|  |  |  |  |
| --- | --- | --- | --- |
|  | **Universidade de Aveiro**  **Ano** 2021/2022 |  | |
| António Jorge  Ferreira Ramos | Validação da autenticidade de documentos impressos | | |
|  |  | | |
|  |  | |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Universidade de Aveiro**  **Ano 2021/2022** |  | |
| António Jorge  Ferreira Ramos | Validação da autenticidade de documentos impressos | | |
|  | Dissertação apresentada à Universidade de Aveiro para cumprimento dos requisitos necessários à obtenção do grau de Mestre em Engenharia Informática, realizada sob a orientação científica do Doutor André Zúquete, Professor Auxiliar do Departamento de Eletrónica, Telecomunicações e Informática da Universidade de Aveiro | | |
|  |  | |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
| o júri |  |
| presidente | Prof. Doutor João Antunes da Silva  professor associado da Universidade de Aveiro |
|  |  |
|  | Prof. Doutor João Antunes da Silva  professor associado da Universidade de Aveiro |
|  |  |
|  | Prof. Doutor João Antunes da Silva  professor associado da Universidade de Aveiro |
|  |  |
|  | Prof. Doutor João Antunes da Silva  professor associado da Universidade de Aveiro |
|  |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
| agradecimentos /  acknowledgements | Agradeço a ajuda dos orientadores, amigos e familiares pelo o incentivo de realizar a tese. |

|  |  |
| --- | --- |
|  |  |
|  |  |
| palavras-chave | Marca de água, códigos de barras 128, documentos impressos, documentos digitais, segurança, processamento de imagem, Qrcode, verificação de integridade |
| resumo |  |

|  |  |
| --- | --- |
|  |  |
|  |  |
| keywords | Watermark, barcodes, textual watermark, printed documents, security, image processing |
| abstract |  |

# Índice

[Índice ii](#_Toc127276809)

[Lista de tabelas iv](#_Toc127276810)

[Lista de Figuras v](#_Toc127276811)

[1. Introdução 1](#_Toc127276812)

[1.1. Descrição do sistema existente 1](#_Toc127276813)

[1.2. Objetivos 1](#_Toc127276814)

[2. Estado da Arte 3](#_Toc127276815)

[2.1. Código de barras 3](#_Toc127276816)

[2.1.1. Análise de código de barras 3](#_Toc127276817)

[2.2. Texto com marca de água 5](#_Toc127276818)

[2.2.1. Ataque zero com mudança de texto 5](#_Toc127276819)

[2.2.1.1. Exemplo 6](#_Toc127276820)

[2.2.1.1. Vantagens 6](#_Toc127276821)

[2.2.1.2. Desvantagens 6](#_Toc127276822)

[2.2.2. Ataque zero sem mudança no texto 6](#_Toc127276823)

[2.2.2.1. Vantagens 7](#_Toc127276824)

[2.2.2.2. Desvantagens 7](#_Toc127276825)

[2.2.3. Documentos texto baseados em Eigenvalues 7](#_Toc127276826)

[2.2.3.1. Testes 7](#_Toc127276827)

[2.2.3.4. Vantagens 8](#_Toc127276828)

[2.2.3.5. Desvantagens 8](#_Toc127276829)

[2.2.4. Line-Shift Coding 8](#_Toc127276830)

[2.2.4.1. Word-Shift Coding 9](#_Toc127276831)

[2.2.4.2. Character Coding 9](#_Toc127276832)

[2.2.4.3. Vantagens dos métodos 9](#_Toc127276833)

[2.2.4.4. Desvantagens dos métodos 9](#_Toc127276834)

[2.2.5. ECL zero-based watermarking 9](#_Toc127276835)

[2.2.5.1. Vantagens 9](#_Toc127276836)

[2.2.5.2. Desvantagens 9](#_Toc127276837)

[3. Qrcode 11](#_Toc127276838)

[4. Código de barras 128 13](#_Toc127276839)

[5. Entropia documento com marca de água 15](#_Toc127276840)

[6. Métodos implementados 17](#_Toc127276841)

[6.1. Primeira implementação 17](#_Toc127276842)

[6.2. Análise de quantidade de qrcode para a verificação de integridade 21](#_Toc127276843)

[6.3. Verificação de integridade com 9 qrcode com processamento de imagem 22](#_Toc127276844)

[7. Proposta de desenvolvimento 26](#_Toc127276845)

[8. Arquitetura da solução 28](#_Toc127276846)

[8.1. Estrutura do documento 28](#_Toc127276847)

[8.2. Base de dados 29](#_Toc127276848)

[8.3. Criação do código de barras e Qrcode 29](#_Toc127276849)

[8.4. Operações a desempenhar pelo utilizador 30](#_Toc127276850)

[8.5. Processamento do documento 30](#_Toc127276851)

[8.6. Retificar documento 30](#_Toc127276852)

[8.7. Verificação de integridade 30](#_Toc127276853)

[8.8. Retificação Manual 31](#_Toc127276854)

[9. Substituição dos Qrcode 34](#_Toc127276855)

[10. Conclusões e futuro trabalho 41](#_Toc127276856)

[11. Referências 43](#_Toc127276857)

[Apêndice A 45](#_Toc127276858)

[OCR 45](#_Toc127276859)

[Iron Software 46](#_Toc127276860)

# Lista de tabelas

[Tabela 1 - Estudo de código de barras lineares 3](#_Toc127276861)

[Tabela 2 - Possíveis escolhas 5](#_Toc127276862)

[Tabela 3 - Resultados Eigenvalues 8](#_Toc127276863)

[Tabela 4 - Classificação de erros do Qrcode 11](#_Toc127276864)

[Tabela 5 - Mudança de cor 18](#_Toc127276865)

[Tabela 6 - Performance 20](#_Toc127276866)

[Tabela 7 – Estudo do número de qrcode a usar no ficheiro pdf 21](#_Toc127276867)

# Lista de Figuras

[Figura 1 – Exemplo de um ataque zero 6](#_Toc127207550)

[Figura 2 - Esquema de geração e extração da watermark 7](#_Toc127207551)

[Figura 3 - Antes do Watermark 7](#_Toc127207552)

[Figura 4 - Depois do Watermark 7](#_Toc127207553)

[Figura 5 - Comparação Watermark 8](#_Toc127207554)

[Figura 6 - Exemplo line-shift 8](#_Toc127207555)

[Figura 7 - Exemplo word-shift 9](#_Toc127207556)

[Figura 8 - Exemplo Character Coding 9](#_Toc127207557)

[Figura 9 - Funcionamento do algoritmo 9](#_Toc127207558)

[Figura 10 – Exemplo de um Qrcode 11](#_Toc127207559)

[Figura 11 - Estrutura Qrcode 11](#_Toc127207560)

[Figura 12 - Código de barra 128 13](#_Toc127207561)

[Figura 13 - Tabela ASCII 13](#_Toc127207562)

[Figura 14 - Zona livre 18](#_Toc127207563)

[Figura 15 - Zonas críticas 18](#_Toc127207564)

[Figura 16 - Ficheiro exemplar 19](#_Toc127207565)

[Figura 17 - Ficheiro processado 19](#_Toc127207566)

[Figura 18 - Resultado Qrcode 20](#_Toc127207567)

[Figura 19 - Resultado consola 20](#_Toc127207568)

[Figura 20 - Verificação de integridade pretendida 21](#_Toc127207569)

