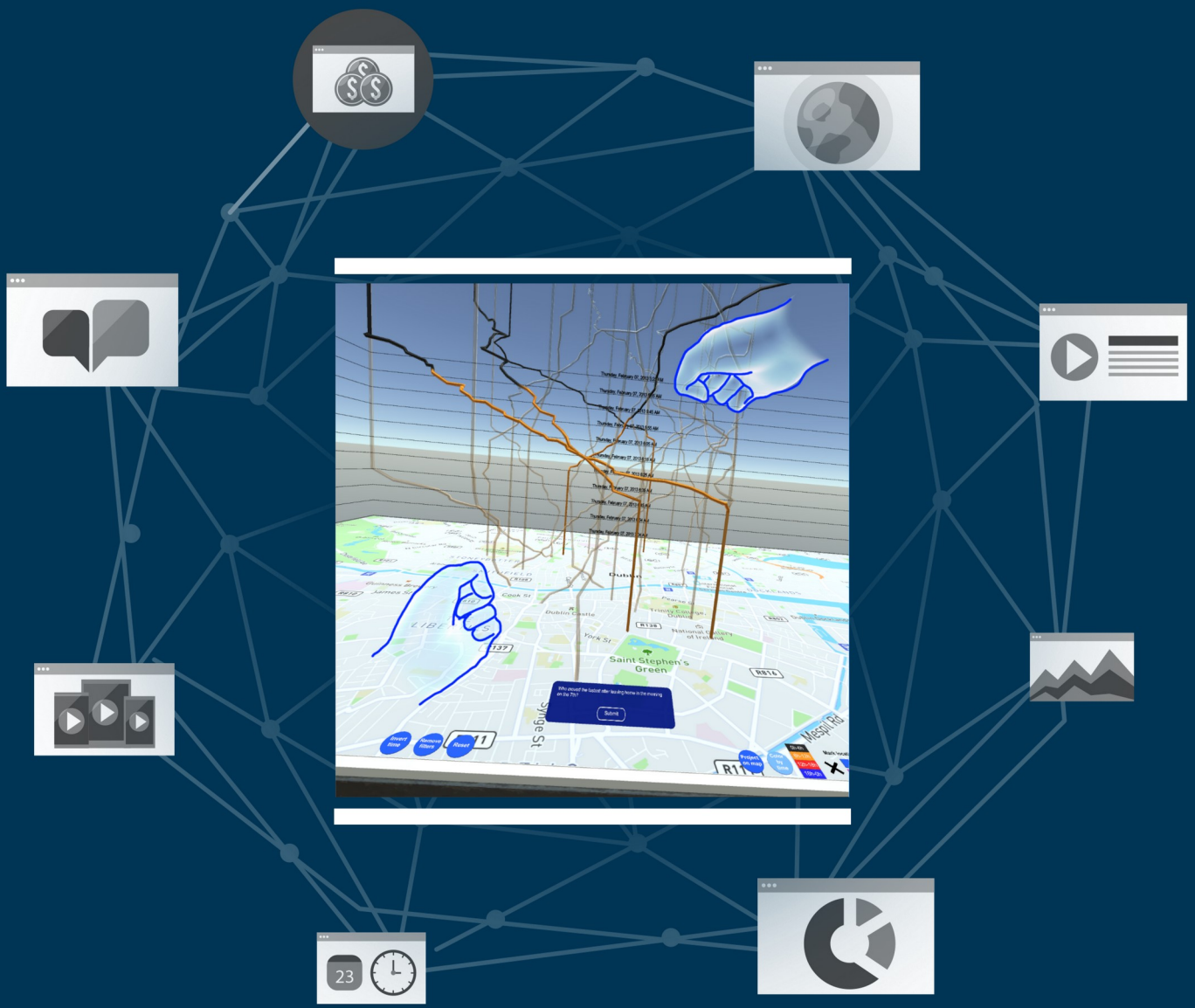
Universidade Federal da Paraíba | Departamento de Informática
Volume 4 Número 1 - Jun//2020

Revista Comunicações em Informática



ISSN: 2595-0622

http://periodicos.ufpb.br/index.php/cei

Editorial

Após 3 anos de atividades, a revista Comunicações em Informática inicia seu processo de consolidação como veículo de rápida divulgação de trabalhos científicos da comunidade brasileira, com destaque à produção discente. Ainda há um longo caminho a percorrer, mas certamente a revista segue na direção certa.

A migração do portal de periódicos da Universidade Federal da Paraíba para uma nova versão do *Open Journal System* propiciou facilidades para autores, revisores e editores, dinamizando o processo editorial. Com acesso às estatísticas de acesso aos trabalhos e maior facilidade da configuração das páginas em 3 línguas (português, inglês e espanhol), têm sido realizados esforços para maior divulgação da revista e expansão do seu corpo de revisores.

No ano de 2019 iniciou-se a realização de chamadas de artigos para seções temáticas. Sendo assim, o número anterior contou com três artigos produzidos a partir de extensões de trabalhos do Workshop de Teses e Dissertações do 20º Symposium on Virtual Reality (SVR 2018) ocorrido no Rio de Janeiro em 2018, com editoria minha e da professora Fátima Nunes da Universidade de São Paulo. Um último artigo desta seção temática aparece na presente edição, totalizando quatro trabalhos aceitos na chamada. No início deste ano de 2020 foi lançada uma nova chamada para seção temática, com foco na Educação em Computação, cuja editoria vem sendo realizada pelas professoras Thaise Costa e Pasqueline Scaico da Universidade Federal da Paraíba, Campus IV.

O número atual possui cinco trabalhos aceitos, assim distribuídos: um trabalho de submissão regular, um trabalho da chamada temática do Workshop de Teses e Dissertações do SVR 2018, e três trabalhos da chamada temática de Educação em Computação. O trabalho de submissão regular, intitulado “Classificação de Singularidades em Imagens de Impressão Digital Baseada em Redes Neurais Convolucionais” aborda as potencialidades do uso de redes neurais convolucionais na classificação de singularidades em biometria, oferecendo resultados iniciais do uso desta abordagem em um modelo proposto. Os demais trabalhos estão apresentados de acordo com a seção temática constante neste editorial.

Gostaria de agradecer a todos que submeteram seus artigos à apreciação desta revista. Embora se espere uma decisão rápida quanto aos trabalhos (aceitação ou rejeição), nem sempre isso é possível. Neste sentido é importante lembrar que o processo de revisão depende não apenas dos editores e revisores, mas também das respostas dos autores, e que o somatório destas ações pode acarretar processos mais ou menos longos de avaliação dos trabalhos.

Agradeço também a todos os revisores e editores associados que vêm colaborando com suas valiosas avaliações e ideias deste o início da revista. Particularmente, agradeço às editoras das seções temáticas, que tanto tem colaborado na proposição das seções e no processo editorial. Este trabalho altruístico e voltado ao crescimento da ciência tem imenso valor.

Convido você, leitor, a navegar pelos artigos e seus diversos temas, esperando que estes possam lhe despertar novas ideias e reflexões no vasto universo da Computação e suas interfaces.

Liliane S. Machado
editora-chefe

Seção Temática: Teses e Dissertações em Realidade Virtual e Aumentada do SVR2018

A pesquisa em realidade virtual e aumentada tem crescido significativamente no Brasil [1]. Desde o ano de 1998 o país passou a realizar um evento nacional sobre o assunto, promovido pela Sociedade Brasileira de Computação, atualmente chamado de Symposium on Virtual and Augmented Reality. O evento ocorre anualmente e dentre suas atividades possui o *workshop* de teses e dissertações, voltado à apresentação e discussão de pesquisas desenvolvidas nesta área no âmbito de programas de pós-graduação.

Nesta seção temática foram aceitos alguns dos trabalhos apresentados no *workshop* de teses e dissertações, cujos resultados prévios já permitem discussão capaz de produzir contribuição científica. Os trabalhos desta seção temática oferecem apenas uma ideia da amplitude de temas relacionados à área de realidade virtual e aumentada que vem sendo pesquisados, bem como as diferentes aplicações destas pesquisas. Na edição atual, o artigo intitulado “Projetando Abordagens Analíticas Imersivas para a Exploração de Dados Espaço-Temporais” apresenta resultados da visualização imersiva de dados com interação tangível, a partir de técnicas de realidade virtual, como uma proposta para melhorar a exploração da dados espaço-temporais. Na edição anterior, de dezembro de 2019, foram apresentados outros três trabalhos, tendo como foco os métodos de testes de software específicos para sistemas de realidade virtual, apresentado em “Fault-based testing approach for VR applications”; o uso da realidade virtual para treinamento militar por meio do mapeamento de terreno, proposto no artigo “Mapeamento das características do terreno em ambiente virtual como ferramenta de apoio

ao ensino militar"; e uma proposta de uso de métodos de aprendizado de máquina para melhorar o processo de visualização de dados médicos por realidade virtual, apresentada no artigo "Visualização 3D interativa de dados médicos temporais baseada em modelo de atenção visual".

Nossos sinceros agradecimentos aos revisores destes artigos, que colaboraram com o enriquecimento dos textos por meio de suas sugestões e construtivas críticas. Agradecemos também a todos os autores, desejando-lhes sucesso na continuidade das suas pesquisas.

Fátima L. Nunes
Liliane S. Machado
editoras da seção temática

Seção Temática: Educação em Computação

Este número da revista Comunicações em Informática traz os primeiros artigos da seção temática "Educação em Computação", que envolve um desafio importante: a pesquisa acerca de melhores formas de ensinar atuais alunos no ensino superior, bem como aqueles que estão na educação básica e, em breve, chegarão nas universidades.

Linhas de estudo e pesquisa que investigam ferramentas, métodos e técnicas que auxiliem o processo de ensino são desafiadoras porque, quanto mais buscamos explorar, mais descobrimos o quanto temos a aprender. Assim, ao buscar ser um meio difusor de trabalhos nesta linha, esta seção temática pretende incentivar estudos e pesquisas na Educação em Computação.

A computação está incorporada em todas as esferas da vida social. O conhecimento acerca da área e as habilidades e competências desenvolvidas por meio da aprendizagem em computação estão se tornando uma necessidade no ambiente social de hoje. Ela emerge como um tipo de alfabetização, descrita como uma base indispensável sobre a qual o conhecimento de outras disciplinas acadêmicas pode ser construído [2].

No entanto, existem inúmeras perguntas sobre métodos, estratégias, técnicas e ferramentas que auxiliem no processo de ensino e aprendizagem de Computação. As pesquisas nesse campo contribuem nesta busca constante pela qualidade desse processo educacional.

