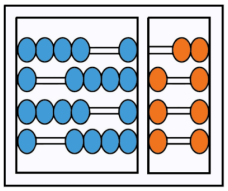

Instituto de Computação

UNIVERSIDADE ESTADUAL DE CAMPINAS

Relatório de Implementação de uma Aplicação de Alinhamento Automático de Documentos Utilizando Projeção Horizontal e Transformada de Hough

Relatório de implementação desenvolvido para disciplina MO443 Introdução ao Processamento de Imagem Digital.

Professor: Hélio Pedrini

Aluna: Juliana Midlej do Espírito Santo

RA: 200208

Campinas - São Paulo

24 de Maio de 2024

1. Introdução 3
2. Materiais e Métodos 3
2.1 Requerimentos 3
2.2 Organização de Arquivos 3
2.3 Alinhamento Automático de Documentos 4
2.4 Cálculo do Ângulo de Inclinação Utilizando Projeção Horizontal 4
2.5 Cálculo do Ângulo de Inclinação Utilizando Transformada de Hough 6
2.6 Determinação do Ângulo de Inclinação com Auxílio de OCR 7
2.7 Imagens de Teste 7
3. Resultados e Discussão 8
3.1 Alinhamento com Projeção Horizontal 9
3.2 Alinhamento com Transformada de Hough 9
3.3 Aplicação de OCR 10
4. Conclusão 11

1. Introdução

A análise de documentos pode ser empregada na conversão automática da informação contida nas imagens de documentos em texto editável. Um problema frequente que ocorre no processo de digitalização é o desalinhamento do documento. A correção da inclinação de um documento é fundamental para o adequado funcionamento de sistemas de reconhecimento ótico de caracteres.

O objetivo deste trabalho é implementar uma aplicação de alinhamento automático de imagens de documentos utilizando os algoritmos de projeção horizontal e transformada de Hough. O projeto que contém a aplicação está disponível no GitHub no link: https://github.com/annwith/digital_image_processing.

2. Materiais e Métodos

Esta seção abordará os requerimentos do projeto, a organização dos arquivos e as escolhas de implementação para a aplicação de alinhamento automático de documentos.

2.1 Requerimentos

Para implementar o algoritmo foram utilizados a linguagem de programação Python em sua versão 3.8.10 e os pacotes listados na Tabela 1.

Pacote	Versão
OpenCV (opencv-python)	4.9.0.80
NumPy (numpy)	1.24.4
Pytesseract (pytesseract)	0.3.10
Scikit Image (scikit-image)	0.21.0

Tabela 1: Pacotes

2.2 Organização de Arquivos

Para melhor organização e clareza, o projeto que contém o código foi dividido nas seguintes pastas e arquivos:

- ➔ ***input_images***: Pasta de imagens de entrada (desalinhadas);
- ➔ ***output_images***: Pasta de imagens de saída (alinhadas);
- ➔ ***output_texts***: Pasta de textos obtidos pelo reconhecimento ótico de caracteres em imagens alinhadas;
- ➔ ***document_align.py***: Arquivo de aplicação do alinhamento de documentos;
- ➔ ***README.md***: Arquivo que contém informações sobre o projeto e como executar o arquivo *document_align.py*;
- ➔ ***requirements.txt***: Arquivo que contém os pacotes necessários para execução do projeto.

2.3 Alinhamento Automático de Documentos

O alinhamento automático de documentos é implementado no arquivo *document_align.py*. A execução desse arquivo deve ser feita da seguinte forma:

python3 document_align.py -i image_path.png -p 1 -m 0

Os parâmetros passados por linha de comando são o arquivo da imagem que será alinhada, a precisão e o modo do alinhamento (0 para projeção horizontal e 1 para transformada de Hough). A precisão informada define a distância entre os ângulos testados e pode assumir valores como 1, 0.1, 0.01 e assim por diante (qualquer potência de 10 entre 0 e 1). Se um valor inválido for informado, um erro será acusado.

A imagem é lida em escala de cinza. Dependendo do modo de alinhamento escolhido, a função apropriada para calcular a inclinação do documento é chamada: slope_from_horizontal_projection ou slope_from_hough_transform. Se o modo informado não for 0 ou 1, um erro é acusado. Ambas as funções serão explicadas nas Seções 2.4 e 2.5.

