Implementação do *Image Matching*: uma nova solução para um problema NP-completo

1st Levy Santiago
Universidade Federal da Bahia (UFBA)
Salvador, Brasil
levy.santiago@ufba.br

2nd Marcos Pinheiro
Universidade Federal da Bahia (UFBA)
Salvador, Brasil
marcos.pinheiro@ufba.br

3rd Rita Barretto
Universidade Federal da Bahia (UFBA)
Salvador, Brasil
rita.barretto@ufba.br

***Resumo*—A questão P $\neq$ NPC é intrigante e complexa para a teoria da ciência da computação, uma vez que ainda não foi descoberto nenhum algoritmo de tempo polinomial para um problema NP-completo, bem como não foi provado que não pode existir nenhum algoritmo de tempo polinomial para eles. Diante desse cenário, este trabalho aborda o problema NP-completo *Image Matching* (IM) e traz uma nova forma de implementação e resolução para ele. A metodologia adotada parte do mapeamento dos *pixels* das imagens e subsequente associação das semelhanças entre elas, considerando o deslocamento de *pixels*, a partir da mencionada nova implementação. Por fim, é definido como será realizada a avaliação dos resultados a serem obtidos pelo novo algoritmo comparado com os obtidos pelo *Scale Invariant Feature Transform* (SIFT), destacando o tempo de execução da análise das imagens, bem como a qualidade e a fidedignidade das semelhanças identificadas.**

***Index Terms*—algoritmo, *image matching*, np-completo, implementação**

I. Introdução

A questão P $\neq$ NPC é extremamente intrigante e complexa para a teoria da ciência da computação. Sendo P uma classe de problemas que podem ser resolvidos em tempo polinomial e NPC, uma classe de problemas considerados intratáveis pela maioria dos teóricos, de acordo com [1]. Portanto, ainda não foi descoberto nenhum algoritmo de tempo polinomial para um problema NP-completo, bem como não foi provado que não pode existir nenhum algoritmo de tempo polinomial para eles. Diante desse cenário, este trabalho aborda o problema do *Image Matching* (IM), inerente ao reconhecimento de imagem e tido como um problema difícil sob a perspectiva do algoritmo, ou seja, NP-completo, como mostrou [2].

Assim, pretende-se trazer uma nova forma de implementação e resolução para o problema IM, de maneira que a mencionada implementação não necessariamente seja melhor ou mais rápida do que as propostas existentes atualmente, mas que demonstre um novo modo de pensar na resolução para o problema. Na literatura, encontram-se diversas metodologias e técnicas para comparação de imagens.Sendo que neste trabalho, destacam-se: o algoritmo *Scale Invariant Feature Transform* (SIFT), a comparação de histogramas e, por fim, a redução da imagem a um código de *hash*. Por sua vez, a metodologia adotada parte do mapeamento dos *pixels* das imagens e subsequente associação das semelhanças entre elas a partir de uma nova implementação. Na avaliação, os resultados obtidos pelo novo algoritmo serão comparados com os resultados obtidos por meio do algoritmo SIFT, verificando-se o tempo de execução da análise das imagens, bem como a qualidade e a fidedignidade das semelhanças identificadas.

O conteúdo do artigo está organizado em cinco seções. A seção I é a introdução, que destaca o escopo e a estrutura do artigo. A seção II apresenta uma definição do problema IM, sua formalização e complexidade. A seção III é o referencial teórico apresentando trabalhos relacionados. A seção IV apresenta toda a metodologia de implementação e como será realizada sua avaliação.

II. *Image Matching*

O problema do IM ou Correspondência de Imagem tem sido abordado por diversos autores ([3]–[6]), a fim de buscar melhores técnicas para uma solução mais rápida e precisa. O problema consiste em analisar duas imagens **A** e **B**, onde **B** é uma transformação geométrica de **A**. O resultado desta análise deve ser o encontro de características semelhantes entre elas, de forma que, no final, estas correspondências sejam realçadas e apresentadas para facilitar uma determinada investigação [7]. Como [8] explica, IM é um componente chave para diversos processos de análise de imagens e é muito importante para inúmeras aplicações, como navegação, orientação, vigilância automática, fotogrametria e visão robótica. Na medicina, por exemplo, é de grande importância para encontrar relações entre diferenças no posicionamento do paciente, modalidades e aquisição de imagens variadas [9].

