Aceleração de Registro Não-Rígido de Imagens em GPU

Thiago de Gouveia Nunes
Orientador: Prof. Marcel P. Jackowski

IME - USP

12 de fevereiro de 2015

O registro é o processo de alinhamento espacial ou geométrico entre duas ou mais imagens. Ele é uma etapa fundamental em uma série de aplicações:

- A criação de imagens panorâmicas
- A fusão de informações (imagens de diferentes modalidades)
- Correção de movimentação em imagens médicas

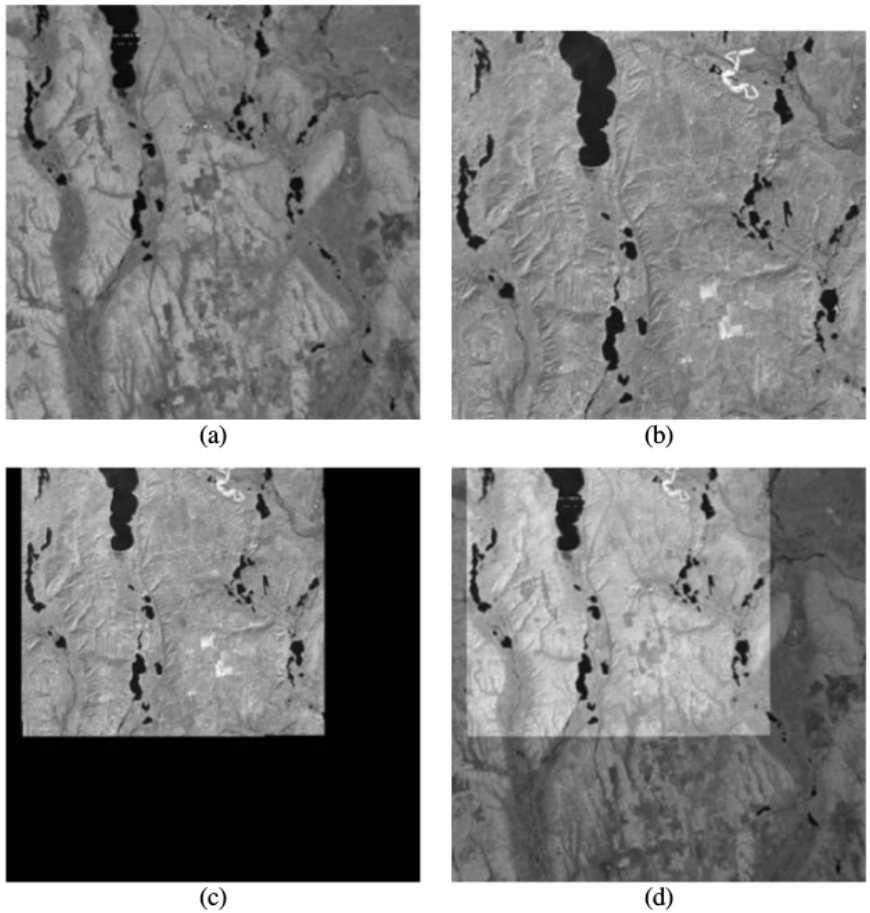

Fig. 1.1 (a) A Landsat MSS image used as the reference image. (b) A Landsat TM image used as the sensed image. (c) Resampling of the sensed image to register the reference image. (d) Overlaying of the reference and resampled sensed images.

Source: http://en.wikipedia.org/wiki/Image_registration

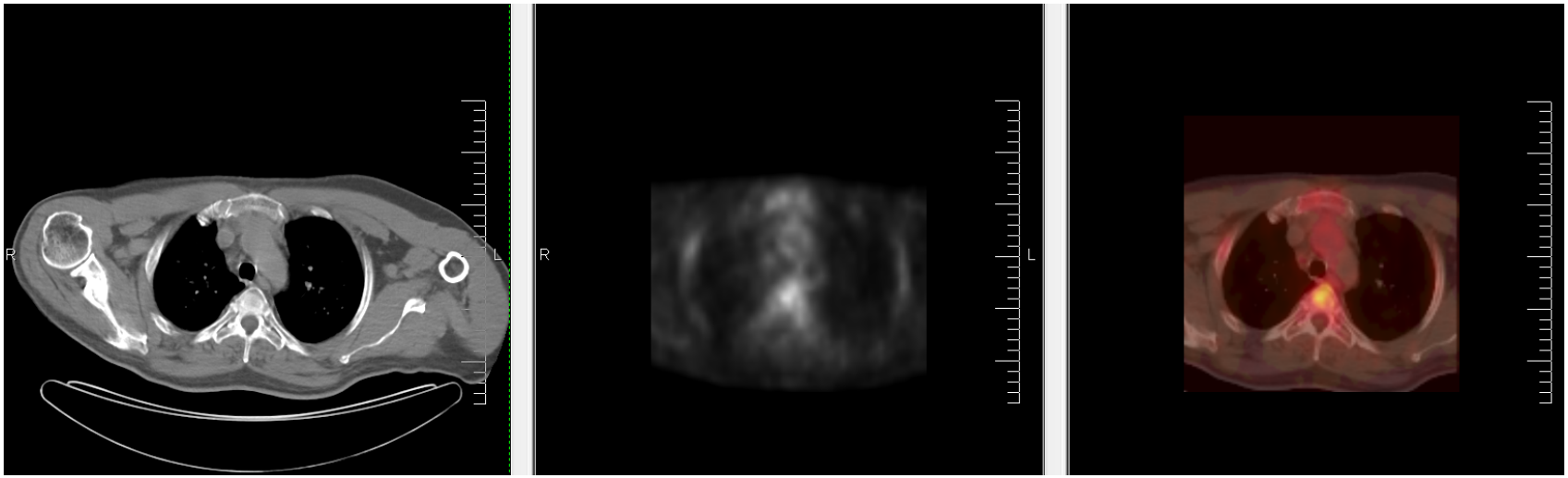

Figura 1 : Exemplo de fusão de imagens de diferentes modalidades. A primeira imagem é uma Tomografia Computadorizada, a segunda é uma Tomografia por Emissão de Pósitrons (PET) e a terceira a fusão das duas.

Source: https://www.leadtools.com/blog/medical-imaging/desktop-medical-viewer-control-updates/

- Os algoritmos de registro utilizam funções para mapear as imagens entre si
- A natureza das funções nos permite classificá-los
- A primeira classe utiliza funções simples como rotações e translações e os algoritmos pertencentes a ela realizam o registro rígido.
- Os algoritmos que alinham as imagens utilizando funções complexas, por exemplo aproximações por fluxo óptico ou funções radiais, são classificados como algoritmos de registro não-rígido.