Neste sentido, os artigos desta seção apresentam a primeira parte de trabalhos com o recorte temático acerca de experiências no ensino de alguns campos na Computação. A edição apresenta dois artigos com enfoque no ensino de programação e outro alinhado a aspectos de projetos de sistemas computacionais. No artigo "Ensino de Programação utilizando Computação Física: uma Revisão Sistemática da Literatura", os autores apresentam discussões importantes acerca das motivações para o uso de Computação Física no ensino de programação e sobre as principais plataformas e ferramentas de computação física utilizadas nas práticas de ensino nos últimos dez anos. Já o artigo "O uso de Ferramentas para o Ensino de Computação: um foco no ensino de programação" apresenta ferramentas e plataformas para apoiar o ensino introdutório de programação com base em um levantamento de literatura. O artigo "Aprendizado Prático de Subsistemas de Entrada/Saída em Projetos de Sistemas Computacionais com Suporte do Simulador CompSim", por sua vez, aborda uma experiência de uso do simulador CompSim para o apoio ao aprendizado de interações do sistema computacional com seus periféricos, no contexto de projetos de sistemas computacionais.

A seleção destes trabalhos tem contado com a colaboração de revisores que, com suas sugestões, comentários e análises, garantem a qualidade dos manuscritos da revista. Agradecemos estes revisores pelo empenho e horas de dedicação voluntários.

Com isso, esperamos contribuir para a divulgação dos estudos abordados nos artigos e instigado o interesse para outras pesquisas no campo da Educação em Computação. Uma boa leitura a todos!

Thaíse K. L. Costa
Pasqueline D. Scaico
editoras da seção temática

Bibliografia

[1] Detroz, J.; Jasinski, M.; Bosse, R.; Berlim, T.; Hounsell, M. (2014) Virtual Reality Evolution in Brazil: A Survey over the Papers in the "Symposium on Virtual and Augmented Reality". Proc. 2014 XVI Symposium on Virtual and Augmented Reality (SVR). DOI: 10.1109/SVR.2014.39

[2] Guzdial, M. & DiSalvo, B. (2013) Computing education: Beyond the classroom. Computer, 46(9): 30-31. DOI: 10.1109/MC.2013.306

Classificação de Singularidades em Imagens de Impressão Digital Baseada em Redes Neurais Convolucionais

Paulo Ricardo P. da Silva[1], Leonardo Vidal Batista[1], Arnaldo Gualberto Silva[2], João Janduy Brasileiro[2], Diogo Ventura Dantas[1]

[1]Centro de Informática - Universidade Federal da Paraíba
[2]Centro de Engenharia Elétrica e Informática - Universidade Federal de Campina Grande
pauloricardo@ppgi.ci.ufpb.br, leonardo@ci.ufpb.br, {arnaldo.g12, jjanduy, dioogoven}@gmail.com

Resumo: A Biometria oferece um mecanismo de autenticação confiável utilizando traços (físicos ou comportamentais) que permitam identificar usuários baseados em suas características naturais. O processo de comparação de impressão digital utiliza como atributo discriminante informações locais como minúcias. Porém, devido a problemas como ruído na captura ou desgaste nas impressões digitais, este atributo nem sempre é suficiente para a realização desta tarefa. Portanto, singularidades do tipo laço e delta podem contribuir nesta etapa para reduzir a taxa de erro. Este trabalho propõe um método para classificar singularidades em imagens de impressão digital, que se baseia em redes neurais convolucionais. Para avaliar a efetividade do algoritmo proposto foi utilizada a base de dados FVC2006-500 sobre a qual o modelo alcançou acurácia de 98%.

Palavras-chave: *Impressão digital; Singularidades; Redes Neurais Convolucionais*.

1. Introdução

Desde 1893, o Departamento de Segurança do Reino Unido assume que dois indivíduos não possuem a mesma impressão digital. Pouco depois da descoberta de Alphonse Bertillon, que afirmava que uma pessoa poderia ser identificada a partir de um conjunto de atributos antropomórficos, muitos dos principais departamentos de aplicação da lei perceberam o potencial das impressões digitais na identificação de criminosos reincidentes que usavam outros nomes para, ao ser recapturado, escaparem de penalidades mais severas, aplicadas a reincidentes [1]. As agências de segurança investiram em um estudo rigoroso das impressões digitais, desenvolvendo métodos científicos para a comparação visual das digitais e instituindo programas para treinamentos de especialistas na área [2].

A monotonicidade e as cargas de trabalho cada vez maiores decorrentes do aumento da demanda de serviços de reconhecimento das impressões digitais levaram ao desenvolvimento dos primeiros Sistemas de Identificação Automática de Impressões Digitais (*Automatic Fingerprint Identification Systems - AFIS*), aproximadamente 40 anos após o início do uso de impressões digitais para identificação biométrica [2].

Em geral, os AFIS usam minúcias do tipo terminação e bifurcação como atributo discriminante [3]. Atributos globais como laço e delta podem ser usados para reduzir a taxa de erro e classificar impressões digitais em *arch* ou arco plano, *tented arch* ou arco angular, *left loop* ou presilha externa, *right loop* ou presilha interna, e *whorl* ou verticilo. Essas classes são usadas na indexação das amostras de impressão digital, o que ajuda a acelerar o tempo de resposta desses sistemas.

Redes neurais convolucionais (*Convolutional Neural Network - CNN*) são um tipo específico de redes neurais que processam dados representados na forma de matriz, como séries temporais (1-D), imagens (2-D) e vídeo (3-D). Como o nome sugere, ao contrário das redes neurais que operam por meio de multiplicação de matrizes, redes convolucionais aplicam um tipo específico de operação matemática linear chamada de convolução [4]. Redes convolucionais vem ganhando destaque pelo seu desempenho na solução de problemas de classificação e detecção relacionados à imagens, no campo da biometria tem surgido trabalhos baseado nessa tecnologia para classificar impressão digital [5], detectar singularidades [6], e detectar minúcias [7].

Este trabalho propõe um método para classificar singularidades entre as classes laço, delta e não singularidade em imagens de impressão digital baseado em CNN. Porém ele não é completo porque não detecta os pontos singulares, mas pode ser usado para detectar se combinado com um algoritmo de janela deslizante que passa por toda imagem se deslocando em bloco de tamanho 50x50 a um passo *p*.

2. Fundamentação Teórica

Em geral, os trabalhos descritos na literatura científica baseiam-se na imagem de orientação para calcular a localização e o tipo dos pontos singulares. Os algoritmos de detecção e classificação de singularidades podem ser categorizados como: baseado no *Poincaré index* (*Plindex*), baseado no particionamento da imagem de orientação, baseado em *template*, baseado na curvatura da orientação de regiões de pontos singulares, e, mais recentemente, baseado em CNN.

Um método sofisticado e prático, que se baseia no *Plindex*, um campo vetorial e uma curva que envolvem esse campo, foi proposto por Kawagoe e Tojo [8]. Por meio da imagem de orientação, em uma janela deslizante, é calculado o somatório das diferenças dos ângulos de cada pixel em torno de uma vizinhança para determinar o tipo e a localização dos pontos singulares de uma imagem de impressão digital. Há outros trabalhos que se baseiam no *Plindex* ou usam-no como parte do processo de detecção e classificação de singularidades, como em [9, 10].

Alguns algoritmos agrupam as orientações da imagem de orientação de acordo com suas similaridades, formando conjuntos separados por linhas que determinam suas fronteiras. As interseções entre essas linhas fornecem a localização das singularidades [11, 12, 13]. Puneet e Phalguni [13] propuseram um método que calcula um conjunto de singularidades candidatas por meio do algoritmo de particionamento da imagem de orientação, esse conjunto é validado pela análise do *Plindex*. Para refinar a localização das singularidades e detectar possíveis deltas que não foram descobertos, é realizado um pós-processamento, e assim, apenas singularidades genuínas são detectadas e classificadas.

Nos métodos baseados em *template*, para cada tipo de singularidade, há um filtro (ou *templat*e) que é convolucionado sobre a imagem de impressão digital para extrair singularidades [14, 15]. Considerado o estado da arte [16], Awad e Baba [17] apresentaram um método baseado em 2 filtros complexos que captura as propriedades de simetria de laço e delta, respectivamente, então convoluciona cada filtro com a imagem de orientação e o ponto que obtiver a reposta mais alta do filtro é considerado ponto singular. Os algoritmos fundamentados na curvatura da orientação de regiões de pontos singulares são bons para detectar e classificar pontos singulares, visto que as áreas onde encontram-se tais pontos são marcadas pela grande mudança de orientação [18, 19]. Neste sentido, o trabalho de Qi e Liu [19], considerado estado da arte [16], apresenta um método sensível a ruído e falsas singularidades são extraídas.

O único trabalho encontrado e acessível, baseado em CNN, sobre detecção e classificação de singularidades foi o de Qin [6], que apresentou um método que combina CNN e um modelo de probabilidade. Primeiramente, eles treinam um classificador com sub-imagens ou blocos rotulados em 3 classes: laço, delta e não singularidade. Então, eles usam uma CNN para estimar se o centro de um bloco é uma singularidade ou não.

Há outras formas de extrair singularidades (p. ex. aprendizagem de máquina e modelos matemáticos). Qi e Liu [19] propuseram um método baseado em um modelo polinomial complexo (*Zero-pole model*) e uma janela deslizante para detectar e classificar singularidades. *Zero-pole model* da imagem de orientação é essencialmente um polinômio racional complexo cujos *zero* e *pole* são considerados como pontos singulares laço e delta, respectivamente.

O método proposto neste trabalho, diferente dos métodos citados como estado da arte, não depende da imagem de orientação, que tem elevado custo computacional e é sensível à qualidade da imagem. No entanto, ele apenas classifica as singularidades, não detecta. Como os métodos encontrados na literatura classificam e detectam singularidades ao mesmo tempo, não há nenhuma comparação com outros trabalhos.

3. Metodologia

Este trabalho propõe um método de classificação de singularidades em imagens de impressão digital baseado em rede neural convolucional, que é dividido em 2 etapas:

1. Pré-processamento
   (a) Equalização de histograma;
   (b) Realce baseado em filtros de *Gabor* [20];
   (c) Binarização;
   (d) Extração manual de amostras (imagens 50x50 pixels);

2. Classificação por meio da Rede Neural Convolucional.

Para remover ruídos e melhorar a qualidade das imagens, são aplicadas a equalização de histograma, o realce baseado em filtros de *Gabor* e a binarização - veja a Figura 1. Como o objetivo deste trabalho é apenas classificar, para localizar as singularidades e não singularidades (ou Neg), foram extraídas manualmente subimagens (50x50 pixels) para compor a base de dados usada pelo método proposto.