[Figura 21 - Segmentos de reta no ficheiro PDF 22](#_Toc127207570)

[Figura 22 - Qrcode com processamento de imagem 23](#_Toc127207571)

[Figura 23 - Pós processamento 24](#_Toc127207572)

[Figura 24 - Resultado processamento de imagem de 9 qrcodes 24](#_Toc127207573)

[Figura 25 - Arquitetura solução proposta 28](#_Toc127207574)

[Figura 26 - Diagrama da Base de dados 29](#_Toc127207575)

[Figura 27 - Algoritmo que conta número de retas que se intersetam 31](#_Toc127207576)

[Figura 28 - Algoritmo interseção retas 31](#_Toc127207577)

[Figura 29 - Exemplo verificação integridade 31](#_Toc127207578)

[Figura 30 – Posições 32](#_Toc127207579)

[Figura 31 - Original 32](#_Toc127207580)

[Figura 32 - Retificar Manual 32](#_Toc127207581)

[Figura 33 - Ficheiro exemplar com círculo X 34](#_Toc127207582)

[Figura 34 - Pontos origem das retas 34](#_Toc127207583)

[Figura 35 - Calcular pontos volta círculo X 34](#_Toc127207584)

[Figura 36 - Posições aleatórias 35](#_Toc127207585)

# Introdução

No âmbito da tese de Mestrado de Engenharia Informática da Universidade de Aveiro, pretende-se que seja desenvolvido um módulo aplicacional a integrar com uma solução de gestão de informação classificada desenvolvida pela empresa iCreate Consulting.

Esta solução prevê a possibilidade de impressão de documentos classificados, devendo ser implementados mecanismos de marcação dos mesmos no sentido de garantir a sua autenticidade.

* 1. Descrição do sistema existente

Estes documentos vão ser tratados em áreas de segurança física destinadas para o efeito, onde para aceder o utilizador têm que ser pessoas idôneas (de confiança) e acesso a documentos com informação classificada deverá ter um certificado periódico.

Cada utilizador tem acesso a um determinado Posto de Trabalho, e aos documentos que foram destinados a esse mesmo Posto de Trabalho.

Por outro lado, esse utilizador só tem permissão para ver documentos até um nível de classificação para o qual está autorizado: por exemplo se só pode ver documentos até ao nível Confidencial, não terá acesso aos documentos Secretos ou de nível de classificação superior.

Apesar de ter acesso a documentos localizados num determinado posto, deve respeitar o princípio da “Necessidade de Conhecer", não devendo consultar indiscriminadamente os documentos a que tem acesso.

* 1. Objetivos

Para assegurar a sua autenticidade propõe-se dois métodos de segurança no documento.

O primeiro é criar um código de barras que permita validar de uma forma explicita se o documento teve origem no sistema, que possível terá mais informações que caracterize o documento, nomeadamente marcas de segurança. A ideia deste primeiro é passar uma pistola que leia o código de barras e averiguar se o código de barras é valido e se a informação do código de barras corresponde ao documento.

O segundo é criar uma watermark segura, única, por documento, gerado por um conjunto de informações do documento, que permitam auxiliar em aspetos mais forenses.

# Estado da Arte

Este capítulo tem como objetivo agregar informações relativas ao tema da tese, sendo divido em dois subcapítulos, códigos de barras e *text* *watermarking*. O subcapítulo código de barras servirá para verificar se um documento é fidedigno, isto é comparar a assinatura do documento com a informação que estará no código de barras. O *text* *watermarking* verificará se a informação do documento foi modificada ou se é o original através de métodos de marca de água*.*

* 1. Código de barras

Código de barras é um modo de representar informação num estado visual, que possa ser visível sem de ter a necessidade de escrever texto. É classificado em duas categorias, linear (1D) e 2D.

Um dos requisitos da empresa era que o código de barras não ocupasse muito espaço no documento, tendo espaço livre no cabeçalho e no rodapé. Como a categoria linear ocupa menos espaço em altura, tendo um comprimento variável, dependendo do número de caracteres contidos na informação do mesmo. Existem várias subcategorias nos códigos de barras, para analisar as mesmas criou-se uma tabela (1), que compara os mesmos, escolhendo o ideal para inserir no documento. Esta tabela é constituída por colunas:

* Nome: designação do código de barras;
* Imagens exemplificativas;
* Comprimento - variável ou fixo;
* Uso;
* Necessidade de aparelho especial para ler.

Em alguns códigos de barras existe um espaço reservado que não se pode alterar que se chama *checksum* ou *check* *digit*, cujo objetivo é verificar se a informação do código de barras foi gerada corretamente.

Existem duas maneiras de ler um código de barras:

1. Máquina própria (leitor de barras de barras)
2. Smartphone (iOS ou android) com uma aplicação que permita a leitura.

Devido as questões de segurança, a empresa proíbe uso de telemóveis pessoais dentro do estabelecimento,

o que levou descartar a hipótese de usar os smartphones como leitores. E leitores de barras especiais, já que têm um custo adicional para a empresa. Para solução optou-se por usar um package em C#, *Iron Bar Code* [1] que permite a leitura de códigos de barras em ficheiros PDF ou imagens (PNG), sendo apenas o reconhecimento em zonas sem distorção, ou seja, sem que apareça texto ou imagens por cima do código de barras.

* + 1. Análise de código de barras

Este subcapítulo tem como objetivo expor a análise de vários códigos de barras lineares, cujos parâmetros de escolha foram comprimento (variável ou fixo), tamanho das barras. A tabela 1 apresenta um conjunto de códigos de barras com as características previamente expostas.

Na tabela 2 encontram-se os possíveis códigos de barras que porventura se possam utilizar. Estes códigos foram selecionados com o critério tamanho máximo de caracteres que este pode guardar, podendo mudar no futuro de acordo com a escolha da empresa. A informação que se pretende guardar nos códigos de barras é a data do documento assinado, o local do posto de trabalho e um identificador único do documento, podendo ser mudado no futuro.

Tabela 1 - Estudo de código de barras lineares

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nome** | **Imagem** | **Fixo/**  **Variável** | **Tamanho das barras** | **Uso** |
| Post Code Austrália |  | Fixo | 4 | Correios Austrália |
| CodaBar |  | Fixo | 2 | Librarias |
| Code 25 |  | Variável | 2 | Librarias |
| Code 11 |  | Fixo | 2 | Telefones, já não se usa muito devido à sua antiguidade |
| Code 32 |  | Fixo | 2 | Farmácia |
| Code 39 |  | Fixo | 2 | Vários |
| Code 49 |  | Variável | Vários | Vários |
| Code 93 | Código de barras Código 93 - Simbologia nos códigos de barras | Cognex | Variável | Vários | Vários |
| Code 128 |  | Variável | Vários | Vários |
| EAN 2 |  | Variável | Vários | Revistas |
| EAN 5 |  | Variável | Vários | Livros |
| EAN-8 |  | Variável | Vários | Retalho |
| GS1-128 |  | Variável | Vários | Vários |
| GS1 Databar |  | Variável | Vários | Vários |
| Intelligent Mail barcode |  | Fixo | 4 | Correios USA |
| ITF-14 |  | Variável | 2 | Encomendas |
| ITF-6 |  | Variável | 2 | Vários |
| JAN |  | Variável | Vários | Usado no Japão |
| Planet |  | Variável | Grande/  Pequeno | Correios USA |
| Plessey | MSI Plessey | Barcode Scanner SDK | Manatee Works | Variável | 2 | Catálogos, revistas, inventários |
| PostBar |  | Fixo | 4 | Correios Canada |
| PostNET |  | Fixo | Grande/  Pequeno | Correios USA |
| RM4SCC/KIX |  | Fixo | 4 | Correios |
| RM Mailmark C |  | Fixo | 4 | Correios |
| Universal Product Code |  | Variável | Vários | Retalho |
| Telepen | Free Online Barcode Generator: Telepen Alpha | Variável | 2 | Librarias  da Inglaterra |

