 main.py -imagem_entrada assets/baboon.png -a 1.25 -m lagrange
2 python3 main.py -imagem_entrada assets/baboon.png -a 1.25 -m bicubic
3 python3 main.py -imagem_entrada assets/baboon.png -a 1.25 -m bilinear
4 python3 main.py -imagem_entrada assets/baboon.png -a 1.25 -m nearneighbours
```

Em todas as suas diferentes formas de execução, o programa **main.py** salva na pasta **out** a imagem transformada com nome adequado e auto explicativo de modo a permitir ao usuário associar as imagens salvas com as transformações realizadas sobre elas. O programa **main.py** dispões das seguintes classes **Interpolation**, **Scale** e **Rotation**. São os grupos de códigos dessas classes que encapsulam as lógicas das interpolações e das transformações de escala e rotação implementadas (logo, não só as implementações das técnicas de interpolação foram feitas, mas também as implementações das técnincias de rotação e escala). Maior enfoque será dado a essas classes na sequênciia.

3.3 Classe Scale

A classe **Scale** é responsável pela aplicação da transformação de escala sobre a imagem de entrada. A escala a ser aplicada na imagem de entrada é fornecida pelo usuário e pode ser um fator multiplicativo ou os valores das dimensões finais de largura e altura. Para o primeiro caso o usuário, deve fornecer para o programa principal o parâmetro **-e** seguido do valor de escala desejado. Para o segundo caso, o usuário deve fornecer o parâmetro de entrada **-d** seguido das dimensões finais de saída no formado **Largura x Altura**. Para conseguir aplicar a transformação de escala, essa classe precisa receber uma função capaz de realizar um método de interpolação. Essas funções estão agrupadas na classe **Interpolation** e serão discutidas mais adiante. A classe **Scale** retorna a matriz transformada de acordo com o fator de escala fornecido. Essa classe é capaz de realizar transformações de escala considerando valores flutuantes.

3.4 Classe Rotation

A classe **Rotation** é resposável pela aplicação da transformação de rotação sobre a imagem de entrada. A rotação a ser aplicada na imagem de entrada é fornecida pelo usuário pelo parâmetro **-a** seguido do valor de rotação desejado. Para conseguir aplicar a transformação de rotação, essa classe precisa receber uma função capaz de realizar um método de interpolação. A classe **Rotation** retorna a matriz de transformação de acordo com a fator de rotação fornecido. É imporante salientar que essa classe sempre considera como origem, para aplicação da transformação de rotação, o ponto (0,0) da imagem. Com isso, uma rotação, por exemplo de 30 graus, vai rotacionar a imagem em sentido anti-horário de modo que os pixels que avancem para além do domínio da imagem são descartados. Logo, uma rotação de 45 graus, resulta na perda de 50% dos valores dos pixels da imagem de entrada e um rotação de 90 graus, resulta na perda de 100% desses pixels. Essa classe é capaz de realizar transformações de rotação considerando valores flutuantes. Os valores de entrada para essa classe são em graus.

3.5 Classe Interpolation

A classe **Interpolation** é responsável por fornecer algumas das funções de interpolação implementadas para as classes **Scale** e **Rotation**. Por meio das funções dessas classes, é possível utilizar as seguintes interpolações: Interpolação pelo Vizinho Mais Próximo, Interpolação Bilinear, Interpolação Bicúbica e Interpolação por Polinômios de Lagrange (os detalhes dessas interpolações estão presentes no material da disciplina). Um ponto em comum dessas interpolações é que todas retorna o valor de nível de cinza igual a zero para todos os pixels que caem fora do domínio da imagem de origem.

4 Resultado

Os resultados apresentados estão organizados em dois grupos: o primeiro grupo apresenta o resultado gerado pela aplicação da projeção perspectiva requerida no trabalho. O segundo grupo apresenta o resultado gerado pela aplicação de transformações geométricas.

4.1 Projeção perspectiva

A projeção perspectiva dos pontos $(37, 51)$, $(342, 42)$, $(485, 467)$, $(73, 380)$ para $(0, 0)$, $(511, 0)$, $(511, 511)$, $(0, 511)$ aplicada na imagem de entrada da Figura 3

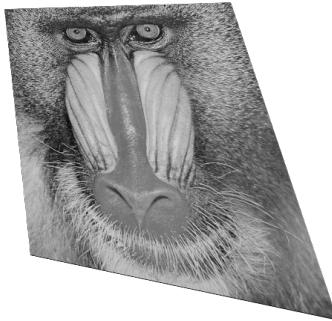


Figura 3: Imagem de entrada para aplicação de projeção perspectiva.