Os algoritmos de registro são altamente custosos computacionalmente por dois motivos:

1. A quantidade de etapas que são realizadas pré registro
2. A quantidade de dados por estudos ou aplicação são muito grandes atualmente

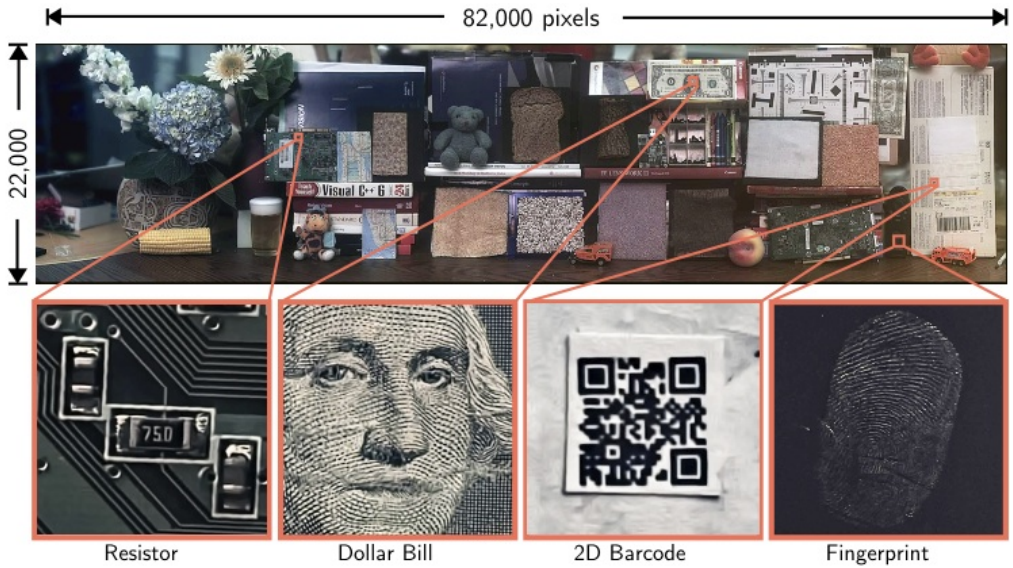

Figura 2 : Imagem *Gigapixel*.

Source: http://www.cs.columbia.edu/CAVE/projects/gigapixel/

A execução de um algoritmo de registro é composta de cinco passos:

1. Pré-processamento
2. Detecção de características
3. Correspondência de características
4. Estimativa da função de transformação
5. Reamostragem da Imagem Alvo

Evolução da área de pesquisa:

2004 Primeiros trabalhos a unir registro não-rígido de imagens em *Graphic Processing Units* (GPUs) [SDR04] [KDR+06]. Reportam aceleração de 4 e 12 vezes respectivamente

2007 As limitações da GPU são levadas em conta [GT08]. Reporta a aceleração de 35 vezes

2009 Linguagens de programação para GPUs são utilizadas [HHW09]. Reporta a aceleração de 15 vezes

2010 Algoritmos de registro desenvolvidos para GPU [MRT+10]. Reporta a execução do algoritmo em menos de 1 minuto

O objetivo principal do trabalho é avaliar a portabilidade de algoritmos de registro não-rígido para execução em GPUs. Os objetivos especificos do trabalho:

- Portar todas as etapas do registro para a arquitetura *Single Instruction, Multiple Data* (SIMD) [PH13], se assim elas ganharem eficiência
- Permitir o registro de múltiplas imagens simultaneamente
- Otimizar a ocupação dos recursos da GPU, preferencialmente entre as etapas do registro, com a finalidade de minimizar a transferência de dados entre a CPU e a GPU
- Avaliar a eficiência comparado com a implementação em CPU

- O registro de imagens encontra um alinhamento geométrico (f) entre duas imagens, a Imagem Alvo (A) e a Imagem Referência (R)

$$R(x, y) = A(f(x, y)) \tag{1}$$

- Ele percorre o espaço de parâmetros da função que representa o alinhamento em busca do melhor resultado

Figura 3 : À esquerda a Imagem Referência. À direita a Imagem Alvo.

1. Processamento:
   - O pré-processamento é opcional e totalmente dependente da aplicação
   - Filtros aplicados as imagens para diminuir a quantidade de ruido, ou realçar a imagem
   - Detecção de bordas (e. g. *Canny* [Can86] ou gradiente morfológico)
   - Algoritmos de segmentação (e. g. *Watersheds* [VS91] ou limiarização)

2. Detecção de características:
   - Características são estruturas de destaque na cena ou objeto dentro das imagens
   - Características devem ser facilmente identificadas
   - São representadas na imagem por um conjunto de pixels
   - Elas são divididas em três grupos:
     - **Baseadas em Regiões**, regiões com diferenças de contraste marcante entre suas vizinhas
     - **Baseadas em Retas**, interface entre duas regiões de uma imagem
     - **Baseadas em Pontos**, representam a região na qual se encontram através de alguma propriedade

Figura 4 : À esquerda as características da Imagem Referência. À direita as características da Imagem Alvo.

3. Correspondência de características:
   - O próximo passo encontra correspondências entre as características das imagens Referência e Alvo
   - Para cada grupo de correspondências existem métodos para a correspondência deles, com a adição de um outro:
     - **Baseadas em Áreas**, utiliza janelas para mensurar áreas das duas imagens
     - **Baseadas em Regiões**, compara o contorno das regiões
     - **Baseadas em Retas**, pareia retas utilizando sua direção, comprimento e largura
     - **Baseadas em Pontos**, utiliza descritores para encontrar as correspondências entre os pontos

Dois algoritmos populares para detecção e correspondência de características:

- *Scale Invariant Feature Transform* (SIFT) [Low04]
- *Speeded Up Robust Features* (SURF) [BETVG08]

Os passos gerais executados por eles:

- Encontrar pontos **chave** no espaço de escala das imagens
- Determinar o vetor de descritores dos pontos **chave**
- Parear as características entre as imagens

Figura 5 : À esquerda as correspondências da Imagem Referência. À direita as correspondências da Imagem Alvo. O SIFT foi o algoritmo utilizado.

4. Função de transformação:
   - A primeira estimativa parte das correspondências
   - O algoritmo de registro passeia no espaço dos parâmetros em busca de um alinhamento ótimo
   - O modelo de transformação deve ser escolhido conforme as necessidades da aplicação
     - Fluxo óptico
     - Informação Mútua
     - Translações e rotações

5. Reamostragem da Imagem Alvo:
   - A reamostragem da imagem Alvo aplica os parâmetros encontrados acima na transformação
   - Reamostragem descrita pela equação abaixo

$$F(x_i, y_j) = A(f(x_i, y_j))), \forall (i = 1, \ldots, n_c), (j = 1, \ldots, n_l) \quad (2)$$

   - Métodos de interpolação são necessários nesse passo

Figura 6 : À esquerda imagem reamostrada usando o método de vizinho mais próximo. À direita a imagem reamostrada usando o método bilinear.