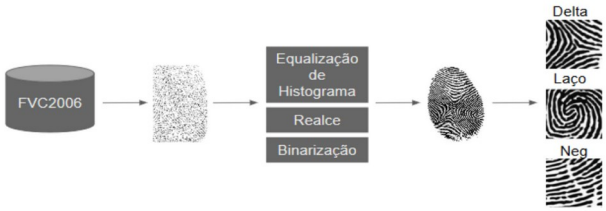


Figura 1. Visão geral do pré-processamento.

Após o pré-processamento, com as imagens separadas em 3 classes (laço, delta e neg), elas são carregadas para um *NumPy Array* [22] de dimensão 2866x50x50x1. Os rótulos das classes são representados seguindo o formato *one hot encoding* e armazenados em um *NumPy Array* de dimensão 2866x3. Todos os valores dos pixels das imagens são convertidos de inteiro para real (*float32*) e normalizados entre [0,1] pela divisão por 255. Para tratar possíveis problemas de superajuste, os dados são divididos em 3 conjuntos: treino, teste e validação. Esse particionamento é feito por meio do método *train test split* da biblioteca *scikit-learn* [23], que faz a divisão de forma aleatória e estratificada, ou seja, mantendo as proporções de cada classe em todas as partições.

Então, a CNN é instanciada e as camadas são adicionadas e configuradas com o inicializador de pesos ou *kernel initializer glorot uniform* e a função de ativação *relu* nas primeiras camadas e na última a função *softmax*. Assim, a CNN é compilada e configurada com a função de perda ou *loss categorical crossentropy*, o otimizador ou *optimizer adam* e com a métrica ou *metrics accuracy* - vide Figura 2 e Tabela 1.

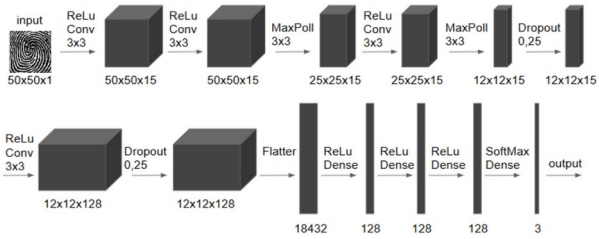


Figura 2. Arquitetura do modelo embasada na LeNet [21].

Tabela 1. Alguns dos hiper-parâmetros usados no modelo.

Nome	Valor
batch_size	128
epochs	50
kernel_initializer	*glorot_uniform*
loss	*categorical_crossentropy*
optimizer	*Adam*
metrics	*accuracy*

No treinamento, a CNN é configurada com os valores dos hiper-parâmetros *batch size* e *epochs*, 128 e 50, respectivamente, e é utilizado o método *imageDataGenerator* do *Keras* [24] para aumentar a quantidade de imagens. Com o objetivo de evitar perda de informação, este método foi configurado apenas para rotacionar (até 45º), girar (*flip*) horizontal e verticalmente.

Para avaliar a capacidade de generalização do modelo, em um dos experimentos, foi aplicada a técnica de validação cruzada *k-fold* com k = 10. Foi utilizado o método *StratifiedKFold* da biblioteca *scikit-learn* que, a cada iteração, aleatoriamente divide os dados em 10 conjuntos, deixando 1 para teste e os outros 9 para treino. Esses 9 conjuntos foram divididos em treino e validação, e então foi realizada a codificação ou *encoding* das classes alvo e usado o aumentador de imagens do *Keras*, como descrito acima.

4. Apresentação e Análise dos Resultados

O primeiro experimento foi usado para encontrar a arquitetura da rede, a configuração dos hiper-parâmetros e o pré-processamento mais adequados a solução do problema. Esta etapa foi realizada de forma empírica, testando diferentes valores de hiper-parâmetros da Tabela 1, camadas da rede e tamanho das imagens extraídas manualmente. Então, após testar algumas configurações diferentes, chegou-se à arquitetura apresentada na Figura 2. O modelo alcançou o índice de acurácia exatidão global de aproximadamente 95% no teste (Tabela 2).

Tabela 2. Matriz de confusão.

		Laço	Delta	Neg
Classe verdadeira	Laço	93	0	1
	Delta	0	38	3
	Neg	1	2	6

Classe Predita

Nesse experimento, o modelo apresentou erro na classificação de 7 imagens, e em todas elas a classe neg estava envolvida. Um fator importante foi que não houve erros entre as classes laço e delta, o que mostra que o modelo consegue distinguir as duas classes principais. A Figura 3 mostra as imagens erradas pelo modelo, e em todas elas não é simples identificar a classe. É possível que esses erros tenham sido causados devido a pouca quantidade de imagens das classes delta e neg, visto que a classe laço, que é a que tem mais amostras, esteve envolvida em apenas 2 erros (ver as Figuras 3b e 3c).

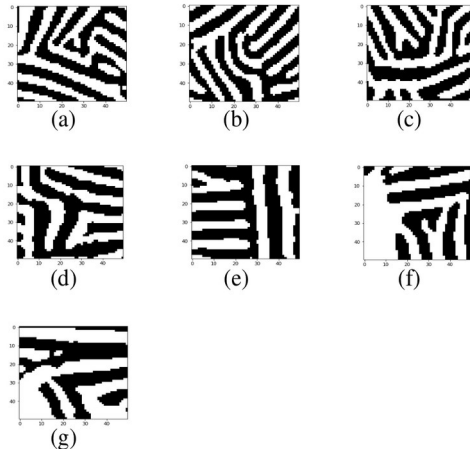


Figura 3. Imagens classificadas erradas pelo modelo no teste do primeiro experimento (gabarito | resultado do modelo): (a) Delta | Neg; (b) Laço | Neg; (c) Neg | Laço; (d) Neg | Delta; (e) Neg | Delta; (f) Delta | Neg; (g) Delta | Neg.

O segundo experimento foi desenvolvido para testar a capacidade de generalização do modelo. Apesar de validação cruzada não ser uma prática comum em modelos baseados em CNN por causa do alto custo computacional, ela foi adotada devido à falta de outros bancos pré-processados e do uso de processamento em GPU. Este experimento durou aproximadamente 4h para concluir as 10 rodadas de validação cruzada, e os resultados obtidos nos testes podem ser vistos na Tabela 3. As matrizes de confusão e as imagens erradas nos testes foram omitidas devido ao espaço que ocupariam neste trabalho.

Tabela 3. Resultados obtidos durante as 10 rodadas de teste de validação cruzada.

	Média	Mediana	Desvio padrão
Acurácia	0.9850	0.9850	0.0024
Perda	0.0469	0.0469	0.0083

O modelo gerou resultados promissores, visto que apresentou o índice de acurácia exatidão global médio de aproximadamente 98%. Esses resultados são preliminares e não garantem que o modelo já está pronto para ser usado como parte de um método de casamento de impressões digitais com desempenho semelhante, pois é necessário testá-lo em outras bases de dados.

5. Conclusão

Apesar de os resultados serem promissores, o modelo precisa ser refinado: é possível reduzir o número de camadas e tornar mais equilibrado o número de amostras das classes. Além disso, é necessário testar o modelo sobre outras bases de dados e aplicar o pré-processamento sobre elas. As pretensões futuras são de desenvolver um modelo baseado em redes neurais convolucionais para fazer tanto a detecção quanto a classificação de singularidades. Mas, para isso ser alcançado, é necessário obter mais bases de imagens para o treinamento do modelo.

Bibliografia

[1] Ross, A. A.; Nandakumar, K.; Jain, A. (2006) Handbook of multibiometrics. Springer Science & Business Media 6.

[2] Maltoni, D. et al. (2009) Handbook of fingerprint recognition. Second edition. Springer Science & Business Media.

[3] Dorizzi, B. et al. (2009) Fingerprint and on-line signature verification competitions at ICB 2009. *Advances in Biometrics – Lecture Notes in Computer Science* 5558: 725–732. DOI: 10.1007/978-3-642-01793-3_74

[4] Goodfellow, I.; Bengio, Y.; Courville, A. (2016) Deep Learning. MIT Press. Online: http://www.deeplearningbook.orgi.

[5] Wu, F.; Zhu, J.; Guo, X. (2020) Fingerprint pattern identification and classification approach based on convolutional neural networks. *Neural Computing and Applications* 32:5725-5734. DOI: 10.1007/s00521-019-04499-w

[6] Qin, J. et al (2017) Multi-scaling detection of singular points based on fully convolutional networks in fingerprint images. *Biometric Recognition – Lecture Notes in Computer Science* 10568: 221–230. DOI: 10.1007/978-3-319-69923-3_24

[7] Nguyen, D.-L.; Cao, K.; Jain, A. K. (2018) Robust minutiae extractor: Integrating deep networks and fingerprint domain knowledge. In: IEEE. 2018 International Conference on Biometrics (ICB), p. 9–16. DOI: 10.1109/ICB2018.2018.00013

[8] Kawagoe, M.; Tojo, A. (1984) Fingerprint pattern classification. *Pattern Recognition* 17(3): 295-303. Elsevier. DOI: 10.1016/0031-3203(84)90079-7

[9] Jin, C.; Kim, H. (2010) Pixel-level singular point detection from multi-scale gaussian filtered orientation field. *Pattern Recognition* 43(11): 3879–3890. Elsevier. DOI: 10.1016/j.patcog.2010.05.023

[10] Zhou, J.; Gu, J.; Zhang, D. (2007) Singular points analysis in fingerprints based on topological structure and orientation field. *Advances in Biometrics – Lecture Notes in Computer Science* 4642: 261–270. DOI: 10.1007/978-3-540-74549-5_28

[11] Ramo, P. et al. (2001) Optimized singular point detection algorithm for fingerprint images. In: Proc. Int. Conf. Image Processing 3: 242–245. DOI: 10.1109/ICIP.2001.958096

[12] Huang, C.-Y.; Liu, L.-m.; Hung, D. D. (2007) Fingerprint analysis and singular point detection. *Pattern Recognition Letters* 28(15): 1937–1945. Elsevier. DOI: 10.1016/j.patrec.2007.04.003.

[13] Gupta, P.; Gupta, P. (2015) A robust singular point detection algorithm. *Applied Soft Computing* 29: 411–423, Elsevier. DOI: 10.1016/j.asoc.2015.01.027.

[14] Nilsson, K.; Bigun, J. (2003) Localization of corresponding points in fingerprints by complex filtering. *Pattern Recognition Letter*s 24(13): 2135–2144. Elsevier. DOI: 10.1016/S0167-8655(03)00083-7.

[15] Jain, A. K. et al. (2000) Filterbank-based fingerprint matching. *IEEE Transactions on Image Processing* 9(5): 846–859. DOI: 10.1109/83.841531.