resulta na imagem da Figura 4 a seguir:

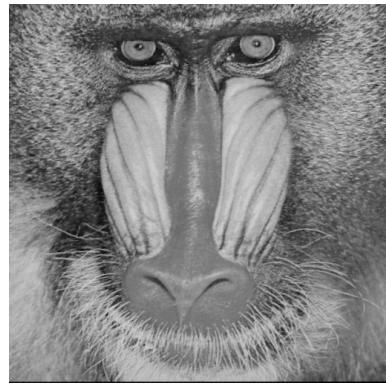


Figura 4: Imagem resultante da aplicação da projeção perspectiva requerida no trabalho.

Podemos ver que a aplicação da projeção perspectiva, resultou no formato original da conhecida imagem **baboon**.

4.2 Transformação geométrica

A seguir apresentaremos o resultado da aplicação das transformações geométricas considerando diferentes técnicas de interpolação. Todas as transformações foram aplicadas sobre a imagem da Figura 5.

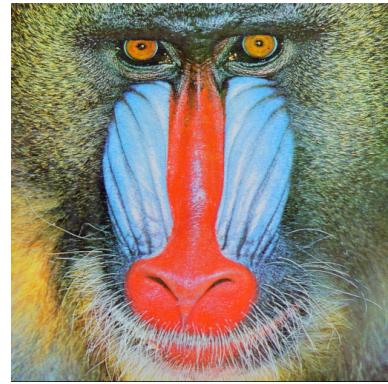
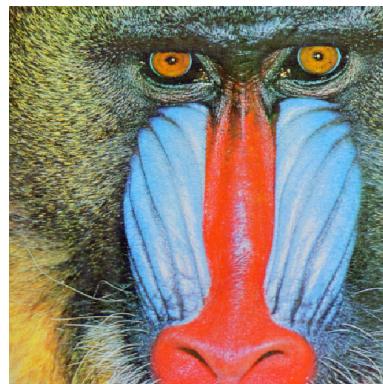


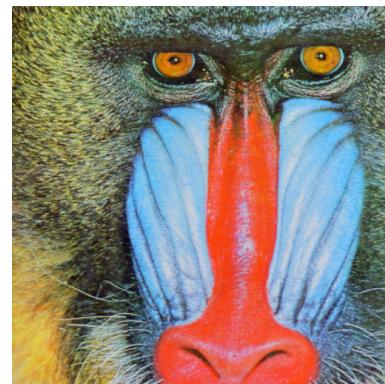
Figura 5: Imagem de entrada para aplicação de transformações geométricas.

4.2.1 Transformação de escala usando o parâmetro -e

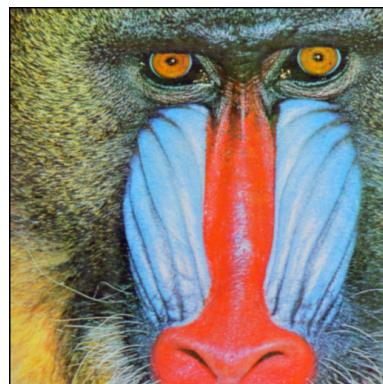
As imagens apresentadas da Figura 6 foram obtidas pela aplicação de um fator de escala de 1,25 considerando as diferentes técnicas de interpolação requeridas.



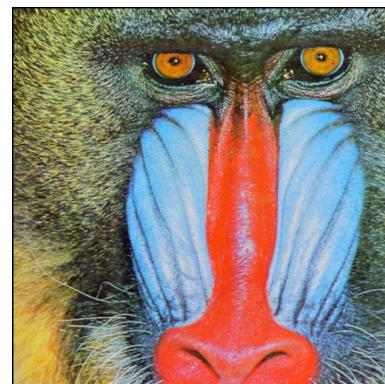
(a) Imagem com nova escala com Interpolação do Vizinho Mais Próximo.



(b) Imagem com nova escala com Interpolação Bilinear.



(c) Imagem com nova escala com Interpolação Bicubica.

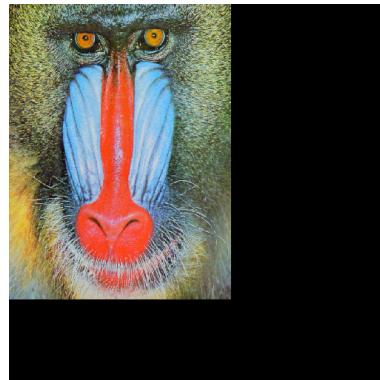


(d) Imagem com nova escala com Interpolação de Lagrange.