As funções de cálculo da inclinação retornam um *NumPy array* contendo a inclinação do documento. No caso da projeção horizontal, mais de um valor pode ser retornado.

Como foi notado em algumas imagens, a saída das funções de cálculo da inclinação pode estar invertida 180º. Para correção, a determinação final do ângulo de inclinação da imagem é feita pela função *find_best_slope_using_ocr* que será explicada na Seção 2.6. Essa função recebe como parâmetro o *NumPy array* retornado por uma função de cálculo de inclinação e retorna o valor que determina a inclinação do documento de forma que o texto contido na imagem, se houver texto, não fique invertido.

Após a correção do alinhamento, os textos das imagens original e alinhada são obtidos através de OCR. Por fim, são salvas a imagem alinhada e os textos obtidos nas pastas *output_images* e *output_texts*. O nome dos arquivos de imagens alinhadas é definido incluindo o nome da imagem, o modo e a inclinação do alinhamento, por exemplo *neg_4_mode_0_rotated_356.png*. Os nomes dos arquivos de textos das imagens possuem o mesmo nome de suas respectivas imagens com a extensão *txt*.

2.4 Cálculo do Ângulo de Inclinação Utilizando Projeção Horizontal

O cálculo do ângulo de inclinação utilizando projeção horizontal está no arquivo *document_align.py* e é implementado pela função ***slope_from_horizontal_projection***. A função recebe como parâmetros uma imagem e a precisão para o cálculo do ângulo de inclinação e retorna um *numpy.array* que contém o ângulo de inclinação da imagem.

A imagem recebida deve estar em escala de cinza e passa por um processo de detecção de bordas com o Canny Edge utilizando como thresholds mínimo e máximo 50 e 150, respectivamente. A escolha dos thresholds para o Canny Edge foi feita empiricamente a partir de testes em imagens.

Para detecção da inclinação, o algoritmo rotaciona a imagem binária, calcula a projeção horizontal e a função objetivo para cada ângulo no intervalo [0º, 360º[de acordo com a precisão definida e retorna o ângulo – em caso de empate, mais de um ângulo é retornado – que maximiza a função objetivo. A função objetivo utilizada para escolher o

ângulo de inclinação foi a soma do quadrado das diferenças entre células adjacentes do perfil da projeção horizontal.

Para melhorar o tempo de execução da função, o método para encontrar o melhor ângulo é descrito da seguinte forma:

1. **Teste Inicial com Distância 1**: Inicialmente, os ângulos são testados com uma distância de 1 grau. Isso significa que os ângulos testados são, por exemplo, 0°, 1°, 2°, ..., 359°.
2. **Escolha do Melhor Ângulo**: Após testar todos os ângulos com distância 1, é escolhido o melhor ângulo (aquele que maximiza a função objetivo). Se houver empate, todos os ângulos que produziram o melhor resultado são escolhidos.
3. **Teste com Maior Precisão**: Em seguida, os ângulos próximos ao melhor ângulo (ou aos melhores ângulos em caso de empate) são testados com uma maior precisão. A nova distância entre os ângulos é de 0.1 grau.
4. **Definição dos Ângulos Próximos**: Se, por exemplo, o melhor ângulo escolhido inicialmente foi 176°, os ângulos próximos que serão testados estarão no intervalo]175°, 177°[com uma distância de 0.1 grau. Portanto, os ângulos testados serão 175.1°, 175.2°, ..., 176.9°.
5. **Iteração até a Precisão Necessária**: Esse processo pode ser repetido, reduzindo a distância entre os ângulos a cada iteração (por exemplo, de 0.1 para 0.01, depois para 0.001, etc.), até atingir a precisão desejada.

A função mostra como resultados intermediários para cada iteração definida no passo 5 os intervalos de ângulos que serão testados e a distância entre cada ângulo, além dos ângulos máximos escolhidos (melhores ângulos). A Figura 1 demonstra os resultados intermediários para uma execução cuja precisão é 0.1. Caso exista mais de um ângulo em “Max angles” em uma iteração, “Start” e “End” terão mais de um elemento na iteração seguinte, de forma que os novos intervalos de teste são definidos como: [Start[i], End[i]] para todo i de 1 até o número de elementos em “Start”.