Figura 7 : À esquerda a Imagem Referência. À direita a Imagem Registrada.

- *High Performance Computing* (HPC) designa sistemas de alta capacidade de processamento e armazenamento de dados
- Uma instanciação de HPC pode ser feita com GPU
- Chamamos a aplicação de GPUs na solução de problemas computacionais de *General-Purpose Computing on Graphics Processing Units* (GPGPU)

Sobre o processamento na GPU:

- Seguem a arquitetura de SIMD
- É composta de milhares de processadores, agrupados em *Streaming Multiprocessors*
- As *Threads* são organizadas em blocos, e eles são divididos para os SM

```
1   __global__ void VecAdd ( float* a,
2                            float* b,
3                            float* c) {
4     int i = threadIdx.x;
5     a[i] = b[i] + c[i];
6   }
7
8   int main () {
9     ...
10    VecAdd<<<1,M>>>(a, b, c);
11    ...
12  }
```

Limitações a se levar em conta:

- O acesso a blocos de memória é concorrente
- A transferência de dados entre GPU e CPU impacta no desempenho
- Evitar o número de bifurcações lógicas

- Dois dos algoritmos mais representativos foram selecionados para serem avaliados
- *Demons*
  - Registra pequenas deformações com baixo custo computacional
  - Modela a transformação ponto a ponto
- TPS
  - Flexível
  - Não utiliza as intensidades das imagens

Um dos algoritmos estudados de registro não-rígido estudado foi o *Demons* [Thi95]

- O *Demons* tem como base o modelo de atratores
- Ele adiciona o gradiente aos atratores para modelar a direção deles

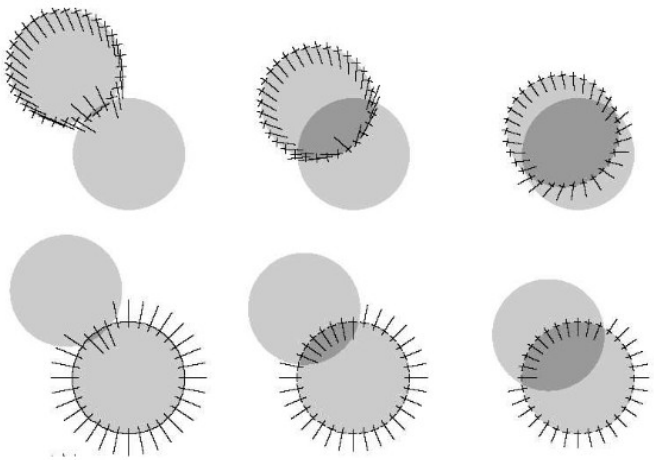

Figura 8 : A primeira linha demonstra o sistema de atratores e a segunda o *Demons*

O *Demons* supõe que a transformação só movimenta os pixels, não muda suas intensidades e que as imagens diferem de uma unidade de tempo.

$$\vec{v} \cdot \vec{\nabla} R = A - R, \text{ onde } \vec{\nabla} R \text{ é o gradiente de R} \tag{3}$$

$$\vec{v} = \frac{2(A-R) * (\vec{\nabla} R \vec{\nabla} A)}{(\vec{\nabla} R + \vec{\nabla} A)^2 * (A-R)^2} \tag{4}$$

O *Demons* é um algoritmo iterativo. Como entrada ele recebe a imagem referência e alvo e possivelmente um campo vetorial inicial. Cada iteração executa os passos:

1. Para cada *Demon* em A_i, calculamos $\vec{v_i}$, criando um novo campo vetorial V_i
2. Aplicamos um filtro Gaussiano para retirar o ruído introduzido pelo processo em V_i
3. Aplicamos V_i em A para obter A_{i+1};

O próximo algoritmo de registro não-rígido estudado foi o Thin Plate Splines (TPS) [Gos05]

- O TPS é uma função de interpolação radial
- As posições das características definem a função de interpolação
- A superfície é definida pela seguinte equação[Boo89]:

$$f(x,y) = A_0 + A_1 x + A_2 y + \sum_{i=0}^{n} F_i r_i^2 ln r_i^2$$
$$r_i^2 = (x - x_i)^2 + (y - y_i)^2 + d^2 \quad (5)$$

O TPS determina os valores das variáveis A_0, A_1, A_2 e dos F_i:

$$\sum_{i=1}^{n} F_i = 0, \sum_{i=1}^{n} F_i x = 0, \sum_{i=1}^{n} F_i y = 0$$

$$\begin{aligned} f(x_1, y_1) &= A_0 + A_1 x + A_2 y + \sum_{i=0}^{n} F_i r_{i1}^2 ln r_{i1}^2 \\ &\vdots \\ f(x_n, y_n) &= A_0 + A_1 x + A_2 y + \sum_{i=0}^{n} F_i r_{in}^2 ln r_{in}^2 \end{aligned} \tag{6}$$

Para os experimentos uma das imagens padrão do software BioImage [PJR+05] foi utilizada. As deformações aplicadas a imagem de teste foram baseadas no trabalho Zargorchev [ZG06].

As transformações serão aplicadas à grade abaixo.

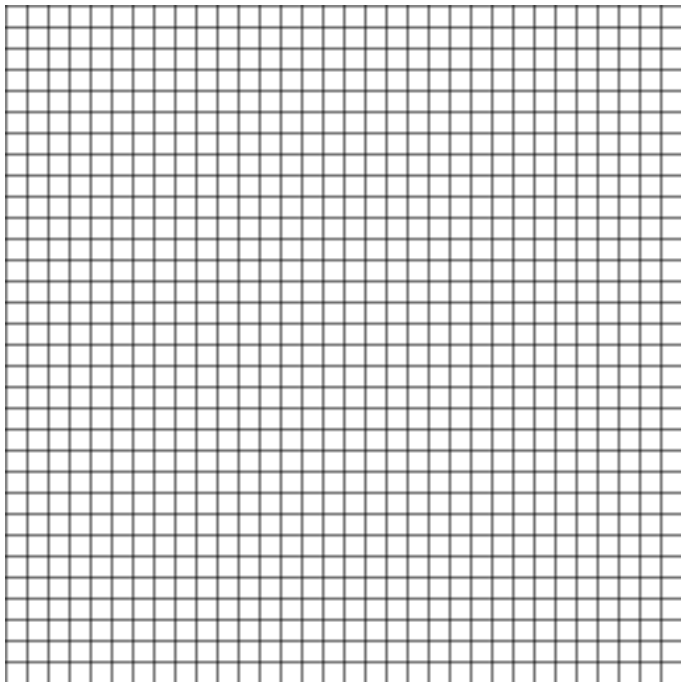

Figura 9 : Grade sem deformações.