Tabela 2 - Possíveis escolhas

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nome** | **Imagem** | **Tamanho máximo de dados** | **Tipos de dados** | **Vantagens** | **Desvantagens** |
| Code 39 |  | 43 | Letras e números | Ocupa menos espaço. | Não processa caracteres especiais tal como ç. |
| Code 128 |  | 48 | Letras, números e caracteres especiais | Faz encoding de strings muito eficiente.  Tem *checksum* | Precisa de espaço reservado no início, fim e *check symbols*.  Tem pouco limite de informação que pode guardar. |
| Code 93 | Código de barras Código 93 - Simbologia nos códigos de barras | Cognex | 30 | Letras, números e caracteres especiais | Open Source | Não tem checksum |

* 1. Texto com marca de água

A *marca de água* é um método que permite a salvaguarda de documentos originais e verificação da autenticidade de documentos ou imagens. É usada nos dias de hoje como por exemplo nas notas europeias, para impedir a impressão de notas não autorizadas. Existem muitos métodos e ataques de marcação de água, neste documento irá ser abordado o método marca de água de textoe os métodos ataques zero devido às necessidades apresentadas pela empresa, contudo poderá mudar consoante o desenvolvimento da tese.

* + 1. Ataque zero com mudança de texto

O objetivo dos ataques zero é aumentar a fragilidade da marca de água do documento contra ataques de pessoas não autorizadas, normalmente denominados por hackers. Estes ataques tendem a alterar o texto do documento sem alterar a marca de água. Estes ataques normalmente são de inserção, reordenação e remoção de texto.

Uma das soluções propostas pelos autores do artigo [2] foi ataque de substituição que explora a troca de ordem das palavras no texto. Esta solução é constituída por dois elementos:

1. Lista de palavras que se pretende substituir no documento.
2. Lista de palavras contidas no documento depois da substituição.

Em suma, escolhe-se um conjunto de palavras no documento que se quer substituir por novas. Quando uma palavra é selecionada para se substituir todas as ocorrências da mesma são substituídas pela palavra nova. A percentagem do ataque determina a quantidade de modificação do documento. Se a palavra escolhida tiver várias ocorrências no documento, a percentagem vai aumentar.

A percentagem da contagem de palavras no documento é dada pela equação 1. A percentagem do ataque de substituição é dada pela expressão da equação 2.

Existem duas maneiras de escolher a lista de palavras para a substituição, normal e avançada. A normal consiste em selecionar palavras aleatórias do documento. A avançada selecionar palavras com mesmo tamanho do que as palavras da lista de substituição.

Em suma, a marcação de água textual proposto por [2] consiste em 3 passos, selecionar palavras do documento para substituição, selecionar as novas palavras e ocorrência de implementação no texto do documento.

Text

Description automatically generated

Equação 1

Text, letter

Description automatically generated

Equação 2

* + - 1. Exemplo

Conforme mencionado no artigo [2] a figura 1, demonstra um exemplo de tipos de substituição que podem ser usados para mascarar informações importantes em um texto. O exemplo demonstra a substituição da palavra "The", que é o pronome mais comum em inglês, por "close" na substituição normal e por "job" na substituição avançada.

A ideia por trás desse método é permitir que palavras-chave ou informações importantes não sejam perdidas, enquanto ainda são mascaradas para outras pessoas que não têm acesso às palavras originais. A substituição de palavras é realizada de tal forma que apenas quem possui o conhecimento necessário para substituir as palavras é capaz de entender o significado original do texto.

Este método pode ser útil em várias situações, como na proteção de informações sensíveis em documentos, comunicações confidenciais, entre outros. No entanto, é importante notar que este método não é infalível.



Figura 1 - Exemplo de um ataque zero

* + - 1. Vantagens

Uma das vantagens do método de substituição descrito é permitir a alteração de palavras em um texto por um utilizador fidedigno e, ao mesmo tempo, permitir que o algoritmo saiba em qual zona do documento as alterações foram feitas.

Isso pode ser útil em situações em que uma equipe de revisão precisa fazer alterações em um documento, como em um processo de edição de textos. Ao usar esse método, a equipe de revisão pode fazer as alterações necessárias sem perder informações importantes ou prejudicar a integridade do documento. Além disso, o algoritmo pode identificar facilmente as alterações realizadas e rastreá-las para fins de auditoria e verificação de alterações autorizadas.

* + - 1. Desvantagens

Uma das principais desvantagens é que as mudanças no texto podem não fazer sentido para pessoas que não estão familiarizadas com a substituição de palavras usada. Isso pode tornar a leitura do texto difícil e confusa, especialmente se muitas palavras forem substituídas. Se a substituição for realizada de forma excessiva, o texto pode se tornar ininteligível.

Além disso, o método de substituição pode não ser capaz de detetar métodos de cópia de texto, como copiar e colar o texto em um novo documento. Se o novo documento for criado com base no texto original antes da substituição, as informações confidenciais podem ser utilizadas sem que a substituição tenha qualquer efeito.

Outra limitação é que esse método de substituição não garante a segurança completa dos dados, pois é possível que alguém com conhecimento suficiente possa decifrar a substituição e utilizar as informações originais

* + 1. Ataque zero sem mudança no texto

A solução proposta no artigo [3] consiste em usar as características do documento para gerar a marca de água com o seguinte layout “autor:watermark:data:tempo” registada pela uma autoridade certificada (CA). O objetivo principal deste método é usar as palavras que tenham um tamanho maior do que quatro caracteres para proteger contra possíveis ataques.

A escolha de palavras maiores do que quatro caracteres é devido a generalidade dos ataques serem direcionados a palavras com tamanho menor do que quatro caracteres.

Para gerar amarca de água, é necessário saber todas as frases de um documento, e assim percorrer cada palavra achando as palavras maiores do que quatro caracteres, quando acabar, guardar o primeiro caracter de cada dessas palavras e por fim gerar a marca de água.

Por exemplo na seguinte frase: “O José gosta muito de ler” a marca de água ficava JGM (José Gosta Muito). Depois de percorrida todas as frases juntasse as marcas de água de cada frase, dando a marca de água final.

A figura 2 demonstra como é que a watermark é gerada e extraída. Deteta-se um ataque quando o padrão da marca de água tem pelo menos ser 70% igual ao documento comparativo.

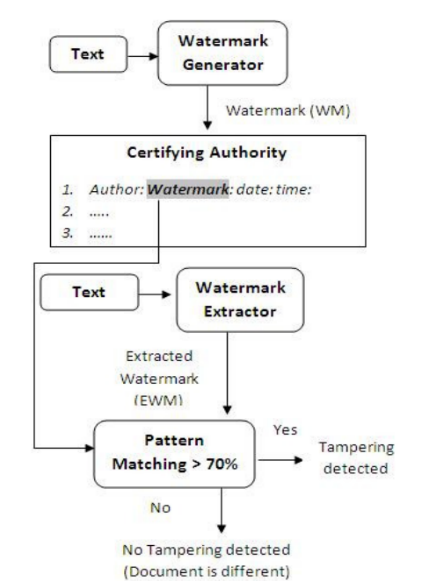


Figura 2 - Esquema de geração e extração da marca de água

* + - 1. Vantagens

Tem uma entidade certificadora que vai possuir a marca de água original, que é a única pessoa que pode comparar os documentos, permitindo assim que o documento esteja só na posse de uma pessoa, diminuindo os ataques ou distribuição do documento. A composição da entidade certificadora é similar ao pretendido no código de barras.