A. Formalização

Existe uma variedade de algoritmos para solucionar este problema, a exemplo de [10], que é citado por [2] para ajudar na formalização do IM. Basicamente, o problema deve receber como entrada (instância) duas imagens de tamanho M x M, onde a segunda imagem é uma transformação da primeira. A solução deve retornar um mapeamento destas imagens de forma que sejam destacadas as semelhanças entre elas, ou seja, deve existir uma função tal que: $f : MxM \rightarrow MxM$. As duas imagens possuem valores em cinza (*grayvalues*) que fazem parte de um alfabeto finito $\alpha = \{1, ..., G\}$. Além disso, são definidas duas funções de distância, d_α, que mede a correspondência em *grayvalues* e d_d, medindo a distorção

introduzida pela correspondência. Assim, é possível formalizar um problema de decisão correspondente, que é definido pela seguinte pergunta: "Tendo uma instância de IM e um limite de custo c', existe uma função de mapeamento tal que $c(A, B, f) \leq c'$?", sendo que o custo é definido pela expressão ilustrada na Equação 1 obtida do [2].

$$c(A,B,f) = \sum_{(i,j)\in M\times M} d_g(A_{i,j}, B_{f(i,j))}) + \sum_{(i,j)\in 1,\ldots,M-1)\times M} d_d(f((i,j)+(1.0)) - f((i,j)+(1.0))) + \sum_{(i,j)\in M\times 1,\ldots,M-1)} d_d(f((i,j)+(1.0)) - f((i,j)+(1.0))) \quad (1)$$

B. Complexidade

Como explicado por [2], o problema IM é de otimização e NP-completo. Ele é assim classificado porque pode ser verificado em tempo polinomial, a partir do problema de decisão $c(A, B, f)$ citado anteriormente, e pode ser construída uma redução do problema 3-SAT (problema de satisfatibilidade booleana) para o problema em questão. Esta redução é construída a partir da definição de uma instância para o 3-SAT. Uma vez definida uma instância, é criado um grafo de dependência, o qual é então representado em um plano 2D (dimensão 2) e aprimorado para representar o comportamento lógico da instância, gerando duas imagens. Por fim, são definidos valores verdadeiros ('*true*') para os *pixels* que possuem uma correspondência e valores falsos ('*false*'), caso contrário, para satisfazer as cláusulas do problema.

III. Revisão de Literatura

Quando se trata de metodologias para comparação de imagens, várias são as técnicas utilizadas no problema. O algoritmo SIFT, desenvolvido por [11], detecta e cria descritores locais relativos a uma determinada imagem, sendo este descritor relativamente invariante a transformações de rotação, iluminação, escala e baixa variação de perspectiva. De uma maneira geral, este algoritmo é divido em quatro etapas principais [12]:

- Detecção de extremos;
- Localização de pontos-chave;
- Definição de Orientação;
- Descritor dos pontos-chave.

Considerando duas imagens i_1 e i_2 diferentes, que representam uma mesma cena, obtém-se com o SIFT duas matrizes M_1 e M_2, as quais contêm descritores de pontos-chave de i_1 e i_2, respectivamente. Para cada descritor de M_1, calcula-se a distância euclidiana com todos os descritores de M_2, assim comparadas as imagens.

A comparação de histogramas é uma outra maneira de se obter informações das imagens [13]. Um histograma é definido como a quantificação do número de *pixels* para cada valor de intensidade de uma imagem ou uma representação gráfica da distribuição de intensidade [14]. A visão estatística sobre a distribuição dos *pixels* e dos níveis de brilho e contraste oferecida pelo histograma são variáveis úteis para comparação de imagens.