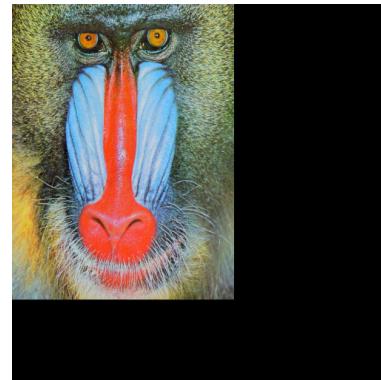
Figura 6: Imagens com fator de escala de 1,25 para diferentes métodos de interpolação.

4.2.2 Transformação de escala usando o parâmetro -d

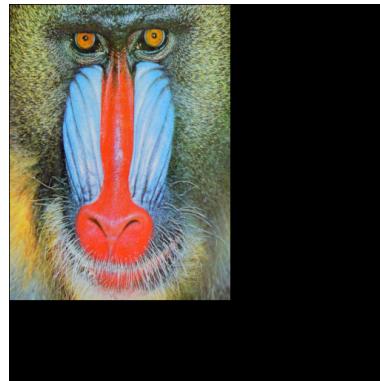
As imagens apresentadas da Figura 7 foram obtidas pela aplicação de valores finais de largura 300 e de altura 400 considerando as diferentes técnicas de interpolação requeridas.



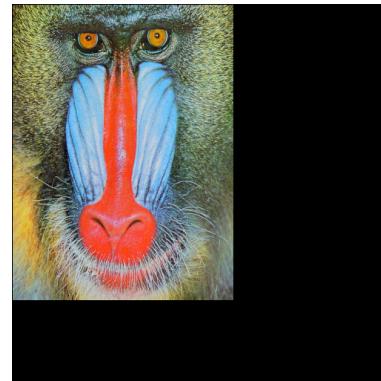
(a) Imagem com nova escala com Interpolação do Vizinho Mais Próximo.



(b) Imagem com nova escala com Interpolação Bilinear.



(c) Imagem com nova escala com Interpolação Bicubica.



(d) Imagem com nova escala com Interpolação de Lagrange.

Figura 7: Imagens com valores finais de largura 300 e de altura 400 para diferentes métodos de interpolação.

4.2.3 Transformação de rotação

As imagens apresentadas da Figura 8 foram obtidas pela aplicação do valor de rotação de 1,25 graus considerando as diferentes tecnicas de interpolação requeridas.

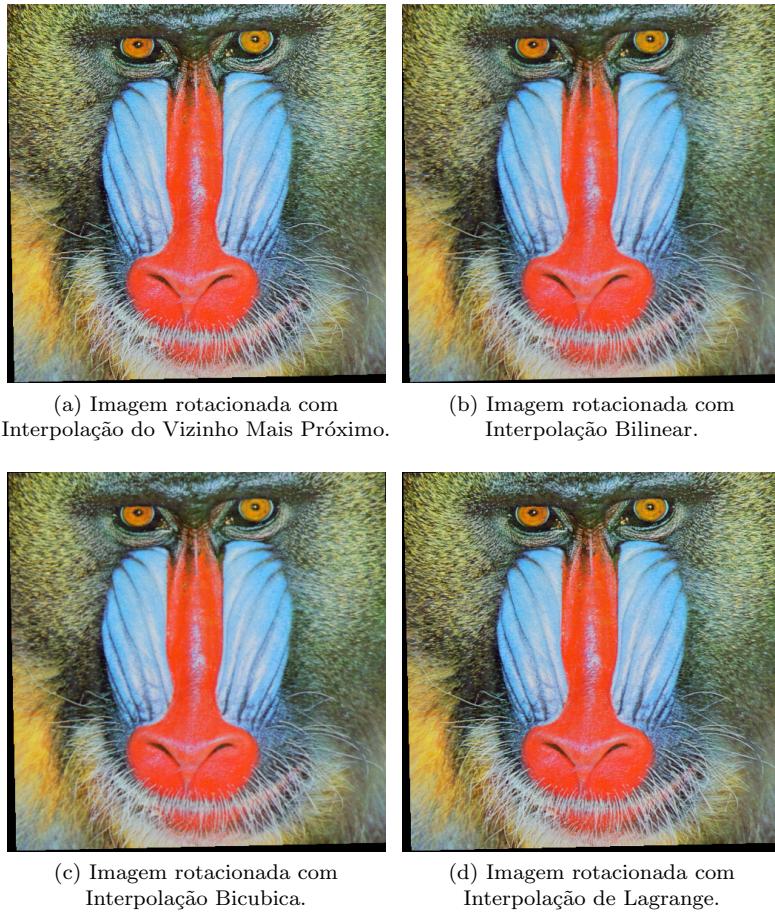


Figura 8: Imagens rotacionadas em 1.25 graus para diferentes métodos de interpolação.