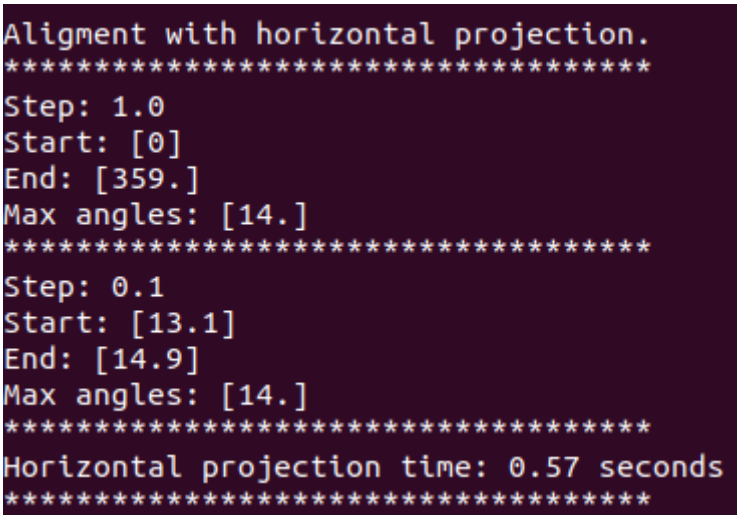

Figura 1: Resultados Intermediários da Projeção Horizontal

O tempo de execução que o algoritmo implementado com projeção horizontal leva para encontrar o ângulo de inclinação da imagem também é mostrado como resultado intermediário. O tempo da detecção de bordas não é contabilizado.

2.5 Cálculo do Ângulo de Inclinação Utilizando Transformada de Hough

O cálculo do ângulo de inclinação utilizando a transformada de Hough está no arquivo *document_align.py* e é implementado pela função ***slope_from_hough_transform***. A função recebe como parâmetros uma imagem e a precisão para o cálculo do ângulo de inclinação e retorna um *numpy.array* que contém o ângulo de inclinação da imagem.

A imagem recebida deve estar em escala de cinza e passará por um processo de detecção de bordas com o Canny Edge exatamente como foi descrito na Seção 2.4.

A imagem binária produzida é passada como entrada para a função *cv.HoughLines*, que utiliza uma resolução de distância de 1 pixel e uma resolução angular definida pela precisão especificada em radianos. O parâmetro threshold é definido como um valor que depende do tamanho da imagem, calculado como um sexto da soma das dimensões da imagem. Esse parâmetro determina o número suficiente de pontos em alinhamento para que uma linha seja detectada e foi escolhido empiricamente através de testes com imagens.

Após a execução da função *cv.HoughLines*, é verificado se alguma linha foi encontrada. Se nenhuma linha for detectada, um erro é acusado com a mensagem "No lines were found". Se linhas forem encontradas, o número de linhas detectadas é impresso. Em seguida, os ângulos dessas linhas são extraídos, convertidos de radianos para graus e somados em 90º. Essa soma é justificada pela Figura 2, a linha de interesse é a vermelha, e o ângulo retornado pela função *cv.HoughLines* é *theta*.

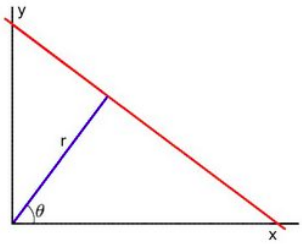

Figura 2: Ângulo de uma Linha pela Transformada de Hough

A função *slope_from_hough_transform* retorna a mediana dos ângulos das linhas detectadas, arredondado para a precisão definida, como um *array NumPy*. A imagem binária com as linhas detectadas, o número total de linhas detectadas, a mediana e o tempo de execução são exibidos como resultados intermediários.