[16] Zhu, E.; Guo, X.; Yin, J. (2016) Walking to singular points of fingerprints. *Pattern Recognition* 56: 116–128. Elsevier. DOI: 10.1016/j.patcog.2016.02.015.

[17] Awad, A. I.; Baba, K. (2012) Singular point detection for efficient fingerprint classification. *Int. Journal on New Computer Architectures and Their Applications* 2(1): 1–7.

[18] Chen, H. et al. (2011) Fingerprint singular point detection based on multiple-scale orientation entropy. *IEEE Signal Processing Letters* 18(11): 679–682. IEEE. DOI: 10.1109/LSP.2011.2169957.

[19] Qi, J.; Liu, S. (2014) A robust approach for singular point extraction based on complex polynomial model. In: Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops, p. 78–83. DOI: 10.1109/CVPRW.2014.17.

[20] Turroni, F.; Cappelli, R.; Maltoni, D. (2012) Fingerprint enhancement using contextual iterative filtering. In: Proc. 5th IAPR International Conference on Biometrics, p. 152–157. DOI: 10.1109/ICB.2012.6199773.

[21] Lecun, Y. et al. (1998) Gradient-based learning applied to document recognition. Proc. IEEE 86(11): 2278–2324. DOI: 10.1109/ICB.2012.6199773.

[22] Walt, S. v. d.; Colbert, S. C.; Varoquaux, G. (2011) The Numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering* 13(2): 22–30. DOI: 10.1109/MCSE.2011.37.

[23] Pedregosa, F. et al. (2011) Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12: 2825–2830.

[24] Chollet, F. et al. (2015) Keras. https://keras.io.

Projetando Abordagens Analíticas Imersivas para a Exploração de Dados Espaço-Temporais

Jorge Wagner, Luciana Nedel

Universidade Federal do Rio Grande do Sul
{jawfilho, nedel}@inf.ufrgs.br

Resumo: Um dos maiores desafios na computação atualmente é extrair informações relevantes de conjuntos de dados cada vez maiores. Técnicas de visualização de dados permitem aplicar as habilidades humanas de compreensão visual e conhecimento do domínio a este processo. A hipótese deste trabalho é que ambientes imersivos e estereoscópicos de Realidade Virtual (RV), combinados com interação natural, suportarão a exploração de representações de dados espaço-temporais inerentemente tridimensionais melhor do que ambientes *desktop* convencionais. Investigando-se esta hipótese, pretende-se identificar as escolhas de projeto mais eficientes para este tipo de aplicação em termos de interação e colaboração, através de sucessivas avaliações controladas com usuários empregando conjuntos de dados reais. Neste artigo, discute-se como resultados iniciais confirmam o potencial deste tipo de abordagem e quais são os próximos passos nesta pesquisa.

Palavras-chave: *Visualização imersiva; Cubo espaço-temporal; Visualização de trajetórias*.

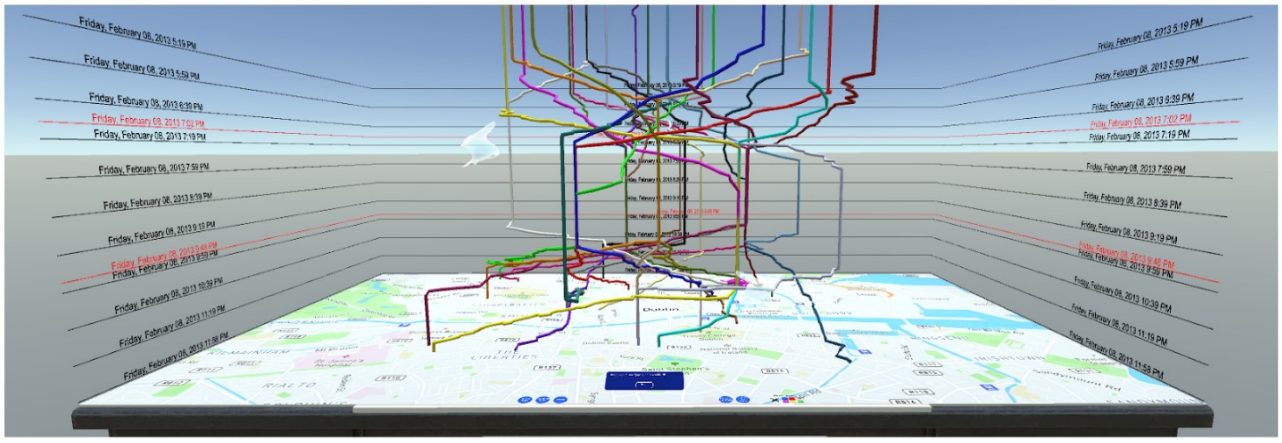


Figura 1. Na primeira abordagem imersiva proposta neste trabalho para explorar dados espaço-temporais, trajetórias de movimentação dispostas tridimensionalmente ao longo do tempo e do espaço foram posicionadas sobre uma mesa virtual, a qual também oferece controles tangíveis.

1. Introdução

Entender como as posições de objetos ou pessoas variam ao longo do tempo e extrair padrões significativos e conclusões a partir destes dados é um tópico de interesse crescente para diversas categorias de usuários, desde pessoas comuns planejando seus deslocamentos até pesquisadores de geografia humana e agentes tomadores de decisões em órgãos públicos interessados em se preparar para possíveis situações futuras ou em analisar as movimentações passadas. Atualmente, dispositivos móveis equipados com GPS, redes de telecomunicação [1] e até mesmo redes sociais [2] são capazes de facilmente coletar grandes e detalhados conjuntos de dados registrando a movimentação de seus usuários ao longo do tempo. Aplicações como cidades inteligentes, planejamento de transportes, estudos comportamentais, controle de epidemias, bem como aquelas que visam aumentar o engajamento dos cidadãos na governança pública podem se beneficiar fortemente da análise de grandes volumes de dados como estes. A alta complexidade e heterogeneidade destes dados invariavelmente requerem a integração da percepção e conhecimento de domínio humano às técnicas automatizadas de análise de dados. Todavia, visualizar atributos e padrões espaço-temporais essenciais nestes conjuntos de dados segue um desafio.

Representações baseadas em mapas bidimensionais focam na natureza espacial dos dados e, mesmo com a ajuda de animações, tornam difícil a observação de características como durações e velocidades variáveis de movimento, locais e durações de paradas e locais de encontro entre diferentes indivíduos (quando estes compartilham a mesma posição no tempo e no espaço). Representações tridimensionais são uma alternativa para abordar de forma mais adequada a natureza temporal destes dados, por exemplo ao empregar o eixo perpendicular ao mapa para representar a componente do tempo, o que resulta em um Cubo Espaço-Temporal (STC – *Space-Time Cube*).

No entanto, da mesma forma que outras representações tridimensionais de dados, o STC possui limitações bem conhecidas em termos de percepção e interação quando usado em ambientes convencionais do

tipo *desktop*. Estas limitações são resultado das dificuldades em estimar distâncias e profundidades a partir de apenas referências visuais monoculares, e da incompatibilidade em termos de controle entre um ambiente 3D e dispositivos de interação 2D como o *mouse*, além dos desafios introduzidos pela oclusão e poluição visual. Isto se torna ainda pior quando se leva em conta que especialistas no domínio dos dados tipicamente não são – e não deveriam precisar ser – treinados em manipulações 3D. Trabalhos anteriores apontaram que especialistas haviam reclamado especificamente da longa curva de aprendizado para a utilização do STC [3].

2. Trabalhos Relacionados

Originalmente proposto por Hägerstrand [4], o STC foi revisitado por Kraak [5] no contexto de um ambiente interativo de geovisualização e, posteriormente, aplicado a diversos domínios, como dados de eventos [6], trajetórias de navios [7], análise de tráfego aéreo [8] e dados de posicionamento de telefones móveis [3].

Alguns esforços iniciais para implementar representações imersivas de dados de movimento, como o STC, foram relatados por Theuns [9] usando um protótipo baseado em capacete de Realidade Virtual (RV), e por Saenz et al. [10] usando um capacete de Realidade Aumentada (RA). Moran et al. [11] também exploraram uma abordagem em RV para a visualização de *posts* geo-posicionados do *Twitter* originados no campus do MIT. Os *tweets* foram distribuídos em uma reprodução virtual do campus e a representação visual específica de cada um foi determinada de acordo com o seu conteúdo. Na aplicação *HoloMaps*, *tweets* geo-posicionados e informações de tráfego foram apresentados em tempo real um modelo de cidade 3D, usando o capacete de RA *HoloLens* [12]. Por fim, no sistema de RA *GeoGate*, Ssin et al. [13] combinaram uma tela 2D do tipo *tabletop* com "hologramas" tridimensionais em RA para visualizar trajetórias no domínio marítimo. Em razão do campo de visão limitado dos dispositivos de RA atuais, o STC foi exibido em tamanho pequeno e a sua posição controlada através da movimentação de um dispositivo tangível, o qual funcionou como um filtro espacial. O GeoGate foi capaz de reduzir erros em tarefas onde os usuários tinham que correlacionar diferentes fontes de dados.

3. Metodologia

O desafio de pesquisa que se propõe para este projeto consiste no planejamento e avaliação de representações imersivas eficientes para dados espaço-temporais. A hipótese é que ambientes de visualização imersiva [14] baseados em capacetes de RV e combinados com interação 3D natural melhor suportarão a exploração visual de tais dados tridimensionais altamente complexos, reduzindo a curva de aprendizado e as dificuldades de interação, como as relatadas anteriormente por Kveladze et al. [3].

A abordagem inicial é baseada na melhoria e expansão da técnica *VirtualDesk* , proposta em estudos anteriores voltados para a visualização imersiva de dados multidimensionais [15], e na sua combinação com metáforas complementares tais como voo virtual e caminhamento real para exploração egocêntrica dos dados. Na *VirtualDesk*, os dados são exibidos em pequena escala sobre uma reprodução virtual da mesa real de trabalho do analista, possibilitando interações incorporadas e tangíveis e oferecendo referências mais fortes de estereopsia e propriocepção. Anteriormente, esta abordagem resultou em benefícios de precisão em relação a uma alternativa *desktop* para tarefas de percepção de distâncias e densidades, ao mesmo tempo em que adicionou pouca ou nenhuma demanda temporal e não causou nem desconforto nem enjoo, até então uma preocupação séria em aplicações imersivas. Tendo em vista a natureza heterogênea dos dados espaço-temporais, múltiplas visualizações coordenadas serão necessárias, podendo ser posicionadas na superfície ou em torno da mesa virtual do analista.

Esta pesquisa segue uma estratégia iterativa de projeto e avaliação, utilizando avaliações controladas com usuários, tanto com participantes especialistas no domínio dos dados quanto com leigos.