A primeira transformação é dada por:

$$X = x - 8sen(x/32)$$
$$Y = y + 4cos(x/16) \quad (7)$$

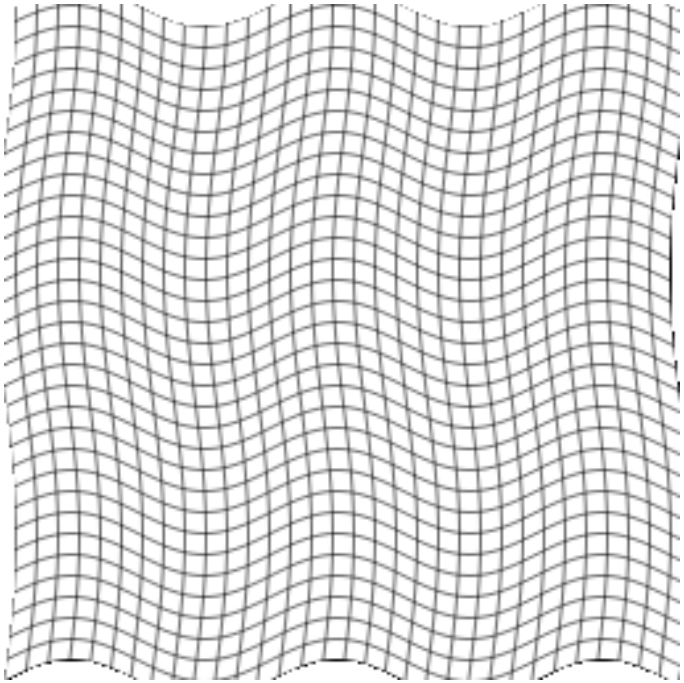

Figura 10 : Grade deformada pela função senoidal.

A próxima transformação é dada por:

$$X = x + 50(x - n_c)/r$$
$$Y = y + 50(y - n_l)/r \quad (8)$$

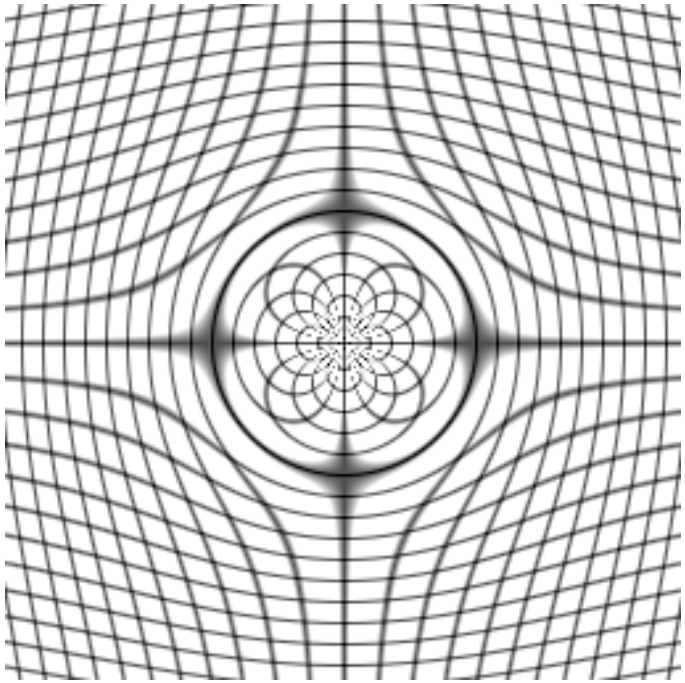

Figura 11 : Grade deformada pela função de distância inversa.

A última transformação é dada pela combinação das duas:

$$X = x - 8sen(x/32) + 50(x - n_c)/r$$
$$Y = y + 4cos(x/16) + 50(y - n_l)/r \quad (9)$$

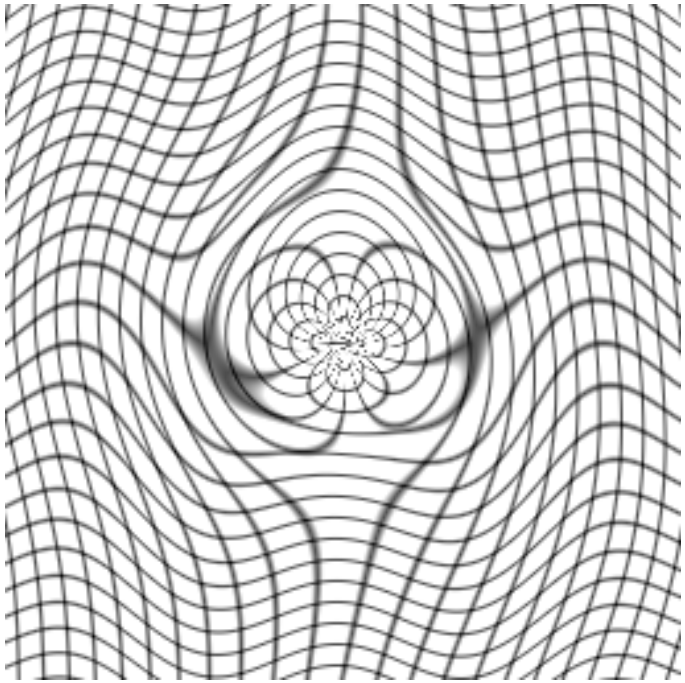

Figura 12 : Grade deformada pela combinação das duas anteriores.