O tempo de execução mostrado considera a execução da função *cv.HoughLines* e o cálculo da mediana, desconsiderando a detecção de bordas e a impressão das imagens.


```
Aligment with hough transform.
Number of lines found: 68
Median: 194.5
**********************************
cv.HoughLines time: 0.01 seconds
**********************************
```

Figura 3: Resultados Intermediários da Transformada de Hough

2.6 Determinação do Ângulo de Inclinação com Auxílio de OCR

A determinação do ângulo de inclinação da imagem é feito pela função ***find_best_slope_using_ocr***. Lembrando que: para cálculos efetuados com a transformada de Hough, apenas um ângulo é retornado; já para cálculos feitos com a projeção horizontal, usualmente apenas um ângulo é retornado, mas é possível que sejam mais.

A função *find_best_slope_using_ocr* determina a inclinação correta de um texto em uma imagem utilizando a função *image_to_data* do *pytesseract*. A função *image_to_data* retorna um dicionário contendo todas as palavras encontradas e a confiança para cada uma delas.

A função *find_best_slope_using_ocr* recebe um *NumPy array* de possíveis inclinações, adiciona 180 graus para considerar, em todos os casos, a imagem invertida, ordena os ângulos e, para cada inclinação, rotaciona a imagem, executa a função *image_to_data* e calcula a mediana das confianças das palavras detectadas.

A função imprime como resultados intermediários para cada inclinação testada a inclinação atual e a mediana das confianças. A inclinação com a maior mediana é retornada; em caso de empate, a menor inclinação é escolhida. Os resultados intermediários são mostrados na Figura 4.

```
**********************************
Find correct slope using OCR.
**********************************
Slope: 14.0
Median confidence: 94.0
**********************************
Slope: 194.0
Median confidence: 8.0
**********************************
Correct slope: 14.0
```

Figura 4: Resultados Intermediários da Correção de Alinhamento com OCR

2.7 Imagens de Teste

As imagens utilizadas para teste estão na pasta *input_images* e são as imagens disponibilizadas no link: https://www.ic.unicamp.br/~helio/imagens_inclinadas_png.

3. Resultados e Discussão

Ambos os métodos utilizados foram capazes de alinhar todas as imagens de teste. O alinhamento é mostrado na Figura 5. Para todos os casos de teste com textos nas imagens, o alinhamento foi capaz de evitar que os textos ficassem invertidos. No caso da partitura, a execução do OCR não reconhece nenhuma palavra, então foi escolhido arbitrariamente o menor ângulo como definido na Seção 2.6 para os casos de empate.

Para precisão igual a 1, todos os resultados dos métodos de projeção horizontal e transformada de Hough foram iguais. Para precisão 0.1, os resultados se mantiveram iguais para as imagens à esquerda da Figura 5, nas imagens a direita, houveram algumas diferenças entre os métodos, sendo as mais significativas as ocorridas nas imagens contendo o código de barras e a partitura, cuja variação chegou a 0.7º e a 0.3º respectivamente.

Todas as imagens geradas estão na pasta *output_images*. É possível saber a inclinação e precisão utilizadas através do nome das imagens, por exemplo *neg_4_mode_0_rotated_356.0.png* e *neg_4_mode_0_rotated_356.png*, ambas imagens possuem o mesmo ângulo e a primeira foi obtida utilizando uma precisão de 0.1, enquanto a segunda foi obtida utilizando a precisão 1.