4. Resultados Iniciais

A primeira etapa da pesquisa consistiu em validar o potencial de ambientes imersivos. Para isso, implementou-se um primeiro protótipo (ver Figuras 1, 2 e 3) e conduziu-se uma avaliação com usuários.

Em um experimento controlado [16], 20 participantes completaram 7 tarefas de diferentes níveis de dificuldade tanto no protótipo imersivo quanto em um ambiente *desktop* convencional. O primeiro atingiu uma pontuação de usabilidade significativamente mais elevada no questionário SUS [17] (82,3 vs. 62,1) e conquistou a preferência da maioria dos participantes - 19 o consideraram mais engajador, 18 mais intuitivo e 13 mais rápido. Além disso, a incidência de desconforto foi muito baixa (incremento médio de 2,8 pontos no questionário SSQ após o experimento), e a carga mental medida pelo questionário NASA-TLX significativamente reduzida (de 41,6 para 32,4).

Por fim, também foram coletadas recomendações de melhorias e novas funcionalidades a partir de uma colaboração em andamento com pesquisadores da geografia [18]. Estas recomendações incluem novas ferramentas, como planos de corte e filtros mais avançados, bem como o enriquecimento dos dados com uma maior variedade de atributos semânticos.

5. Discussão

As próximas etapas desta pesquisa incluem avaliações mais detalhadas com especialistas no domínio dos dados, e a aplicação da abordagem a conjuntos de dados espaço-temporais reais em diferentes domínios, tais como dados de mobilidade urbana e de saúde pública. Diferentes categorias de dados (e.g., dados de eventos, dados do tipo origem-destino, ou trajetórias de GPS) e maiores volumes de dados resultarão em diferentes requisitos de projeto, que precisarão ser suportados.

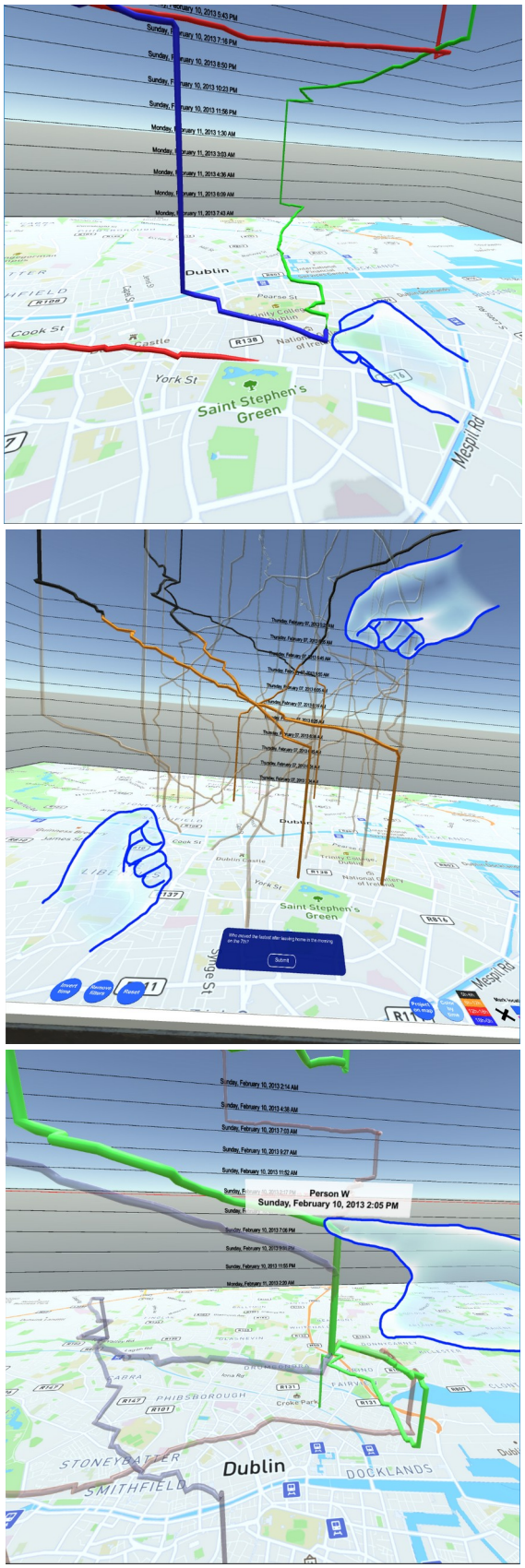


Figura 2. No cubo espaço-temporal imersivo, todas ações são implementadas por meio de gestos intuitivos, como segurar (topo), esticar (centro) e encostar (baixo). Contornos das mãos foram adicionados à figura para maior clareza

Figura 3. Ao reproduzir a mesa real do analista de dados no ambiente virtual, permite-se a interação tangível com comandos dispostos na superfície da mesma, além de agregar uma referência do mundo real.

Levando em conta o processo de trabalho típico de analistas de dados, também serão investigadas diferentes possibilidades de técnicas de interação e, especialmente, colaboração (tanto local quanto remota) e seus efeitos no desempenho analítico geral.

Espera-se que esta pesquisa culmine em um ambiente analítico imersivo completo, que suporte metáforas de exploração complementares e que seja capaz de auxiliar diferentes tipos de usuários, incluindo autoridades municipais, planejadores urbanos e pesquisadores em seus processos de tomada de decisões. Baseando-se no uso de dispositivos de baixo custo, este ambiente também deve ser capaz de engajar usuários regulares na exploração casual de dados [19].

De uma forma mais ampla, essas avaliações também resultarão em novas diretrizes para a construção de aplicações eficientes para visualização imersiva de dados [14], e para o projeto de técnicas apropriadas de interação e colaboração neste contexto.

6. Conclusão

Ambientes imersivos de Realidade Virtual mostram-se úteis para atividades de análise de grandes volumes de dados, possibilitando que analistas interajam de forma natural com dados tridimensionais. Resultados iniciais confirmaram que um Cubo Espaço-Temporal imersivo pode proporcionar maior usabilidade e menor carga mental em comparação a um ambiente *desktop* convencional. A partir de avaliações mais detalhadas com especialistas e novos conjuntos de dados, espera-se definir as abordagens mais adequadas para a visualização e interação com os dados nestes ambientes, de forma que possam auxiliar analistas e tomadores de decisão em ambientes reais.

Agradecimentos

Agradecemos financiamento recebido do Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e de Global Affairs Canada. O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Código de Financiamento 001.

Bibliografia

[1] Calabrese, F.; Ferrari, L.; Blondel, V. D. (2015) Urban sensing using mobile phone network data: A survey of research. *ACM Computing Surveys* 47(3):1–20. DOI: 10.1145/2655691.

[2] Noulas, A.; Scellato, S.; Mascolo, C.; Pontil, M. (2011) An empirical study of geographic user activity patterns in foursquare. Proc. Fifth International AAAI Conference on Weblogs and Social Media.

[3] Kveladze, I.; Kraak, M.-J.; Van Elzakker, C. P. (2015) The space-time cube as part of a geovisual analytics environment to support the understanding of movement data. *International Journal of Geographical Information Science* 29(11):2001–2016. DOI: 10.1080/13658816.2015.1058386.

[4] Hagerstraand, T. (1970) What about people in regional science? *Papers in Regional Science* 24(1):7–24. DOI: 10.1007/BF01936872.

[5] Kraak, M.-J. (2003) The space-time cube revisited from a geovisualization perspective. Proc. 21st International Cartographic Conference. Citeseer, pp. 1988–1996.

[6] Gatalsky, P.; Andrienko, N.; Andrienko, G. (2004) Interactive analysis of event data using space-time cube. In: Proc. International Conference on Information Visualisation (IV). IEEE, pp. 145–152 .

[7] Andrienko, N.; Andrienko, G. (2013) Visual analytics of movement: An overview of methods, tools and procedures. *Information Visualization*, 12(1):3–24. DOI: 10.1177/1473871612457601.

[8] Buschmann, S.; Trapp, M.; Dollner, J. (2016) Animated visualization of spatial–temporal trajectory data for air-traffic analysis. *The Visual Computer* 32(3):371–381. DOI: 10.1007/s00371-015-1185-9.

[9] Theuns, J. (2017) Visualising origin-destination data with virtual reality: Functional prototypes and a framework for continued VR research at the itc faculty. B.S. Thesis, University of Twente.

[10] Saenz, M.; Baigelenov, A.; Hung, Y.-H.; Parsons, P. (2017) Reexamining the cognitive utility of 3D visualizations using augmented reality holograms. Proc. IEEE VIS Workshop on Immersive Analytics. IEEE

[11] Moran, A.; Gadepally, V.; Hubbell, M.; Kepner, J. (2015) Improving big data visual analytics with interactive virtual reality. Proc. IEEE High Performance Extreme Computing Conference. p. 1-6. DOI: 10.1109/HPEC.2015.7322473.

[12] Hills-Duty, R. (2017) Taqtile are creating new holomaps for the hololens. Online: https://www.vrfocus.com/2017/07/taqtile-are-creating-new-holomapsfor-the-hololens. Acesso em 22/10/17.

[13] Ssin, S. Y.; Walsh, J. A.; Smith, R. T.; Cunningham, A.; Thomas, B. H. (2019) Geogate: Correlating geo-temporal datasets using an augmented reality space-time cube and tangible interactions. Proc. 26th IEEE Conference on Virtual Reality and 3D User Interfaces. p. 210-219. DOI: 10.1109/VR.2019.8797812.

[14] Chandler, T.; Cordeil, M.; Czauderna, T.; Dwyer, T.; Glowacki, J.; Goncu, C.; Klapperstueck, M.; Klein, K.; Marriott, K.; Schreiber F. et al. (2015) Immersive analytics. Proc. 2015 Big Data Visual Analytics (BDVA). p. 1-8. DOI:10.1109/BDVA.2015.7314296.

[15] Wagner Filho, J. A.; Freitas, C. M.; Nedel, L. (2018) VirtualDesk: A Comfortable and Efficient Immersive Information Visualization Approach. *Computer Graphics Forum* 37(3):415-426. DOI: 10.1111/cgf.13430.

[16] Wagner Filho, J. A.; Stuerzlinger, W.; Nedel, L. (2019) Evaluating an immersive space-time cube geovisualization for intuitive trajectory data exploration. *IEEE Transactions on Visualization and Computer Graphics* 26(1): 514-524. DOI: 10.1109/TVCG.2019.2934415.

[17] Brooke, J. (1996) SUS-A quick and dirty usability scale. Usability Evaluation in Industry, ch. 21.

[18] Wagner Filho, J. A.; Freitas, C. M. D. S.; Nedel, L. (2019) Comfortable immersive analytics with the virtualdesk metaphor. *IEEE Computer Graphics and Applications* 39(3): 41-53. DOI: 10.1109/MCG.2019.2898856.