* + - 1. Desvantagens

Tem testes realizados com ataques aleatórios, como inserir, apagar, reordenar e alterar, que para textos reais os resultados poderão ser diferenciados, levando com que o comportamento do algoritmo seja diferenciado entre documentos, dificultando a sua implementação para a retificação da integridade do documento.

* + 1. Documentos texto baseados em Eigenvalues

O artigo [4] propõe um algoritmo que guarda as posições de todas as palavras do documento numa matriz e o peso das mesmas em ASCII. Um documento normalmente é constituído por palavras, espaçamento, números e pontuações. Os autores consideram cada contagem para calcular o peso ASCII para gerar o esquema da marca de água com base numa chave privada, que um utilizador fidedigno retifica o documento recebido.

* + - 1. Testes

No artigo [4] existe testes realizados a um documento com 47 linhas, não tendo disponibilizado o mesmo, apenas prints de pequenas secções do mesmo.

A figura 3 demonstra um excerto do texto antes de ter a marca de água, e na figura 4 demonstra o texto depois da geração da marca de água*.* Para simular um ataque foi alterada a palavra “OFF” para “ON” numa zona do texto e pelos resultados binários na figura 5, sendo que em primeiro está o documento original e de seguida o alterado, retira-se que os números são diferentes. Uma pequena alteração da palavra levou à mudança de 11 bits.

Os autores realizaram testes na mudança de vogais, consoantes, caracteres especiais, palavras, números pontuações, e alterações aleatórias no texto, obtendo os resultados da tabela 3, podendo retificar que na coluna “tamper detection” está tudo a 100% que leva a retificar que o algoritmo detetou alterações em todos os casos.

Text

Description automatically generated

Figura 3 - Antes da Watermark

Text

Description automatically generated

Figura 4 - Depois da Marca de água

Text

Description automatically generated

Figura 5 - Comparação Marca de água

Tabela 3 - Resultados Eigenvalues

Table

Description automatically generated

* + - 1. Vantagens

Guarda a posição das palavras do texto numa matriz. Tem uma identidade certificadora que guarda a marca de águae é a única pessoa que pode verificar se o documento é original ou falsificado.

* + - 1. Desvantagens

Difícil de compreensão, a execução do algoritmo leva muito tempo de execução e processamento para textos grandes.

* + 1. Line-Shift Coding

No artigo [5] expuseram outra forma de fazer marcação de água denominado, *Line-Shift Coding*, que consiste em mover as linhas de texto de um documento para cima ou para baixo, enquanto as linhas adjacentes não são movidas. Na figura 6, demonstra-se um exemplo. Neste exemplo a linha do meio começada por “Effects…”, foi movida para baixo 1/300 inches, que equivale a 0.00846666667 cm. Esta mudança não é percetível ao olho humano, só através de um OCR, Optical Character Recognition, é que se conseguiria comparar ficheiros para verificar se existe ou não mudança nas linhas.

A screenshot of a computer

Description automatically generated with medium confidence

Figura 6 - Exemplo line-shift

2.2.4.1. Word-Shift Coding

No mesmo artigo [5], previamente exposto, expuseram outra forma de fazer marcação de água denominado *Word-Shift Coding*, que consiste em mover as palavras para a esquerda ou para a direita, enquanto as palavras adjacentes não são alteradas. A figura 7, tem-se um exemplo. A segunda linha, contém quatro palavras movidas com espaçamento de 1/150 inch, que equivale a 0.0169333333 cm, enquanto na primeira linha não se altera, a terceira é uma junção das duas anteriores. Como referido no ponto anterior esta mudança não percetível a olho humano, e para retificar a marca de água recorrer a um OCR.



Figura 7 - Exemplo word-shift

2.2.4.2. Character Coding

No mesmo artigo [5], previamente exposto, deram outra forma similar, referidas anteriormente, mas desta vez consiste em mover o caracter para cima ou baixo, enquanto adjacentes não se alteram. A figura 8, apresenta um exemplo. A primeira letra “e” da palavra “internet” foi movida para baixo 1/600 inches que equivale a 0.00423333333 cm. Como foi referido anteriormente noutros capítulos é necessário um OCR, para retificar as mudanças feitas no texto, isto é dispendioso em termos de tempo, e difícil de perceção.

A picture containing text

Description automatically generated

Figura 8 - Exemplo Character Coding

2.2.4.3. Vantagens dos métodos

Tem marca de água invisível, ou seja, não é percetível para os olhos de um humano.

2.2.4.4. Desvantagens dos métodos

Uso de OCR para verificar documento, e ter acesso ao texto para inserir o espaçamento. Erros de impressão, por exemplo faltar letras no documento, pode levar à mal classificação de um documento, por exemplo ser falso, quando se tem a certeza de que o documento é original e não foi modificado.

* + 1. ECL zero-based watermarking

O artigo [6], propõe um tipo de marcação de água denominado *ECL zero-based watermarking* que consiste num algoritmo que mantém o conteúdo original do texto do documento e constrói a *marca de água* pela seleção de caracteres no documento. A marca de água é guardada num sítio de confiança que se denomina *Certifying Authority*, para quando houver a necessidade de comparar o documento com a *marca de água* para verificação da sua autenticidade, perceber que se o documento é original ou falsificado. Os autores forneceram uma imagem (figura 9) em que se demonstra como funciona o algoritmo.

Diagram

Description automatically generated

Figura 9 - Funcionamento do algoritmo

2.2.5.1. Vantagens

Tem uma identidade certificadora que garante que o documento original esteja seguro com uma pessoa confiável na empresa.

2.2.5.2. Desvantagens

Falta de testes em cópias de documentos. Pouco efetivo com documentos que tenham um *effectiveness* *ratio* (ER) menor que 0.1. Effectiveness Ratio determina quais os caracteres para a geração da marca de água.

# Qrcode

Atualmente vemos qrcode, nas faturas, em cartões de empresas e até mesmo na rua, maior parte das pessoas já viu, mas maior parte pode não saber o processo de leitura ou como surgiu o mesmo, para isso dediquei este capítulo ao qrcode.

O Qrcode (figura 10) surgiu em 1994 pela empresa Denso Wave, onde originalmente foi criado para categorizar peças de automóvel. Os Qrcode podem ter links para páginas web, texto, um endereço geográfico, uma imagem, um vídeo ou contacto telefónico.

Relativamente à estrutura do qrcode é constituído por 3 quadrados de deteção de posição (4.1. Figura 11) que permite a leitura em várias posições do scanner ou de um smartphone com câmara.

Por um padrão alinhamento (4.2. Figura 11) que corrige a distorção do qrcode em superfícies curvadas, o seu número varia consoante a informação contida.

Por padrões de temporização (4.3. Figura 11) que permite obter o tamanho da matriz de dados. Por informações da versão, que indica que versão do qrcode está a ser utilizada (1. Figura 11).