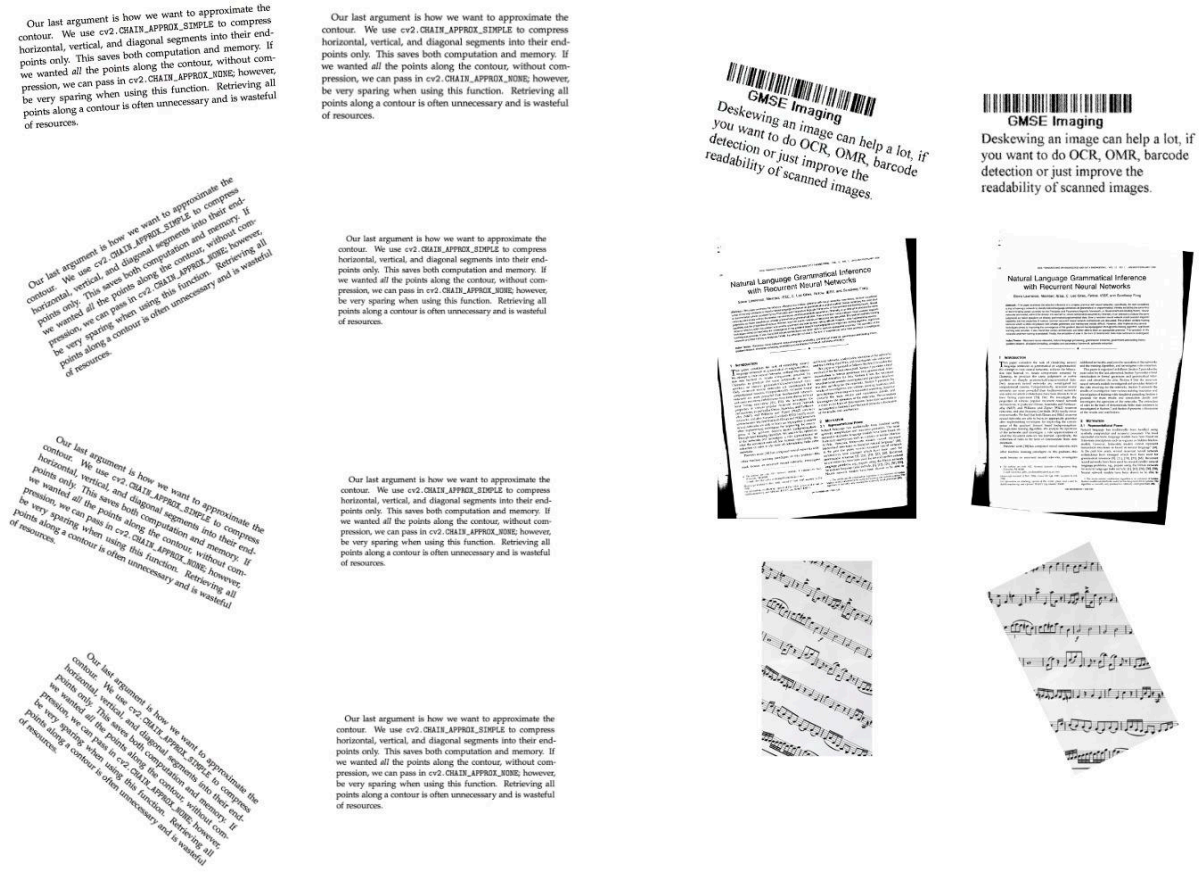

Figura 5: Alinhamento das Imagens de Teste

3.1 Alinhamento com Projeção Horizontal

O alinhamento automático com o método da projeção horizontal foi capaz de alinhar todas as imagens de teste, como foi mostrado na Figura 5. Entretanto, o método se mostrou mais lento do que a transformada de Hough, especialmente para imagens maiores. Uma vantagem desse método é ser menos paramétrico.

3.2 Alinhamento com Transformada de Hough

O alinhamento automático com o método da transformada de Hough foi capaz de alinhar todas as imagens de teste, como foi mostrado na Figura 5. A Figura 6 mostra a detecção das linhas com precisão 1 para as imagens de teste. Na imagem do artigo, foi possível notar que apenas duas linhas foram encontradas.