[19] Pousman, Z.; Stasko, J.T (2007) Casual information visualization: depictions of data in everyday life. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1145–1152. DOI: 10.1109/TVCG.2007.70541.

O Uso de Ferramentas para o Ensino de Computação: Um Foco no Ensino de Programação

Rodrigo Rodrigues de Lima e Milton Miranda Neto

Universidade do Estado de Minas Gerais
Ituiutaba/MG – Brasil
rodrigorl17@gmail.com, milton.neto@uemg.com

Resumo: Ante a iminência de um mundo digital, este artigo traz como objetivo apontar quais são as ferramentas mais usadas atualmente para o ensino introdutório de programação, considerando para isto as ferramentas reutilizáveis. Para o alcance do mesmo foi empregada a metodologia de pesquisa documental e bibliográfica, que consiste, respectivamente, na utilização de artigos jornalísticos e acadêmicos. Basicamente, o que se percebe é que as ferramentas m*ais* utilizadas atualmente são aquelas que concedem ao usuário uma experiência simples e interativa. É importante enfatizar que tanto as ferramentas como os temas abordados neste artigo são passíveis de estudos mais específicos, isto, a fim de se agregar maior conhecimento aos interessados no assunto.

Palavras-chave: *Ensino da computação; Ferramentas de programação*; Framework.

1. Introdução

Ante a uma sociedade cada vez mais inserida no universo *high-tech* (alta tecnologia), poder compreender, utilizar e se adaptar à tecnologia se tornou competência essencial para a evolução humana. Logo, inovar no que diz respeito a utilizar e criar tecnologia passou a ser uma aptidão necessária a todos [1, 2, 3].

Esta digitalização do mundo têm impactando diretamente na forma como as coisas são feitas, isto, em todas as áreas. Na área da educação, por exemplo, esta nova cultura fez com que outros conteúdos e métodos de ensino passassem a ser considerados, como a computação, pois se enxergou que se "a educação permanecesse *off-line"*, seria muito difícil a condução eficiente do indivíduo pelo caminho do aprendizado [1 n. p., 2, 3, 4].

Inserida no Brasil como ciência na década de 90, inicialmente a computação era vista como a inserção de recursos tecnológicos como ferramentas de auxílio à educação. Atualmente ela é vista como um conteúdo educacional essencial para a vida em sociedade, pois ela está presente em tudo, logo, seu ensino tornou-se uma questão de integrar o indivíduo ao mundo atual [1, 5, 7].

Em tempos onde a computação, seja como ferramenta de auxílio ou como ciência, é um elemento universal, a capacidade de produzir tecnologia tornou-se uma competência vigente, destacando assim a importância do seu ensino a todos. Seu ensino proporciona benefícios como a formação do raciocínio lógico e do pensamento computacional, contribuindo assim para a transformação social. No entanto, cabe mencionar que não há uma forma padrão para isto, nos levando a ponderar sobre as diversas formas existentes para o ensino da computação e a focar naquela considera como a mais alinhada a essa necessidade atual de produzir tecnologia, a programação de computadores, programação [7].

A programação se apresenta como agente de integração humana e tecnológica, e seu processo de ensino é capaz de gerar vários benefícios, como a fomentação de habilidades intelectuais e a experiência do trabalho em equipe, por exemplo, sendo a escolha da ferramenta tecnológica a ser utilizada para tal seu maior desafio. Em suma, ocorre que, assim como no ensino da computação, não há um padrão neste cenário, pois ensinar/aprender são atividades singulares e diretamente influenciadas pelos agentes envolvidos [2, 3, 4, 7, 8].

Softwares educacionais, animações e vídeos para a exemplificação e explicação de conceitos abstratos e ferramentas de programação, *frameworks*, são exemplos das formas utilizadas para ensinar programação, sendo este último escolhido como objeto de estudo deste artigo pela sua característica particular de oferecer funções básicas pré-prontas, onde o utilizador apenas adapta seu objetivo ao 'esqueleto' oferecido [2, 3, 5, 8, 9].

Assim, ante todo o exposto, mantendo nossa atenção no ensino da computação através da programação e nosso foco no uso de ferramentas como aliadas para este ensino, especificamente nas reutilizáveis, *frameworks* a nosso ver, o objetivo deste artigo é apontar quais são as ferramentas mais usadas atualmente para o ensino introdutório de programação, isto, a fim de desmistificar a ideia de que programar é algo só para experts em informática e de despertar o interesse do público para o tema, contribuindo com a sociedade de forma geral [8].

2. Metodologia

Neste artigo utilizamos a pesquisa documental e a pesquisa bibliográfica como forma de coleta de dados, onde a primeira se refere ao uso de artigos jornalísticos e de divulgação em sites relacionados ao assunto e a segunda ao uso de artigos publicados e/ou apresentados em congressos, revistas científicas e livros, os artigos científicos [10].

Inicialmente, através da plataforma de pesquisa Google e Google Acadêmico, realizamos uma busca por periódicos ligados a área da computação, usando o termo 'revistas de computação' como chave de busca. Em seguida, através da leitura dos títulos e resumos, realizamos uma busca por artigos relacionados ao nosso tema, o uso de ferramentas no ensino/aprendizagem da computação, e dos tópicos relativos a ele, como, por exemplo, o histórico da área no Brasil. Em seguida realizamos uma busca por publicações relacionadas ao tema utilizando termos como 'ensino de programação, ferramentas para ensinar programação, *frameworks* para

ensino da programação', etc., o que nos levou a pesquisa de tópicos específicos, como das ferramentas mencionadas na seção 4.

É oportuno mencionar, a fim de justificar o porquê da utilização desse método de pesquisa, que a pesquisa bibliográfica e a documental possibilitam um maior e mais diferenciado volume de informações, conferindo, a nosso ver, uma maior relevância ao artigo por reunir em si informações 'antigas' e atuais sobre o tema [11]. Portanto, a escolha dos artigos utilizados foi feita, considerando seu objetivo/ano/local de publicação, de forma a gerar conhecimento e dissipar ideias preconcebidas sobre o ensino e aprendizagem da programação. Assim, na busca pelo objetivo proposto, dividimos esse artigo em: Fundamentação Teórica; Resultados; Discussão; Conclusão e Bibliografia conforme segue.

3. Fundamentação Teórica

Na atualidade, o conhecimento sobre o que são e como utilizar os recursos tecnológicos tornou-se questão de sobrevivência, tornando evidente a necessidade do aprendizado de novos conteúdos. O surgimento dessa nova necessidade 'abriu as portas do mundo' para a inserção da tecnologia em vários âmbitos, na educação, por exemplo, essa inserção pode ser vista na introdução do ensino da computação na área. Conceituado como o ato de desenvolver e/ou aprimorar a capacidade do indivíduo para criar e/ou aperfeiçoar recursos tecnológicos, o ensino da computação é muitas vezes confundido com o ensino da informática, que consiste no ensino de como utilizar os recursos tecnológicos existentes, e não em desenvolver habilidades, como é o caso do primeiro [1, 4, 6, 12].

O ensino da computação como ciência no Brasil teve origem na década de 90, com a implantação do curso de Licenciatura em Computação e Informática no Ensino Superior público pela Universidade Federal de Brasília (UnB). No início dos anos 2000 a disciplina ganhou espaço nas Instituições de Ensino Superior privado (IES), e em 2010, com o surgimento dos Institutos Federais (IF), passou a ter lugar no Ensino Médio brasileiro. No entanto, o ensino da computação nesse nível só aconteceu de fato dentro dos IF, pois ainda hoje são raras as instituições públicas de ensino que o ofereçam [4, 5, 12, 13, 14].

O ensino da computação como disciplina no Brasil é ministrado predominantemente por escolas particulares e especializadas e nos Cursos Superiores e Técnicos, sendo importante ressaltar que, embora utilizando o termo informática, há registros de mobilizações em prol da implantação deste ensino nas instituições públicas de ensino a nível Fundamental e Médio desde a década de 70, embora não fique claro se a intenção era ensinar computação ou informática [4, 5, 12, 13, 14].

Podemos citar a cooperação ocorrida em 1975 entre a Universidade Estadual de Campinas (UNICAMP) e o laboratório do Instituto de Tecnologia de Massachusetts, Media LAB, a fim de "investigar o uso de computadores com a linguagem LOGO na educação de crianças", como exemplo dessa mobilização. Outros exemplos relevantes são: a criação da Secretaria Especial de Informática (SEI), em 1979, uma organização que buscava discutir políticas nacionais para integrar a informática à educação, com ações como a implantação do projeto EDUCOM em 1983, e a realização do I Simpósio Brasileiro de Informática na Educação (SBIE), 1990, evento apontado como o núcleo das deliberações sobre a computação aplicada à educação [14 n. p., 16].

Estas mobilizações são vistas como marco dos esforços para implantar o ensino da computação nas instituições públicas de ensino, luta necessária para a transformação do ser humano, logo do mundo. São estas transformações que estimulam o processo mental no indivíduo, determinando quais as ações necessárias para resolver um problema, é o pensamento/raciocínio computacional, que encontra na programação o 'gênesis' de sua construção [7, 12].

O ensino de programação se apresenta no campo educacional mundial como o caminho para o desenvolvimento cognitivo, fato que por si só mostra sua importância. Presente na base curricular de ensino Fundamental e Médio de países como Estados Unidos, Inglaterra e Portugal, no cenário educacional brasileiro este ensino pode ser encontrado na forma de parcerias de projetos entre Estado e Município e/ou Município e Universidades; disciplinas optativas em colégios privados; Cursos Superiores; Cursos Técnicos nos IF e cursos em escolas particulares especializadas. Essas iniciativas são válidas, mas não são suficientes para garantir a inclusão de todos no mundo *high-tech*, pois são voltadas a grupos específicos, dificultando o acesso de alguns. Daí a importância da inserção da computação, logo do ensino da programação no ensino público, pois, na chamada 'Era do Conhecimento', é a capacidade de raciocínio crítico que garantirá a inclusão do indivíduo nesse novo mundo, o que só pode ser alcançado através do ensino [2, 4, 17, 18, 19, 20].

O pensamento crítico gerado pelo ensino da programação 'cabe em qualquer lugar', pois está presente em diversas áreas do desenvolvimento e aprendizado, sendo possível sua utilização para o ensino de qualquer coisa. Este ensino pode ocorrer de formas distintas e percorrendo diversos caminhos, como os da Matemática (jogos) e os da Tecnologia (*frameworks),* por exemplo, sendo o uso deste último considerado cada vez mais importante. Entre tantos benefícios, a utilização dos caminhos da tecnologia para o ensino da programação facilita o desenvolvimento de programas e aplicações; poupa tempo de trabalho; minimiza erros durante a programação e aumenta a qualidade do que é desenvolvido, e como exemplo deste 'caminho' podemos citar o uso dos *frameworks* [4, 20, 22, 23].