Por informações de formato, que contém informações sobre a tolerância de erros e o padrão da máscara de dados. Por códigos de dados e erros que podem ser do tipo L, M, Q, H, cujos estão apresentados na tabela 6.

Qr code

Description automatically generated

Figura 10 - Exemplo Qrcode

Graphical user interface, qr code

Description automatically generated

Figura 11 - Estrutura Qrcode

Tabela 4 - Classificação de erros do Qrcode

|  |  |
| --- | --- |
| **Nível de correção de erros** | **Percentagem de área danificada (%)** |
| L (Low) – Baixo | 7 |
| M (Medium) – Médio | 15 |
| Q (Quartile) - Quartil | 25 |
| H (High) – Alto | 30 |

1. Código de barras 128

O código de barras 128 (figura 12 [7]) é um tipo de códigos de barras linear que permite ter 128 caracteres da tabela ASCII (figura 13).

É constituído por zonas quietas 1:figura 12, por símbolo de início 2:figura 12, por dados que são colocados no qrcode 3:figura 12, por um símbolo de verificação que serve para verificar se o barcode foi bem gerado 4:figura 12, e por último um símbolo de stop, onde acaba o código de barras.

A largura do código de barras é dada pela quantidade de informação que este tem no ponto 3:figura 12.



Figura 12 - Código de barra 128

Shape

Description automatically generated with medium confidence

Figura 13 - Tabela ASCII

# Entropia documento com marca de água

O presente projeto de tese apresenta estudos e desenvolvimento de documentos com marca de água, já que estes documentos são confidenciais, ou seja, não se pretende que qualquer pessoa que não seja autorizada os veja. Contudo no mundo atual existem pessoas que pretendem descobrir os documentos mais confidenciais e os divulgar, ou até mesmo, alterar para outros efeitos. Um programa de segurança deve evitar ao máximo que o algoritmo seja descoberto, ou seja, deve garantir que a entropia do algoritmo seja menor possível.

Entropia é definida como uma forma de medir o grau de incerteza de um sistema, ou seja, quanto mais um sistema for incerto menos probabilidade tem se de descobrir como um sistema ou algoritmo funciona. Para isso é necessário que o algoritmo desenvolvido seja mais discreto e mais aleatório possível.

# Sistema de verificação de documentos impressos ou digitais

A primeira intenção para a criação da solução era o uso dos Qrcode, contudo retificou-se que em scans, podiam desaparecer e influenciava na visualização do documento. Apesar de serem implementados várias maneiras de contornar este problema, como o processamento de imagem, que reconhece os Qrcode no texto (através de substituição de cores), posições aleatórias, número de Qrcode a colocar no documento e tamanho do Qrcode foram descartadas devido ao elevado tempo de processamento no caso do processamento de imagem e à elegibilidade do documento.

O sistema desenvolvido envolve duas ações distintas, processamento e retificação do documento.

O processamento do ficheiro envolve utilizar códigos de barras, para a retificação rápida de um documento que seja aceite na base de dados através do utilizador, e posições aleatórias no documento para guardar letras que resultam de interseções de retas.

A retificação do documento/verificação de integridade/análise forense consiste em determinar se a informação do código de barras corresponder ao documento, se não procedesse a uma análise mais profunda através das interseções de retas para verificar se as letras que aparecem no sistema são iguais ou não aquelas que aparecem no documento.

* 1. Arquitetura

De acordo com o descrito anteriormente é criada a arquitetura da figura 14, onde surge 3 camadas.

* Servidor destinado à base de dados que irá conter dados armazenados acerca dos ficheiros, códigos de barras, segmentos de reta traçados, posição dos caracteres no ficheiro de input e a geração da marca de água.
* Programa onde acontece o desenvolvimento da criação do código de barras, do processamento do ficheiro para obtenção das posições dos caracteres no documento, da retificação do documento, e verificação de integridade, esta também é responsável pelas conexões entre camadas, desencadeadas pelo utilizador.
* Utilizador é responsável por agregar as ações que o utilizador pode fazer como por exemplo processar o documento ou retificar.

Diagram

Description automatically generated

Figura 14 - Arquitetura solução

* 1. Estrutura do documento

Para a utilização de testes do programa foi disponibilizado pela empresa 4 documentos diferentes em formato PDF e digitais. Cabe à empresa inserir manualmente os ficheiros na diretoria correspondente na inicialização do programa e os metadados no programa para tratar os mesmos. É de salientar que os documentos disponibilizados têm uma classificação de segurança mínima pelo que se pode partilhar os resultados do mesmo.

O algoritmo lida com três tipos de documentos:

- Documento sem marca de água;

- Documento com marca de água;

- Documento (scan) com marca de água;

* 1. Base de dados

O intuito de usar base de dados no sistema é garantir segurança sobre as informações do documento através do uso de código de barras, como se referiu anteriormente, o código de barras tem um id aleatório representante das características do documento cujo é inserido na base de dados aquando do seu processamento do documento para a criação da marca de água.

Existem dois tipos de base de dados as relacionais e não relacionais (NoSQL). [8]

As bases de dados relacionais guardam os dados nas tabelas, tendo algumas delas partilha de informação, causando uma relação entre tabelas.

Cada tabela contém colunas que definem a informação que se pode guardar, e linhas que contém a informação.

Normalmente a tabela contém um identificador único que referencia cada linha chamado de chave primária (primary key), caso se queira referenciar os valores a outra tabela utiliza-se a chave estrangeira (foreign key), que obrigatoriamente tem de existir previamente.

A linguagem que se usa para tratar base de dados relacionais é SQL, existem vários programas que permitem correr SQL, como por exemplo mySQL [9], Oracle SQL Developer [10] e Microsoft SQL Server Management Studio 2018 [11].

As bases de dados não relacionais, não usam tabelas relacionais, em vez disso faz cria grupos em se que guarda a informação em informações diferentes.

Como o objetivo do trabalho é sempre perceber que documento é qual, é necessário haver relações entre a marca de água e o documento, então optou-se por usar uma base de dados relacional com o auxílio da ferramenta Microsoft SQL Server Management Studio 2018.

A base de dados SQL foi criada localmente, tendo um utilizador e base de dados dedicado apenas à consulta de informações pelo programa. As informações que se guardam são as características do documento, o código de barras, segmentos de reta traçados entre pontos, posições dos caracteres no documento de entrada (processamento apenas), e a criação da marca de água (aceitação ou rejeição).

Na figura abaixo, está presente um diagrama da base de dados que contém as tabelas usadas e respetivas conexões.

O diagrama é constituído por as seguintes tabelas:

* “document”: guarda características do documento (metados);
* “barcode”: guarda informações relativas ao documento;
* “watermark”: guarda as confirmações do documento com marca de água, se foi aceite ou não, para efeitos de rastreamento;
* “forense\_analises”: guarda os segmentos de reta traçados entre dois pontos, o ponto de interseção e a letra que aparece no ponto de interseção para efeitos de verificação de integridade do documento;
* “position\_char\_file”: guarda as posições dos caracteres no documento.

Diagram

Description automatically generated

Figura 15 - Diagrama da Base de dados

* 1. Criação do código de barras

Para a criação do código de barras 128 foi utilizado um package para C# denominado Iron Barcode, abordado mais detalhadamente no Anexo.

Como referido anteriormente o código de barras irá ter um id referenciador para o documento e para as posições utilizadas para traçar os segmentos de reta no documento.