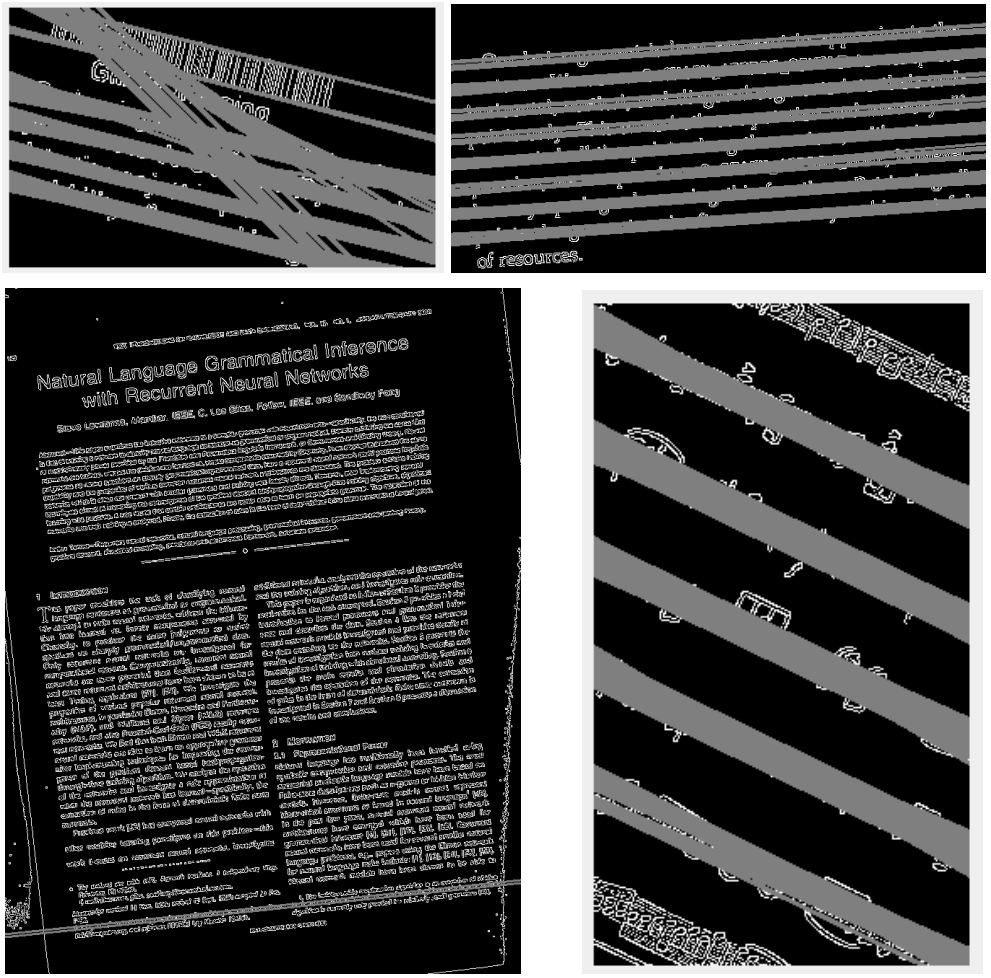

Figura 6: Linhas Detectadas pela Transformada de Hough (*cv.HoughLines*)

Uma desvantagem desse alinhamento é que linhas podem não ser encontradas em uma imagem a depender do número mínimo de pontos necessários para detectar uma linha. Uma solução iterativa para esse problema pode ser implementada, de forma que, se nenhuma linha for encontrada, o threshold é reduzido. Essa solução adiciona complexidade ao método de alinhamento pela transformada de Hough. A escolha arbitrária do número de pontos necessários funcionou para todas as imagens testadas.

Uma função da transformada que retorne o acumulador também pode solucionar de forma mais eficiente esse problema.

Foi percebido também que aumentar a precisão do método pode levar a detecção de mais linhas, já que mais ângulos são testados. Isso foi notado na imagem do artigo, em que utilizar a precisão 0.1 levou a detecção de 38 linhas, enquanto com a precisão 1 foram encontradas apenas 2 linhas. A Figura 7 mostra as linhas encontradas pelo método utilizando a precisão 1 e 0.1.