Outra metodologia seria com o uso de *hash*, onde cada imagem é reduzida a um pequeno código de *hash*, identificando recursos destacados no arquivo de imagem original e misturando uma representação compacta desses recursos. Em [15], é apresentada uma técnica, utilizando *hashing*, onde, inicialmente, se processa a imagem reduzindo-a para o tamanho 256 × 256. Após isso, converte-se a imagem para escala de cinza, depois, extrai-se um vetor de características. A partir dele, constrói-se um *hashing* de múltiplas tabelas que é utilizado em comparações com outras imagens.

IV. Metodologia

A proposta será implementar um algoritmo na linguagem Python. Esta linguagem foi escolhida pelo fato de oferecer um bom desempenho, além de diversas ferramentas que ajudam nas implementações de algoritmos para reconhecimento de imagens [16]–[18]. Ela também oferece bibliotecas que resolvem o problema IM, as quais, logicamente, não serão utilizadas na implementação do algoritmo deste projeto, uma vez que, o seu objetivo é criar uma nova implementação. Entretanto, será escolhido um deles para realizar a comparação com o algoritmo proposto.

A princípio, planeja-se realizar um mapeamento dos *pixels* de uma das duas imagens. Este mapeamento irá gerar uma lista que permitirá ter algumas características das imagens, a exemplo do código da cor e localização i,j de cada *pixel* da imagem. A posição dos *pixels* será utilizada para saber a exata localização no plano da imagem e, assim, possibilitar o cálculo das distâncias entre *pixels*. Sabendo-se a cor de cada *pixel*, pode-se determinar possíveis objetos e respectivas cores que os caracterizam. Uma vez mapeada a primeira imagem, a ideia é realizar o mesmo mapeamento na segunda imagem, para, em seguida, associar possíveis semelhanças entre as listas de cada uma. As semelhanças serão tanto um conjunto de *pixels* que possuem o mesmo espectro de cores nas duas imagens, quanto a mesma distância entre grupos de *pixels*, ou ao menos uma distância muito próxima da encontrada na primeira imagem. A implementação visa utilizar os *grayvalues*, pois assim se obtém apenas um valor da cor da imagem (de 0 a 255). Caso fosse utilizada uma imagem colorida, far-se-ia necessária a realização de alguns cálculos para considerar os três valores RGB (vermelho, verde e azul).

O pseudocódigo abaixo (Algoritmo 1) organiza os passos básicos que a implementação seguirá. O algoritmo recebe como entrada duas imagens **A** e **B** de mesmo tamanho MxM. O primeiro passo será obter os *pixels* das imagens **A** e **B** em escalas de cinza (`gvA` e `gvB`). Depois desta etapa, será realizado o mapeamento da imagem **A** (mapA), ou seja, o algoritmo percorrerá todos os *pixels* `(i,j)` em busca de valores menores do que um limite (que poderá também ser de 0 a 255). Este limite será definido para capturar *pixels* mais escuros da imagem, a fim de tentar encontrar objetos que estejam na imagem, desconsiderando então o *background*. Em seguida, é obtido o primeiro *pixel* deste mapa (`ipA`), pois a partir deste,

serão estimadas as localizações dos mesmos *pixels* na imagem **B** de forma deslocada. Antes de iniciar o mapeamento de **B**, é preciso encontrar o seu *pixel* inicial (`ipB`), ou seja, o *pixel* que deve ser o primeiro *pixel* de **B** menor do que o `limit`. Com o `mapA`, `ipA`, `ipB` e `limit`, pode-se calcular o mapa de **B** (`mapB`), o qual será obtido realizando alguns cálculos com o `mapA` a partir do `ipA`, a fim de estimar as possíveis posições dos mesmos *pixels* na imagem **B**, pois, como espera-se que **B** seja uma transformação da imagem **A**, deve conter os mesmos *pixels*, porém transladados para outra posição. Existe também a possibilidade de a imagem **B** além de transladada, pode ter pequenas diferenças nos valores dos *pixels*, por isto é utilizado um valor limite para que os cálculos considerem os *pixels* em que a diferença dos seus valores devem ser menor do que este limite, ou seja: $|gvA(i,j) - gvB(i,j)| < limit$. Depois de obtido o `mapB`, serão realizadas as marcações na imagem das correlações que foram encontradas, finalizando assim o algoritmo.