Por definição, os *frameworks* são "um conjunto de técnicas, ferramentas ou conceitos pré-definidos usados para resolver um problema [...] específico. Basicamente, é uma estrutura de trabalho que atua com funções preestabelecidas e que se adaptam à [uma] situação", ou seja, que podem ser modificadas/reutilizadas conforme o necessário, possuindo várias linguagens de programação, funcionalidades, formas de interação,

níveis de complexidade e finalidades de uso. [5, 15 n. p.; 17; 22, 24].

4. Resultados

Baseados no objetivo deste artigo, destacamos aqui cinco ferramentas de programação e *frameworks* utilizados na atualidade e considerados como os melhores para o ensino da mesma conforme o I do Code (centro tecnológico de ensino de programação) e a Hostinger (plataforma de hospedagem de sites de fama internacional). São elas: Blockly, Alice, Swift Playgrounds, Twine e Scratch, ambas consideradas como uma forma *learning by doing* de aprender, o aluno aprende ensinando o computador o que deve ser feito, isto é programar [24, 26, 27].

Tida como adequada ao ensino de programação a crianças e adolescentes, a Blockly, ou Google Blockly, é uma biblioteca visual gratuita para o ensino de lógica de programação inspirada na ferramenta Scratch. Ela é uma "biblioteca [...] que permite a integração de um editor visual de programação baseado em blocos em qualquer página da web ou aplicativo Android", construindo as soluções computacionais de forma simples (comando arrastar e soltar), interativa de acordo com a necessidade do programador. Ela é responsável pela construção de vários projetos educacionais, como o Code.org, BBC micro: bit e o CodeBug, por exemplo, e seus resultados podem ser exportados em linguagem Java Script, Python, PHP, Lua e Dart [24, 26 p. 48].

Podendo ser utilizado no Ensino Superior como recurso introdutório à programação, o Alice é um ambiente de programação gratuito que, assim como a Blockly, tem seu foco no ensino de programação a crianças e adolescentes, porém de forma pouco mais avançada e com uma concepção orientada ao objeto. Estimulando o usuário à interatividade, ele usa o método de blocos para o ensino de programação 3D, possibilitando a criação de animações e narrativas interativas, jogos mais simples e aplicativos, disponibilizando resultados em linguagem Java, C++ Builder, Delphi e VB [24, 27, 21].

Por sua vez, a Swift Playgrounds é vista como uma das ferramentas gratuitas mais avançadas para o ensino da programação. De interface visual, moderna, interativa e divertida, ela é fundamentada em uma metodologia de blocos e possui sua própria linguagem. Sendo adequada ao ensino de conceitos de lógica às crianças e adolescentes de nível iniciante em programação, tem no fato de disponibilizar com excelência conceitos de programação considerados complexos a sua maior vantagem [24, 27].

Já o Twine é uma ferramenta visual gratuita que utiliza textos e imagens para ensinar lógica de programação para adolescentes a partir dos 12 anos, liberando seus resultados na linguagem HTML. Com ele iniciantes em programação podem facilmente criar jogos e histórias interativas e não lineares utilizando basicamente palavras e colchetes [24, 27].

Por fim, a linguagem de programação Scratch utiliza de recursos interativos visuais, sonoros e textuais para a criação de jogos, animações, histórias interativas, aplicativos, simulações, projetos científicos e tutoriais, fazendo uso da metodologia de blocos para ensinar lógica e transformar conceitos abstratos em algo sólido. Recomendada para crianças entre 8 e 16 anos, ela é uma ferramenta gratuita voltada para pessoas com pouca ou nenhuma vivência em programação, sendo mencionada em muitos estudos como uma ferramenta simples, intuitiva e agradável de trabalhar [4, 17, 24, 25, 27].

5. Discussão

Considerando as ferramentas aqui apresentadas se nota que, de forma geral, elas são ferramentas voltadas para pessoas com nível fundamental de alfabetização, ainda que não possuam conhecimentos amplos em programação, focando no ensino dos seus conceitos básicos à lógica de programação propriamente dita.

Usando recursos visuais, sonoros e/ou textuais, essas ferramentas apresentam como ponto comum ter o mesmo objetivo, facilitar o ensino/aprendizado da programação usando recursos e interfaces simples e interativas, o que, a nosso ver, é o que as caracteriza como sendo as melhores ferramentas para o ensino de programação.

Assim, respondendo ao objetivo proposto, podemos inferir que os *frameworks* mais utilizados na atualidade para o ensino da programação de computadores são, basicamente, aqueles que proporcionam interatividade ao aluno e simplifiquem seu aprendizado, como as ferramentas aqui apresentadas, visando, a nosso ver, desmitificar a máxima de que programar é difícil.

6. Conclusão

É importante mencionar que durante o desenvolvimento deste artigo pôde-se notar que é impossível falar da inserção do ensino da computação no Brasil, logo da programação, sem falar da sua implantação no ensino público. O que se entende é que, somente através da inserção dos mesmos no ensino público é que o ensino de computação/programação para todos será possível.

Ao pesquisarmos sobre as ferramentas utilizadas na atualidade para o ensino da programação, não podemos deixar de notar, embora não seja foco deste artigo, que todas as ferramentas aqui apresentadas são gratuitas, fato que nos fez questionar o porquê deste ensino a nível público ser predominantemente oferecido em Instituições de Ensino Superior e Institutos Federais, ponto que, a nosso ver, é passível de investigações mais amplas.

Nota-se também que através do ensino da computação, logo, da programação, é possível se estimular o pensamento crítico do individuo, sendo esta, a nosso ver, sua principal contribuição à sociedade, pois se entende que este é o princípio para que haja significativas transformações sociais.

Por fim, sugere-se uma pesquisa mais aprofundada acerca dos tópicos abordados neste estudo, assim como dos próprios *frameworks* aqui mencionados, pois, sem dúvida, ambos foram abordados de forma introdutória, e um estudo mais aprofundado sobre eles, a nosso ver, agregaria maior conhecimento.

Bibliografia

[1] Construir Notícias. https://www.construirnoticias.com.br/educacao-x-geracao-high-tech/. Acesso em 16/03/20.

[2] G1. https://g1.globo.com/sp/sorocaba-jundiai/especial-publicitario/ctrlplay/noticia/2019/07/26/ensino-da-programacao-e-robotica-contribui-para-o-desenvolvimento-das-criancas-e-adolescentes.ghtml. Acesso em 17/03/20.

[3] G1. http://www.iplace-educacional.com.br/2018/08/14/ensino-de-programacao-ja-e-realidade-em-escolas-brasileiras/. Acesso em 16/03/20.

[4] Ferreira, R.; Duarte, S. (2019) Ensino de Programação: trajetória histórico social e os avanços na cultura digital do Brasil. *Revista Brasileira de Ensino e Tecnologia* 12(1): 386-408. DOI: 10.3895/rbect.v12n1.7532.

[5] Lacerda, M. (2012) Informática como disciplina obrigatória na educação básica. Anais do VI Congresso Internacional de Linguagem e Tecnologia Online. UFMG.

[6] Nascimento, J. (2007) Informática aplicada à educação**.** 1ª edição. Universidade de Brasília.

[7] Matos, E. (2017) O ensino de computação como mecanismo de integração digital. *Revista Computação Brasil* 2(34): 48-53. SBC.

[8] Medium. https://medium.com/@thaisavelino/como-ensinar-programa%C3%A7%C3%A3o-para-crian%C3%A7as-ou-iniciantes-sem-saber-programar-a2291bf8dc47. Acesso em 18/03/20.

[9] Becode. https://becode.com.br/frameworks-front-end-mais-amados-segundo-github/. Acesso em 18/03/20.

[10] Gil, A. (2002) Como Elaborar Projetos de Pesquisa, 4ª edição. Atlas.

[11] Souza, G. et al. (2017) A Gestão de Talentos Atualmente Adotada nas Organizações. *Revista Cientifica Univiçosa* 9(1): 540-545.

[12] Revista Direcional Escolas. https://direcionalescolas.com.br/ensino-da-computacao-curricular/. Acesso em 23/03/20.

[13] Lemos, A.; Freitas, De. (2017) Ensino da Ciência da Computação na Educação Básica: o que alguns países de fala espanhola estão fazendo, e o que podemos fazer no Brasil? Anais do VI Cong. Brasileiro de Informática na Educação, p. 873-882. DOI: 10.5753/cbie.wcbie.2017.873.

[14] Timetoast. https://www.timetoast.com/timelines/25048. Acesso em 23/03/20.

[15] Dinamica.blog. http://dinamicatreinamentos.com/blog/projetos/o-que-sao-frameworks/. Acesso em 19/05/20.

[16] Castro, A.; Gimenes, I.; Castro Filho, J. (2019) Computação na Educação Básica: breve contextualização histórica. *Revista Computação Brasil* 1(41): 30-34. SBC.

[17] Silva Junior, S.; França, S. (2019) Programação para todos: análise comparativa de ferramentas utilizadas no ensino de programação. In: Princípios e Aplicações da Computação no Brasil. 1(1): Atena.

[18] Brasil Escola. https://educador.brasilescola.uol.com.br/noticias/escolas-apostam-aulas-robotica-programacao-para-alunos-ensino/33259.html. Acesso em 24/03/20.

[19] Educação. https://revistaeducacao.com.br/2014/11/04/ensino-de-programacao-e-aposta-de-colegios-em-todo-o-mundo. Acesso em 24/03/20

[20] Computação e Educação. https://blogcomputacaoeeducacao.blogspot.com/2016/05/a-import-ncia-do-ensino-da-programacao.html. Acesso em 24/03/20.

[21] Barros, E. A. R. (2012) Alice: uso do *software* no processo educacional junto aos cursos de engenharia. XL Congresso Brasileiro de Educação em Engenharia, p. 1-9.

[22] Mastertech. https://blog.mastertech.com.br/tecnologia/o-que-e-framework-e-como-isto-influencia-na-programacao/. Acesso em 26/03/20.

[23] Picanço, W. (2016) FDRobô: um framework didático para auxiliar o ensino de linguagem de programação adaptada ao método de aprendizagem cooperativa e competitiva. Dissertação de Mestrado. Programa de Pós-Graduação em Engenharia Elétrica. Universidade Federal do Amazonas.

[24] Idocode. https://idocode.com.br/blog/programacao/melhores-linguagens-de-programacao-para-criancas/Acesso em 26/03/20.

[25] Hornink, G. G. (Org.). (2018) Contribuições da Computação para as Tecnologias Educacionais, 1ª edição. Universidade Federal de Alfenas.