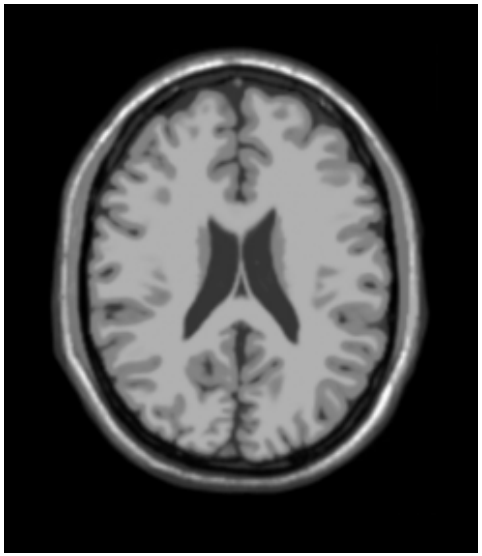

Imagem Referência

Senoidal

Distância inversa

Combinação

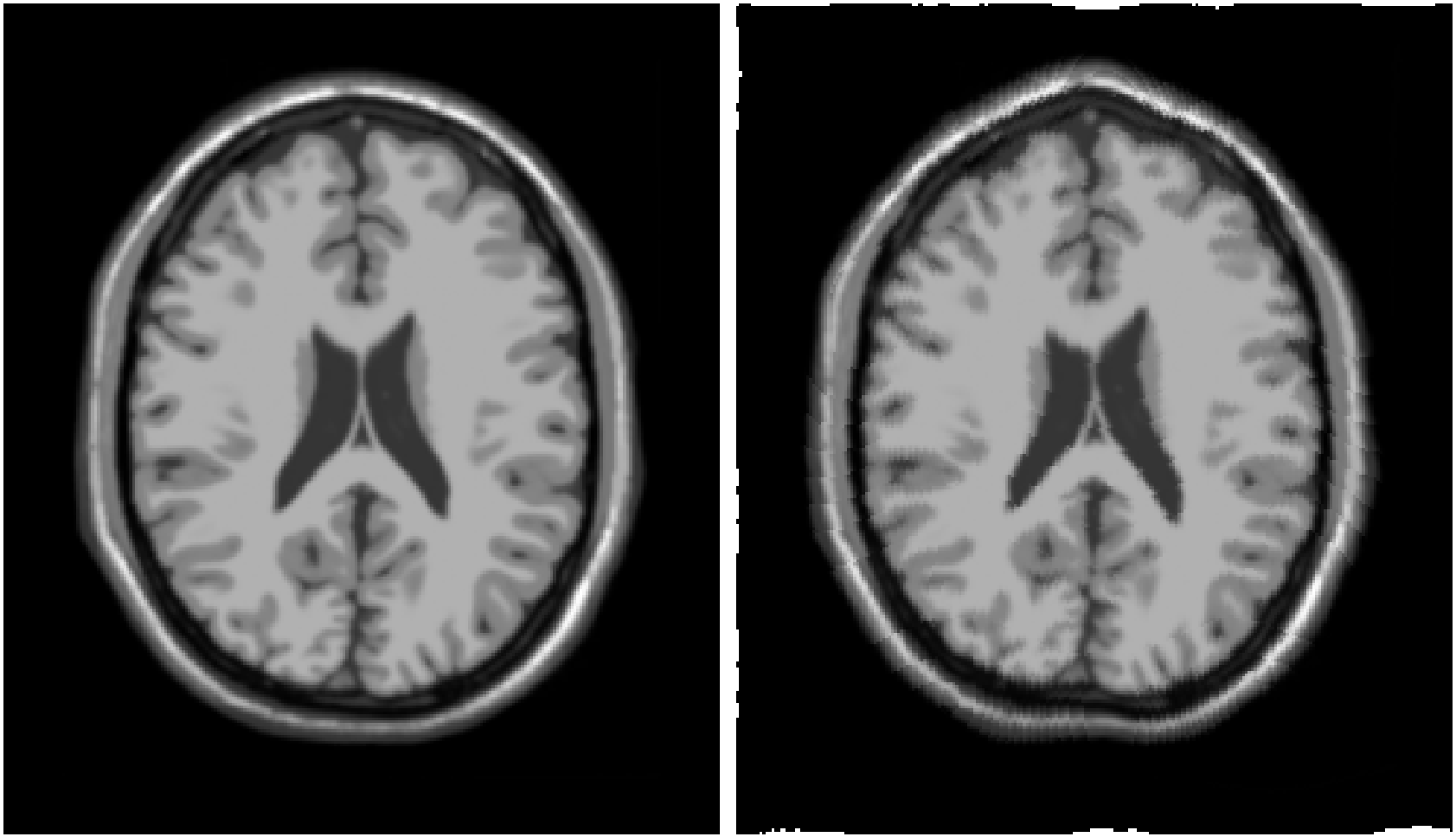

Figura 13 : À esquerda a Imagem Referência. À direita a imagem registrada pelo TPS.

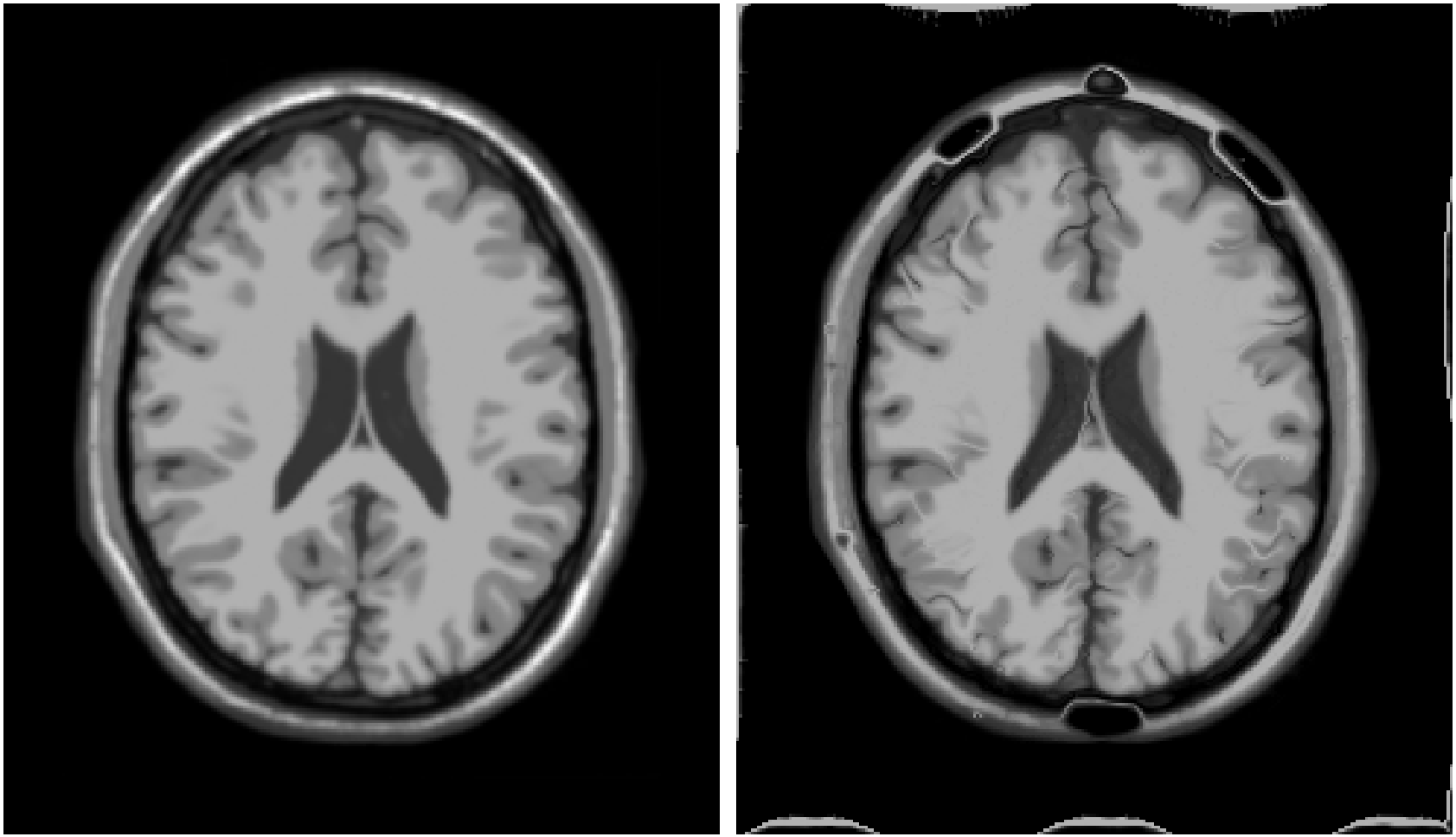

Figura 14 : À esquerda a Imagem Referência. À direita a imagem registrada pelo *Demons*.