O código de barras 128 irá ser colocado sempre no cabeçalho da primeira página devido a ser uma zona livre do documento, e permitir maior parte das vezes a sua leitura.

* 1. Operações desempenhar pelo utilizador

Um utilizador exerce 4 tipos de funções.

* Escolher entre
  + Processamento: que processa o documento sem a marca de água para originar a marca de água;
  + Retificar: que retifica o documento de input com a marca de água para verificar se é o documento que se apresenta ser.
* Aceitar ou rejeitar o documento com marca de água;
* Comparar a informação do documento com a marca de água no submenu retificar, para determinar a sua autenticidade;
* Verificação de integridade, para determinar a zona da alteração do documento, se por venturar o documento não ser autêntico, através da comparação de informações que aparecem na aplicação versus o documento.
  1. Processamento do documento

Conforme mencionado anteriormente o utilizador escolhe o ficheiro a processar, o algoritmo só aceita caso o ficheiro não tiver na base de dados e se o nome do mesmo não conter “watermark” que se adiciona ao nome do ficheiro já existente para diferenciar os ficheiros sem e com marca de água, também se adicionou o dia e a hora que se executou o processamento do ficheiro levando ao nome final do ficheiro ser “nome\_ficheiro\_watermark\_dd\_mm\_yy\_hh\_m\_ss”, sendo dd:dia, mm:mês, yy:ano, hh:hora, m: min, ss:segundos. Antes de processar, caso o utilizador queira visualizar se o ficheiro selecionado é pretendido, o algoritmo mostra o ficheiro numa janela nova com os botões processar, aceitar e rejeitar (figura 16). Caso o utilizador clique em aceitar ou rejeitar o ficheiro sem antes de clicar “processar” aparece o erro da figura 17.

Consoante o clique “processar” o programa vai abrir uma consola que vai executar um ficheiro jar, fechando automaticamente depois da execução, que vai retirar os caracteres do ficheiro e as respetivas posições no documento. Para retirar as posições o algoritmo foi realizado em Java porque em C# não foi encontrado packages que retirassem a posição com certeza dos caracteres no documento (figura 18).

O processamento demora à volta de 2 min, colocando a janela do processamento bloqueada até que as ações de calcular os pontos, obtenção dos pontos de interseção e letras, inserções na base de dados sejam concluídas.

Quando o processamento acaba o utilizador é abordado com uma mensagem da figura 19 para aceitar ou rejeitar o documento que se pode pré visualizar figura 20. Caso o documento seja aceite o utilizador recebe a mensagem “Documento Aceite!” figura 21, por ventura se tentar rejeitar ou aceitar de novo o documento é informado de que o documento já foi aceite (figura 22). Caso rejeite o documento é guardado, e o processamento é feito novamente. Caso feche a aplicação sempre poderá consultar o documento que foi gerado para a diretoria que abriu o documento na extensão previamente exposta. A escolha é feita no menu principal quando se escolhe a opção pretendida. É aberto a diretoria predefinida da execução (figura 23) e pode-se mudar consoante o necessário, tendo a noção que ficheiros fora da pasta do algoritmo podem dar origem a erros.

Graphical user interface, text

Description automatically generated

Figura 16 - Processamento exemplo

Graphical user interface, application, Word

Description automatically generated

Figura 17 - Erro ao clicar em aceitar ou rejeitar sem antes de processar o ficheiro

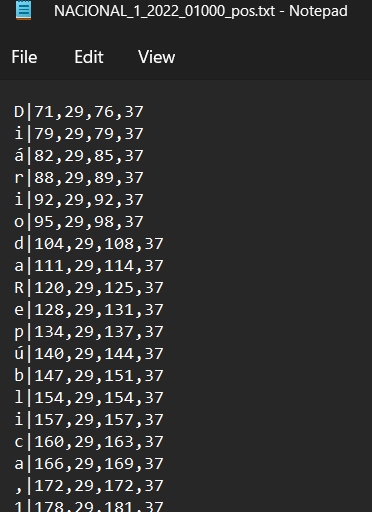


Figura 18 - Reconhecimento caracteres

Graphical user interface, text, application

Description automatically generated

Figura 19 - Finalização do processamento

Graphical user interface, text, application, email

Description automatically generated

Figura 20 - Finalização da marca de água

Graphical user interface, application, chat or text message

Description automatically generated

Figura 21 - Mensagem Documento Aceite

Graphical user interface, text, application, chat or text message

Description automatically generated

Figura 22 - Mensagem documento já aceite

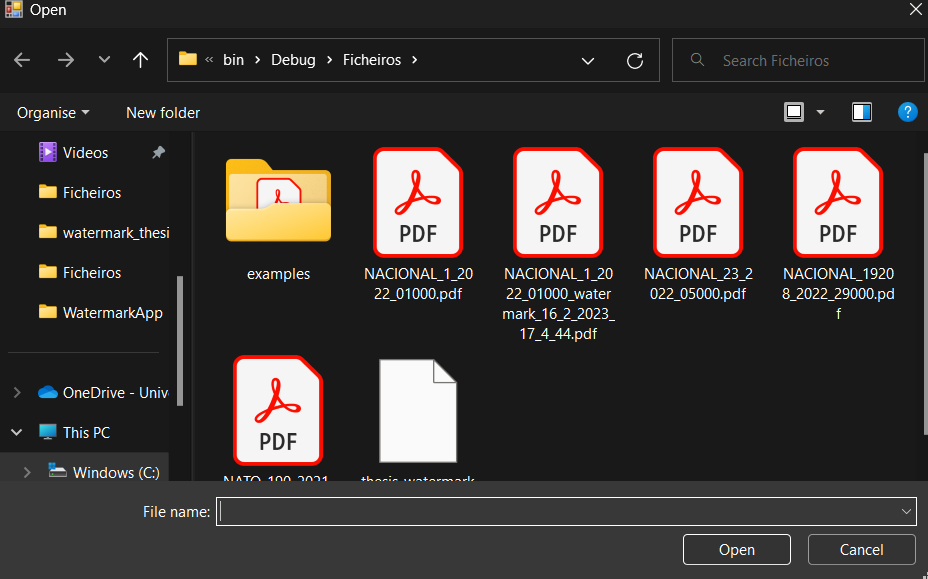


Figura 23 - Abrir Documento

* 1. Retificação do documento

Para retificar um documento é necessário que o documento escolhido para retificação contenha a marca de água e cuja informação esteja na base de dados. Caso esteja, o programa, lê o código de barras 128 e mostra as informações contidas para o utilizador numa janela onde é possível ter a visualização do documento com a marca de água escolhido e algumas informações sobre as características do documento (metadados) (figura 24). Cabe ao utilizador averiguar se o documento e as características demonstradas são diferentes ou iguais, se determinar que são diferentes o utilizador pode proceder à verificação de integridade do documento, referido no ponto 6, determinando as zonas que foram alteradas no documento.

Graphical user interface, text, email

Description automatically generated

Figura 24 - Retificação documento

* 1. Verificação de integridade

A verificação de integridade ou análise forense, serve para retificar se o documento foi alterado, e em que zonas.

A solução para retificar um documento, é criar 3 segmentos de retas calculados através dos pontos criados nas extremidades da folha 9 (3 3) pontos, totalizando 81(9 9) pontos.

Para saber o número total de retas usou-se a expressão somatória do Gauss [12] (equação 1). O resultado da equação dá igual ao valor no código através do sublinhado a preto na figura 25.