Algoritmo 1. Algoritmo em pseudocódigo

```
lr2Matching(A,B):
  gvA <- grayvalues(A)
  gvB <- grayvalues(B)
  mapA <- selectPixels(gvA, limit)
  ipA <- mapA[0]
  ipB <- getInitialPixel(gvB, limit)
  mapB <- selectPixelsBy(mapA,ipA,ipB,limit)
  highlightPixels(A,B,mapA,mapB,gvA,gvB,limit)
```

A avaliação da implementação será uma comparação com o mencionado método SIFT. Serão calculados o tempo em que cada algoritmo gastou para analisar as imagens, a qualidade e a fidedignidade das semelhanças encontradas. A verificação da qualidade será baseada na visualização dos resultados apresentados, após gerar a imagem final com as marcações das correlações encontradas. A implementação do algoritmo relacionado (SIFT) será também na linguagem Python e será executado na mesma máquina e com as mesmas imagens de entrada, a fim de construir um ambiente justo para realizar a comparação.

REFERÊNCIAS

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to algorithms*. MIT press, 2009.

[2] D. Keysers and W. Unger, "Elastic image matching is np-complete," *Pattern Recognition Letters*, vol. 24, no. 1-3, pp. 445–453, 2003.

[3] Y. Amit, "A nonlinear variational problem for image matching," *SIAM Journal on Scientific Computing*, vol. 15, no. 1, pp. 207–224, 1994.

[4] P. Dupuis, U. Grenander, and M. I. Miller, "Variational problems on flows of diffeomorphisms for image matching," *Quarterly of applied mathematics*, pp. 587–600, 1998.

[5] J. Ma, X. Jiang, A. Fan, J. Jiang, and J. Yan, "Image matching from handcrafted to deep features: A survey," *International Journal of Computer Vision*, pp. 1–57, 2020.

[6] R. A. Paringer, Y. Donon, and A. V. Kupriyanov, "Modification of blurred image matching method," *Computer Optics*, vol. 44, no. 3, pp. 441–445, 2020.

[7] G. Hermosillo, C. Chefd'Hotel, and O. Faugeras, "Variational methods for multimodal image matching," *International Journal of Computer Vision*, vol. 50, no. 3, pp. 329–343, 2002.

[8] A. Gruen, "Adaptive least squares correlation: a powerful image matching technique," *South African Journal of Photogrammetry, Remote Sensing and Cartography*, vol. 14, no. 3, pp. 175–187, 1985.

[9] P. A. Van den Elsen, E.-J. Pol, and M. A. Viergever, "Medical image matching-a review with classification," *IEEE Engineering in Medicine and Biology Magazine*, vol. 12, no. 1, pp. 26–39, 1993.

[10] S. Uchida and H. Sakoe, "A monotonic and continuous two-dimensional warping based on dynamic programming," in *Proceedings. Fourteenth International Conference on Pattern Recognition (Cat. No. 98EX170)*, vol. 1. IEEE, 1998, pp. 521–524.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] F. A. W. Belo, "Desenvolvimento de algoritmos de exploração e mapeamento visual para robôs móveis de baixo custo," *Rio de Janeiro*, 2006.

[13] B. Huet and E. R. Hancock, "Structural indexing of infra-red images using statistical histogram comparison," in *Proceedings IWISP'96*. Elsevier, 1996, pp. 653–656.

[14] R. C. Gonzales and R. E. Woods, "Digital image processing," 2002.

[15] S.-L. Hsieh, C.-C. Chen, and C.-R. Chen, "A novel approach to detecting duplicate images using multiple hash tables," *Multimedia Tools and Applications*, vol. 74, no. 13, pp. 4947–4964, 2015.

[16] S. Kapur, *Computer Vision with Python 3*. Packt Publishing Ltd, 2017.

[17] J. E. Solem, *Programming Computer Vision with Python: Tools and algorithms for analyzing images*. "O'Reilly Media, Inc.", 2012.

[18] S. Dey, *Hands-On Image Processing with Python: Expert techniques for advanced image analysis and effective interpretation of image data*. Packt Publishing Ltd, 2018.