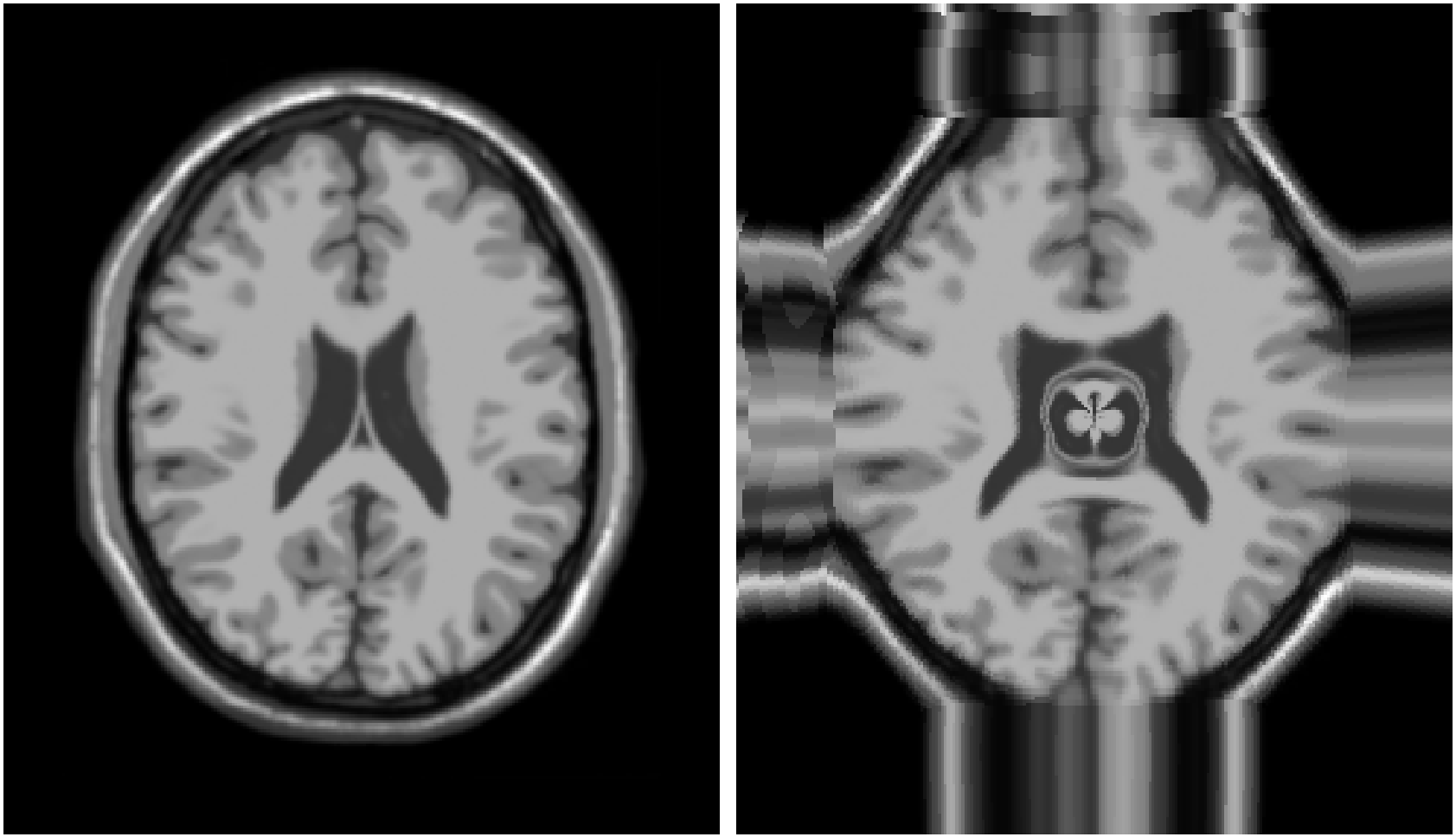

Figura 15 : À esquerda a Imagem Referência. À direita a imagem registrada pelo TPS.