Equação 1 - Gauss

Aquando das retas traçadas vai se calcular o ponto de interseção da mesma [13] , com a expressão da figura 26, de seguida com base nas coordenadas dos caracteres obtidos no processamento do ficheiro verifica-se se o ponto pertence ao subconjunto das coordenadas, dando como output a letra correspondida.

Na figura 27 mostra um possível output da verificação da integridade , sendo o círculo amarelo o ponto de interseção e a azul o caracter que está na base de dados, extraído no processamento.

Text

Description automatically generated

Figura 25 - Algoritmo que conta número de retas que se intersetam

Graphical user interface, text, application, email

Description automatically generated

Figura 26 - Algoritmo interseção retas

Text

Description automatically generated

Figura 27 - Exemplo verificação integridade

* 1. Retificação Manual

A posição do Qrcode do documento digitalizado pode ser alterada de acordo com a posição do papel no scanner. Para não afetar a verificação de sanidade, foi criada uma função que permite ao usuário escolher a posição do código QR.

Para entender a diferença entre retificação manual e automática, dois testes são necessários.

A Figura 30 mostra o conjunto de posições que serão utilizadas, as chamadas "originais" referentes às correções automáticas, "clicadas" referentes às posições selecionadas pelo usuário no programa, e "alfa" indicando que a correção da posição corrige a posição no documento digitalizado com base no número de cliques e na localização original muda.

No primeiro teste com um documento exemplificar, retirei uma zona do documento que está presente na figura 31 onde se pode visualizar o resultado da retificação automática.

No segundo teste realizado ao mesmo documento do primeiro, removi a mesma zona do documento e conclui-se que nesta tem-se mais interseções, por exemplo no sublinhado “de” da figura 31 não existe nenhuma interseção, mas na 32 a mesma aparece com duas interseções no “d”.

Embora se tenha ponderado a remoção de letras duplicadas em interseções, verificou-se que era um processamento muito lento, já que se tinha de verificar condições muito específicas, e apesar de ilegíveis, percebe-se que a obtenção da letra funciona, pelo que o utilizador consegue inferir a letra através do círculo amarelo e das várias sobreposições da mesma letra.