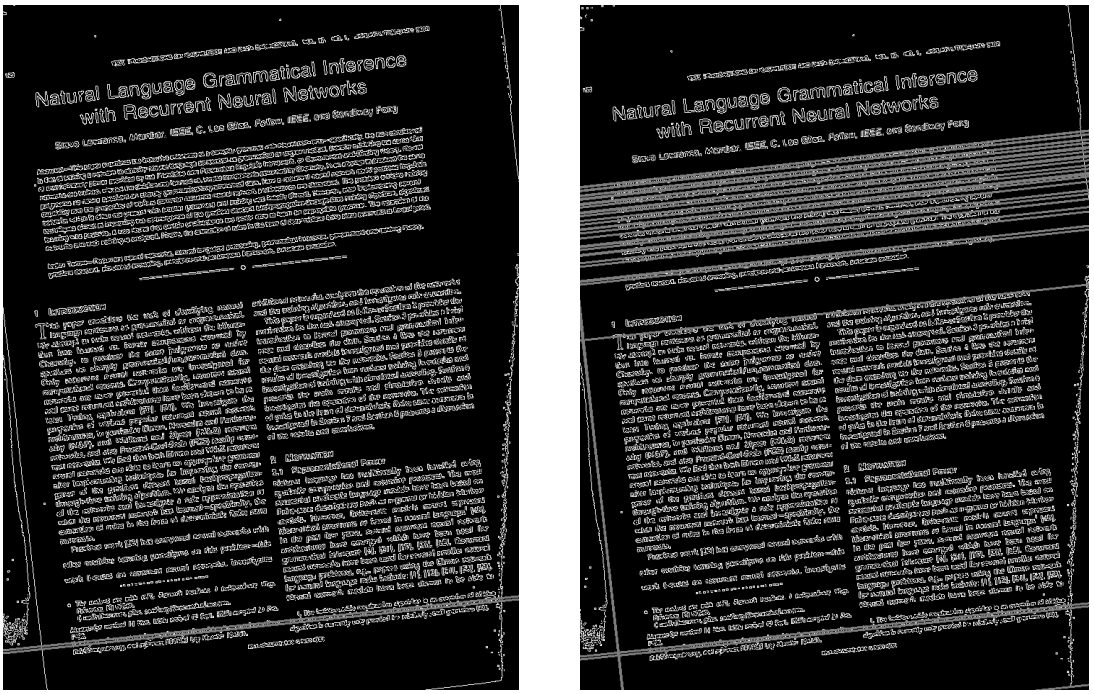

Figura 7: Alteração da Precisão no Método da Transformada de Hough

3.3 Aplicação de OCR

Com a aplicação de OCR nas imagens originais e alinhadas notou-se que o alinhamento é essencial para o funcionamento do reconhecimento óptico de caracteres. Entretanto, pequenos desalinhamentos fazem uma diferença menor, como foi percebido com os resultados obtidos com a imagem *neg_4.png*, na qual a imagem original teve um resultado inferior, mas próximo ao da imagem alinhada. Esses resultados são mostrados na Figura 8.

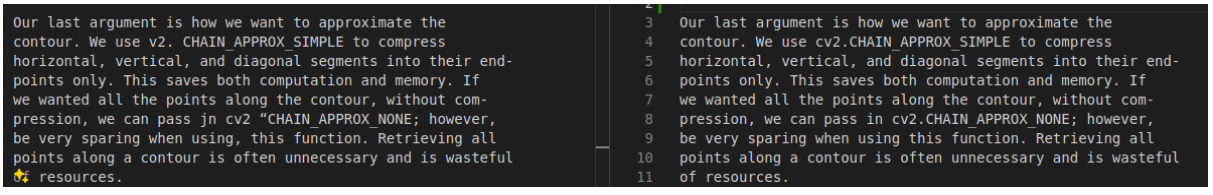

Figura 8: OCR na Imagem *neg_4.png*

No restante das imagens, em que a inclinação é mais significativa, o OCR funcionou de forma satisfatória apenas em imagens alinhadas.

Todos os textos estão salvos na pasta *output_texts* com o mesmo nome de suas imagens referentes.

4. Conclusão

Ao longo da implementação do projeto, foi possível perceber que é difícil resolver todos os problemas para todas as imagens. Notou-se também que métodos de limiarização para alinhamento automático dessas imagens tiveram resultados insatisfatórios, por conta disso foi preferível utilizar a detecção de bordas. Especialmente para a imagem contendo código de barras e a imagem do artigo, os métodos de limiarização testados não obtiveram bons resultados.

Vale ressaltar que o método de projeção horizontal implementado leva mais tempo, mas é menos paramétrico do que a Transformada de Hough da maneira que foi utilizada neste projeto.

Além disso, para ambos os métodos, foi percebido que quanto maior a imagem, maior deve ser a precisão.