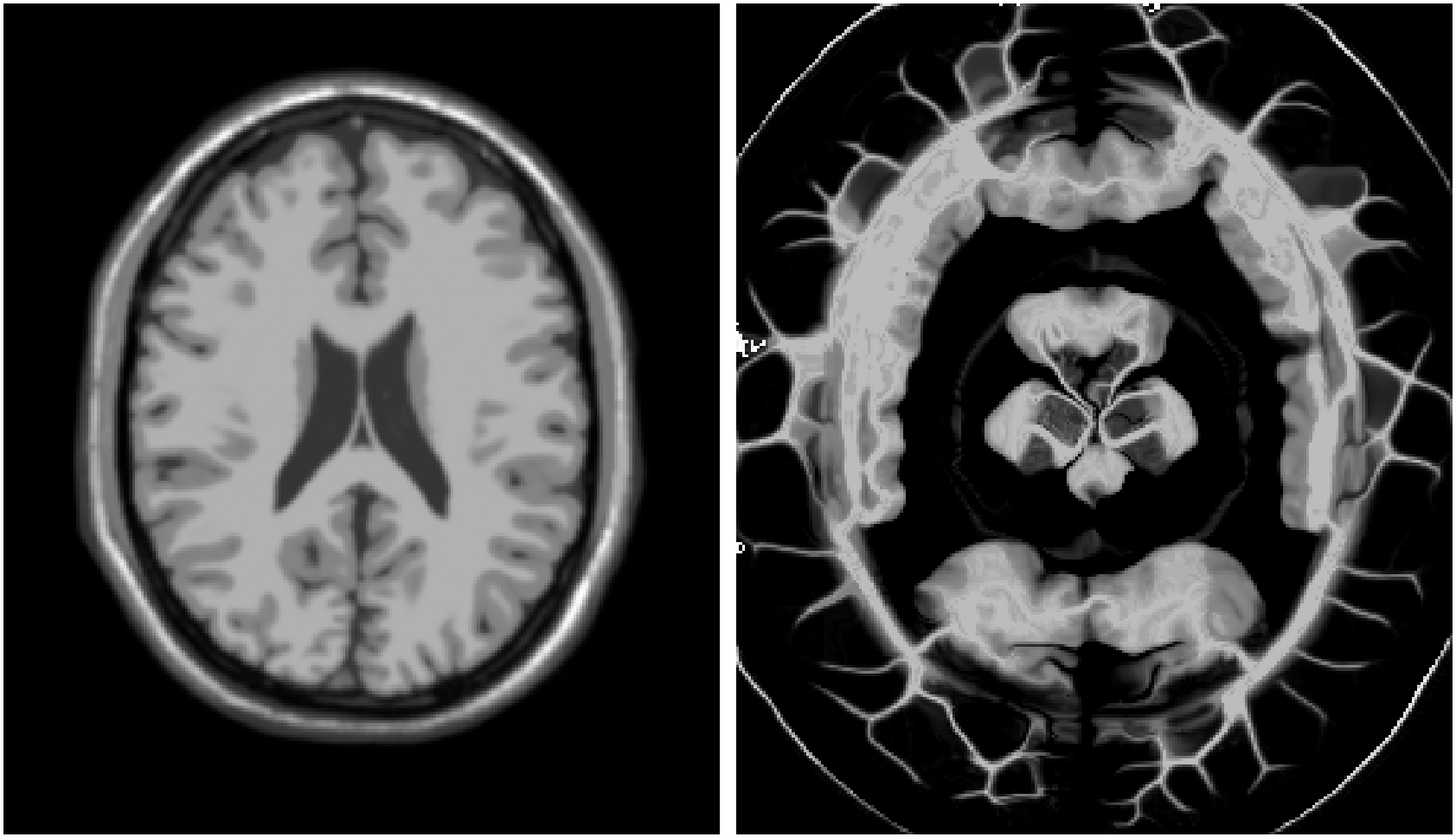

Figura 16 : À esquerda a Imagem Referência. À direita a imagem registrada pelo *Demons*.

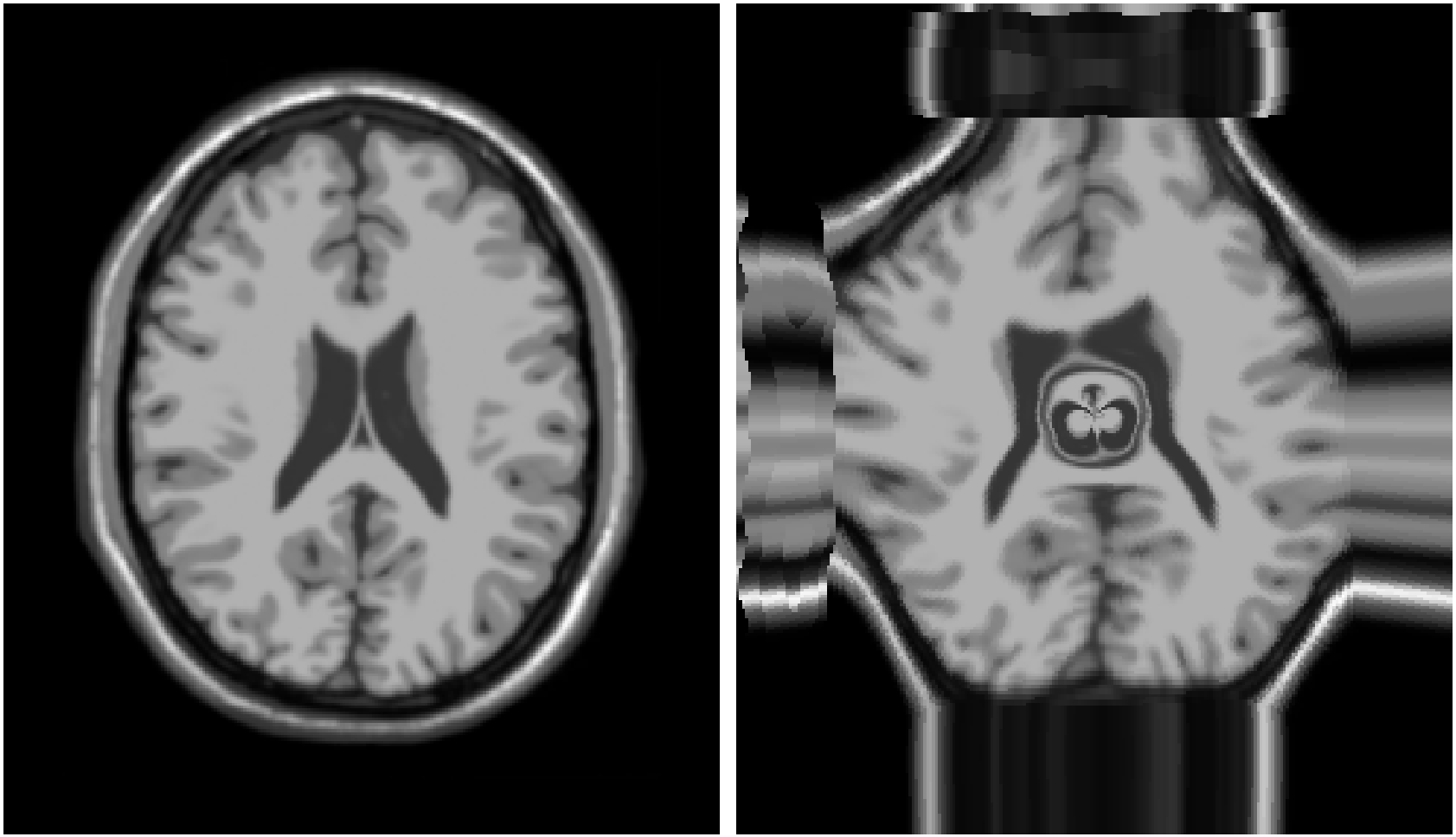

Figura 17 : À esquerda a Imagem Referência. À direita a imagem registrada pelo TPS.