Graphical user interface, text, application

Description automatically generated

Figura 28 – Posições

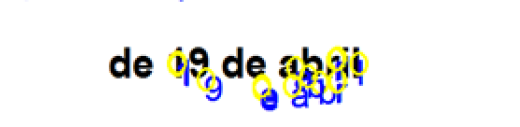


Figura 29 - Original



Figura 30 - Retificar Manual

Text

Description automatically generated

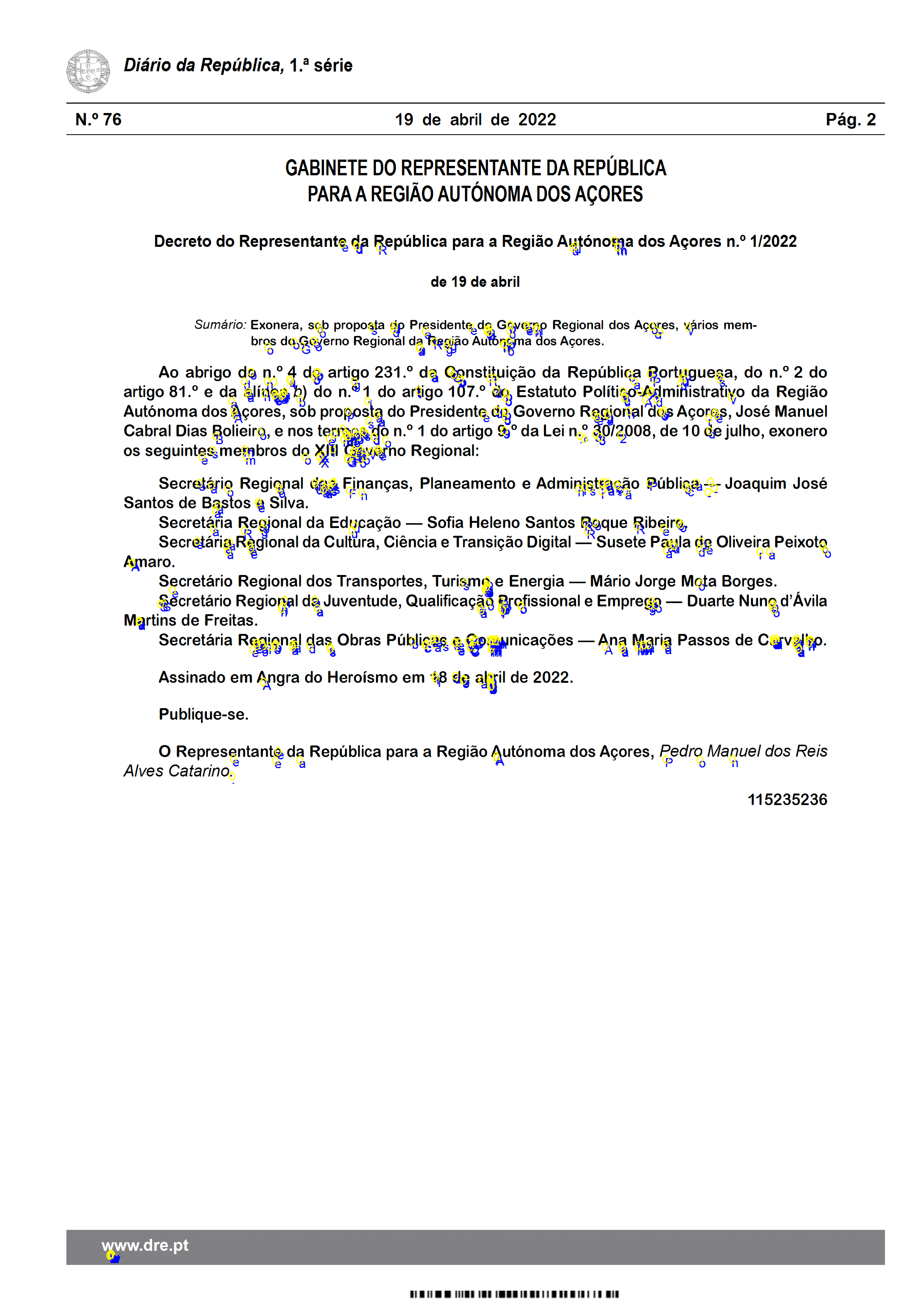
b

00:00:05 segundos

Text

Description automatically generated

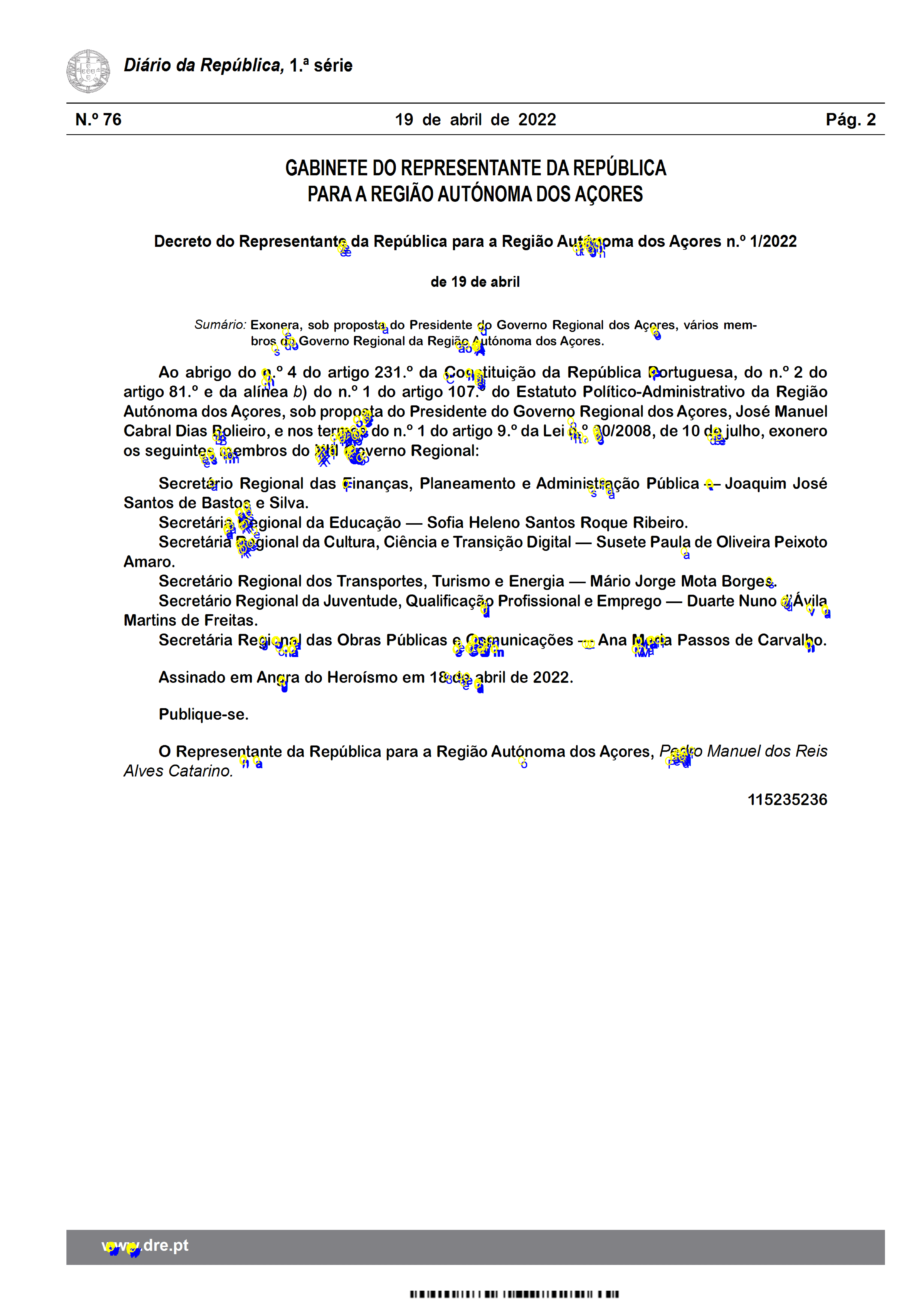
00:02:37



Text

Description automatically generated

00:01:30



Quanto mais próximos os pontos do ponto de origem, menos interseções resultam e menos tempo de execução.

# Conclusões e futuro trabalho

# Referências

[1] “C# .NET Barcode Quickstart Guide, Code Examples | Iron Barcode.” https://ironsoftware.com/csharp/barcode/examples/barcode-quickstart/ (accessed Jul. 26, 2022).

[2] M. B. Mohd, S. Mohd, R. Tanzila, and S. A. Rehman, “Replacement Attack: A New Zero Text Watermarking Attack,” *3D Res.*, vol. 8, 2017, doi: 10.1007/s13319-017-0118-y.

[3] Z. Jalil, A. M. Mirza, and H. Jabeen, “Word length based zero-watermarking algorithm for tamper detection in text documents,” *ICCET 2010 - 2010 Int. Conf. Comput. Eng. Technol. Proc.*, vol. 6, no. May, 2010, doi: 10.1109/ICCET.2010.5486185.

[4] T. Rethika, I. Prathap, R. Anitha, and S. V. Raghavan, “A novel approach to watermark text documents based on eigen values,” *2009 Int. Conf. Netw. Serv. Secur. N2S 2009*, no. c, pp. 1–5, 2009.

[5] J. T. Brassil, S. Low, and N. F. Maxemchuk, “Copyright protection for the electronic distribution of text documents,” *Proc. IEEE*, vol. 87, no. 7, pp. 1181–1196, 1999, doi: 10.1109/5.771071.

[6] T. Saba, M. Bashardoost, H. Kolivand, M. S. M. Rahim, A. Rehman, and M. A. Khan, “Enhancing fragility of zero-based text watermarking utilizing effective characters list,” *Multimed. Tools Appl.*, vol. 79, no. 1–2, pp. 341–354, Jan. 2020, doi: 10.1007/S11042-019-08084-0/TABLES/5.

[7] “Code 128 - Wikipedia.” https://it.wikipedia.org/wiki/Code\_128 (accessed Jan. 19, 2023).

[8] “Relational Vs. Non-Relational Databases | MongoDB | MongoDB.” https://www.mongodb.com/compare/relational-vs-non-relational-databases (accessed Jan. 19, 2023).

[9] “MySQL.” https://www.mysql.com/ (accessed Jan. 19, 2023).

[10] “Oracle SQL Developer Downloads.” https://www.oracle.com/database/sqldeveloper/technologies/download/ (accessed Jan. 19, 2023).

[11] “SQL Server Management Studio (SSMS) - SQL Server Management Studio (SSMS) | Microsoft Learn.” https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16 (accessed Jan. 19, 2023).

[12] “The Story of Gauss - National Council of Teachers of Mathematics.” https://www.nctm.org/Publications/TCM-blog/Blog/The-Story-of-Gauss/ (accessed Feb. 08, 2023).

[13] “Line–line intersection - Wikipedia.” https://en.wikipedia.org/wiki/Line–line\_intersection (accessed Oct. 16, 2022).

[14] “About Us | IronSoftware.com.” https://ironsoftware.com/about-us/ (accessed Jul. 26, 2022).

[15] “C# Tesseract OCR In 1 Line of Code | Iron OCR.” https://ironsoftware.com/csharp/ocr/examples/simple-csharp-ocr-tesseract/ (accessed Jul. 26, 2022).

Apêndice A

OCR

Iron Software

Para desenvolver o programa foi necessário usar tecnologias de criação de Qrcode e leitura de Qrcode. Para isso usou-se um software open-Source, que tem a opção de comprar a licença para efeito de segurança e apoio, denominado Iron Software

[14]. O Iron software é usado por mais de 6.9 milhões de utilizadores tendo vários produtos como o Iron PDF, Iron OCR, Iron OCR, Iron Barcode, Iron XL, Iron Updates e Iron suite, listados na imagem abaixo. Sendo que neste projeto se usou o Iron OCR para leitura do qrcode e Iron barcode para gerar o qrcode.

Optou-se por usar o Iron Barcode [1] para gerar o qrcode devido a fornecer documentação e pelas propriedades de geração do qrcode, permitindo mudar o estilo do qrcode, como por exemplo a cor do qrcode, também permite colocar o qrcode no ficheiro PDF, mas eu optei por usar outro método, já que alguns documentos PDF são protegidos e neste pacote não tem opção de escrever neles sem saber a password.

Para ler/detetar o qrcode no ficheiro PDF optou-se por usar o Iron OCR [15], cujo procura num documento PDF o qrcode, e devolve o resultado.