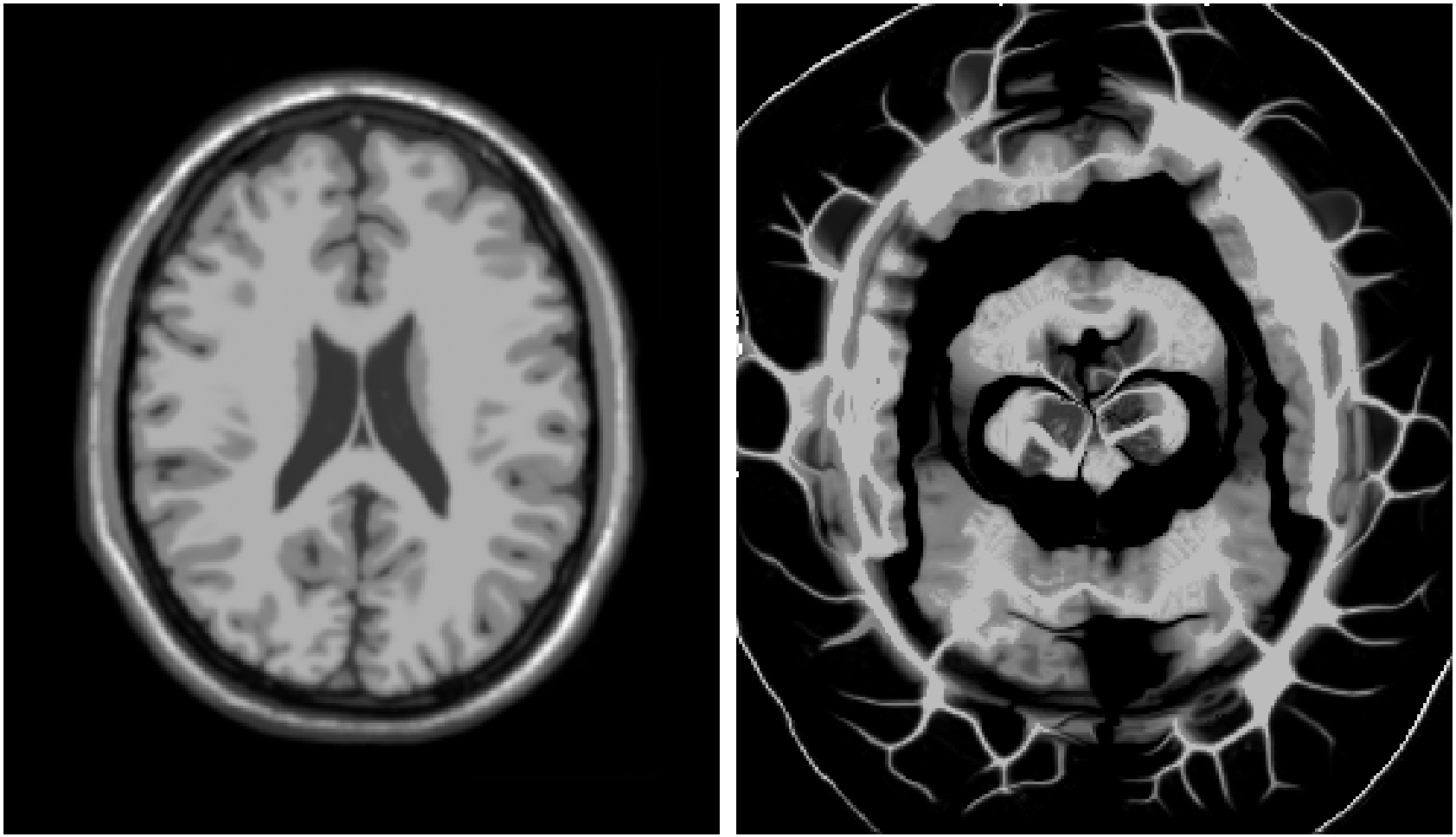

Figura 18 : À esquerda a Imagem Referência. À direita a imagem registrada pelo *Demons*.

Algoritmo	Tempo médio	desvio padrão	máximo	mínimo
Demons	686,42 s	27,71 s	740,21 s	632,90 s
TPS	329,19 s	44,57 s	432,75 s	289,43 s

Tabela 1 : Tempo de execução dos testes com a deformação senoidal

Algoritmo	Tempo Assintótico de execução
Demons	$\mathcal{O}(x * n_l^2 * n_c^2)$
TPS	$\mathcal{O}(\frac{4}{3} n_{cp}^3 + n_l * n_c * n_{cp})$

Tabela 2 : Tempo Assintótico de execução dos algoritmos

Os próximos passos para completar o estudo são, em prioridade:

1. Avaliar a portabilidade do SIFT e SURF para GPU
2. Criar uma versão GPGPU dos algoritmos utilizando a linguagem CUDA
3. Possibilitar a execução com múltiplas entradas (imagens)
4. Escrever um artigo científico
5. Escrever a dissertação

Tarefa	Março	Abril	Maio	Junho	Julho	Agosto	Setembro
1	X						
2	X	X	X				
3			X	X			
4	X	X	X				
5			X	X	X	X	X

Tabela 3 : Cronograma.

Referências I

Herbert Bay, Andreas Ess, Tinne Tuytelaars e Luc Van Gool.
Speeded-up robust features (surf).
Computer vision and image understanding, 110(3):346–359, 2008.

Fred L. Bookstein.
Principal warps: Thin-plate splines and the decomposition of deformations.
IEEE Transactions on pattern analysis and machine intelligence, 11(6):567–585, 1989.

John Canny.
A computational approach to edge detection.
Pattern Analysis and Machine Intelligence, IEEE Transactions on, (6):679–698, 1986.

Referências II

A. Ardeshir Goshtasby.
2DD and 3-D Image Registration: For Medical, Remote Sensing, and Industrial Applications.
Wiley-Interscience, 2005.

Harald Grossauer e Peter Thoman.
Gpu-based multigrid: Real-time performance in high resolution nonlinear image processing.
Em *Computer Vision Systems*, páginas 141–150. Springer, 2008.

Referências III

Xiao Han, Lyndon S Hibbard e Virgil Willcut.
Gpu-accelerated, gradient-free mi deformable registration for atlas-based mr brain image segmentation.
Em *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, páginas 141–148. IEEE, 2009.

Alexander Köhn, Johann Drexl, Felix Ritter, Matthias König e Heinz-Otto Peitgen.
Gpu accelerated image registration in two and three dimensions.
Em *Bildverarbeitung für die Medizin 2006*, páginas 261–265. Springer, 2006.

Referências IV

David G Lowe.
Distinctive image features from scale-invariant keypoints.
International journal of computer vision, 60(2):91–110, 2004.

Marc Modat, Gerard R Ridgway, Zeike A Taylor, Manja Lehmann, Josephine Barnes, David J Hawkes, Nick C Fox e Sébastien Ourselin.
Fast free-form deformation using graphics processing units.
Computer methods and programs in biomedicine, 98(3):278–284, 2010.

Referências V

David A Patterson e John L Hennessy.
Computer organization and design: the hardware/software interface.
Newnes, 2013.

Xenophon Papademetris, Marcel Jackowski, Nallakkandi Rajeevan, R Todd Constable e LH Staib.
Bioimage suite: An integrated medical image analysis suite.
The Insight Journal, 1:3, 2005.

Robert Strzodka, Marc Droske e Martin Rumpf.
Image registration by a regularized gradient flow. a streaming implementation in dx9 graphics hardware.
Computing, 73(4):373–389, 2004.

Referências VI

Jean-Philippe Thirion.
Fast non-rigid matching of 3d medical images.
1995.

Luc Vincent e Pierre Soille.
Watersheds in digital spaces: an efficient algorithm based on immersion simulations.
IEEE transactions on pattern analysis and machine intelligence, 13(6):583–598, 1991.

Lyubomir Zagorchev e Ardeshir Goshtasby.
A comparative study of transformation functions for nonrigid image registration.
Image Processing, IEEE Transactions on, 15(3):529–538, 2006.