[26] Eli, P.H. (2017) Desenvolvimento de um Ambiente de Apoio ao Ensino de Algoritmos e Programação: usando Blockly. Dissertação de Mestrado. Programa de Pós-Graduação em Tecnologia e Comunicação. Universidade Federal de Santa Catarina.

[27] Hostinger Tutoriais. https://www.hostinger.com.br/tutoriais/linguagens-de-programacao-para-criancas/. Acesso em 28/03/20.

Aprendizado Prático de Subsistemas de Entrada/Saída em Projetos de Sistemas Computacionais com Suporte do Simulador CompSim

Guilherme Álvaro Esmeraldo[1], Edson Lisboa[2], Lucas Cartaxo[1], Cícero Samuel Mendes[1]

[1]Laboratório de Sistemas Embarcados e Distribuídos – LEDS/IFCE
Instituto Federal do Ceará – *Campus* Crato
[2]Laboratório de Estudos Avançados em Eletrônica – LEA/IFS
Instituto Federal de Sergipe – *Campus* Aracaju

guilhermealvaro@ifce.edu.br, edson.lisboa@academico.ifs.edu.br, {lfonteesc, mr.samuelmendes}@gmail.com

Resumo: No estudo de projetos de sistemas computacionais, o tema Subsistemas de Entrada/Saída inclui conceitos complexos e extensos, que não são tratados de forma adequada pelos simuladores computacionais presentes na literatura. Este artigo apresenta uma proposta de uso do simulador computacional CompSim para suportar os processos de ensino-aprendizagem práticos em Subsistemas de Entrada/Saída. Para o desenvolvimento do trabalho aqui proposto, realizou-se um levantamento bibliográfico de aspectos de subsistemas de entrada/saída, e de como os simuladores e sistemas computacionais reais os tratam. Com isso, formulou-se mecanismos que simplificam a criação e conexão de novos periféricos ao sistema computacional virtual do CompSim. A abordagem proposta tem sido empregada em diferentes cursos e os resultados mostram a efetividade do processo de ensino-aprendizagem.

Palavras-chave: *Subsistema de Entrada/Saída; Aprendizado Prático; Simulador CompSim*.

1. Introdução

Em cursos técnicos e superiores nas áreas de computação e engenharia eletrônica, há disciplinas que tratam de aspectos de projetos de sistemas computacionais, tal como a de Arquitetura e Organização de Computadores (AOC)[1][2]. Nessas disciplinas, geralmente, os conteúdos trabalhados incluem desde as estruturas e comportamento dos componentes do computador até sua programação em linguagem de baixo nível.

Nesses conteúdos, um dos aspectos mais importantes se refere à interação do sistema computacional com seus periféricos. Essa interação envolve diversos elementos como interfaces, barramentos, modelos e protocolos de comunicação, módulos de entrada/saída e os próprios periféricos. Esses conceitos são considerados complexos e não triviais, bem como têm sido frequentemente abordados de forma puramente conceitual e com uso de abstrações [3].

Na literatura tem-se optado pela utilização de simuladores para apoio ao aprendizado dos diferentes aspectos de projetos de sistemas computacionais [4]. Os simuladores são ferramentas que buscam representar cenários reais no projeto de sistemas computacionais e têm como principais benefícios a abstração dos diferentes recursos do computador, não necessitarem de laboratórios de hardware e técnicos especializados, bem como permitem configuração e *feedback* rápidos nas simulações.

Vários estudos, como os apresentados em Pena e Freitas [5] e Esmeraldo et al. [7], realizaram comparativos entre simuladores da literatura, onde consideraram determinadas métricas comparativas, como, por exemplo, alto desempenho, suporte de interface gráfica, disponibilidade de documentação, distribuição livre, entre outras. Observa-se todavia que os simuladores: ou não apresentam todos os recursos didáticos necessários para apoio à disciplina [5]; ou, devido à adoção de abstrações, não abordam adequadamente conteúdos importantes [3]; ou, por focarem em determinados componentes do computador, há a necessidade de se utilizar mais de um simulador [4]; ou buscam fidelizar as características dos componentes do computador, tornando-se complexos de configurar e interagir [6].

Nesse sentido, este artigo apresenta um novo recurso do simulador CompSim [11], o qual vem com a proposta de apoiar o aprendizado e exploração, de forma prática, da interação entre o computador e os periféricos. Esse recurso consiste de uma interface de Entrada/Saída (E/S) padronizada que permite conectar, de forma simplificada e automatizada, o sistema computacional simulado à diferentes periféricos, com objetivo de apoiar a realização de experimentos que envolvam desde o projeto de periféricos virtuais e físicos até sistemas computacionais completos reais.

2. Fundamentação Teórica

Dispositivos de E/S são fundamentais em sistemas computacionais, pois possibilitam a interface para a interação entre o sistema e o ambiente exterior. Tais interações podem se configurar entre homem-máquina (IHM), máquina-máquina (IMM) e de comunicação [8]. Portanto, dispositivos periféricos apresentam grandes variações com relação às funcionalidades, tecnologias empregadas, diferentes taxas de transmissão de dados, além de tamanho e formato dos dados na comunicação. Essa heterogeneidade inviabiliza que o processador trate diretamente as particularidades de cada dispositivo periférico. Assim, o projeto de subsistemas de E/S é uma solução integrada de *hardware* e *software* para cada dispositivo específico a ser conectado ao sistema.

De uma forma geral, um Subsistema de E/S é composto por dois submódulos: Módulo de E/S, também conhecido como Controlador, e o próprio periférico, também conhecido como dispositivo externo. A Figura 1 ilustra esses dois elementos e como estão relacionados.

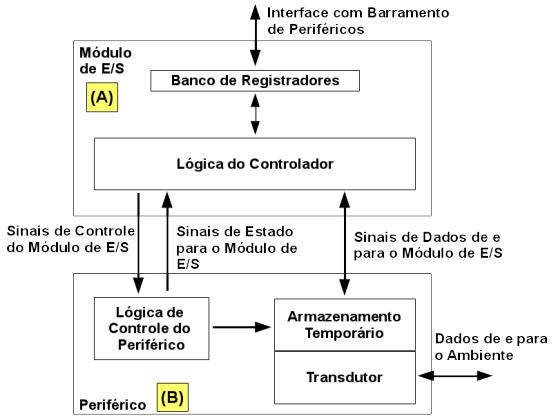


Figura 1. Subsistema de E/S.

O Módulo de E/S, que faz interface com o barramento de periféricos, possui um Banco de Registradores – que armazenam dados, informações de controle, comandos e estados – e a Lógica do Controlador das funções do periférico, como pode ser visto na Figura 1(A). Já o periférico, ilustrado na Figura 1(B) contém: 1) Lógica de Controle do Periférico, que se encarrega de interpretar comandos enviados pelo Módulo de E/S e faz com que a função relativa ao comando seja executada; 2) Transdutor, que é necessário para fazer as devidas conversões entre sinais de natureza diferentes (e.g. conversão entre sinal elétrico proveniente do sistema computacional em uma forma de energia compatível com o ambiente externo, tais como ondas magnéticas ou sonoras); e 3) Armazenamento temporário, que está associado ao Transdutor para que os dados possam ser transferidos entre o periférico e o Módulo de E/S.

Evidentemente, que os Subsistemas de E/S variam com a natureza e complexidade do periférico. Desta forma, percebe-se que o estudo, compreensão e desenvolvimento não são atividades imediatas. Assim, plataformas virtuais têm surgido como um mecanismo para otimizar tanto os processos de ensino-aprendizagem quanto para aumentar o desempenho no desenvolvimento de *software* e *hardware*, em projetos de sistemas computacionais, tal como será apresentado a seguir.

3. O Simulador CompSim

CompSim é um simulador de sistema completo – são tipos de simuladores que incluem todos os componentes do computador –, o qual inclui uma plataforma de *hardware* simulável, conhecida por “Mandacaru”, a qual inclui os seguintes componentes computacionais de simulação: 1) CPU: um processador de 16-bits, com arquitetura mista (RISC e CISC), composta por 16 instruções para realização de operações de transferência de dados, lógicas e aritméticas, desvio de fluxo de programa e de entrada/saída com periféricos. Possui ainda os seguintes submódulos: banco de registradores, contador de instrução, unidade de controle, unidade lógica e aritmética e subsistema de tratamento de interrupções de *software* e *hardware*; 2) Memória cache: é utilizada para otimizar o desempenho dos programas (aproveitando as características de localidade espacial e temporal) e suporta diferentes tipos de configuração, como de políticas de mapeamento, de substituição, entre outras; 3) Memória RAM: é utilizada para armazenamento de dados, instruções e pilha dos programas que serão executados no simulador; 4) Barramentos: o simulador inclui dois barramentos, sendo um de Sistema, que permite a comunicação entre o processador e as memórias Cache e RAM, e um de Periféricos, que permite que o processador se comunique com o Subsistema de Entrada/Saída; 5) Subsistema de Entrada/Saída: inclui uma interface padronizada com módulos de entrada/saída, que possibilita a comunicação do processador com diferentes tipos de periféricos.

O simulador também conta com uma interface gráfica para configurar os componentes virtuais da plataforma Mandacaru, ajustar parâmetros de simulação e suportar a codificação de aplicações em baixo nível (*Assembly*).

A Figura 2 mostra a interface gráfica do CompSim, na qual pode-se visualizar os seguintes componentes gráficos: A) Editor de código; B) Processador; C) Memória cache; D) Memória RAM; e E) Componentes de controle de configuração e execução de simulação.

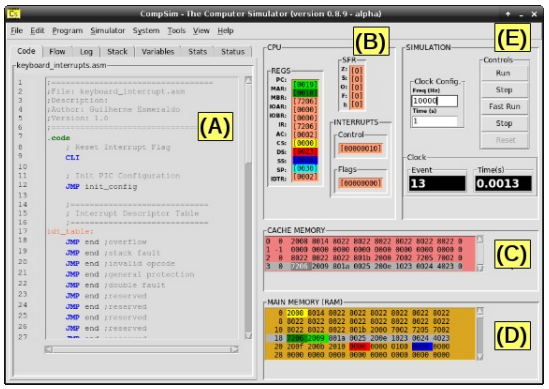


Figura 2. Interface Gráfica do CompSim.

3.1 Subsistema de Entrada/Saída

O Subsistema de E/S, presente na plataforma de *hardware* virtual do CompSim, permite que novos periféricos sejam conectados automaticamente ao barramento de periféricos. Para tanto, cada periférico deve incluir dois arquivos: 1) o primeiro (arquivo com extensão “.csd”) contém uma especificação de sua interface, a qual inclui as seguintes definições: número(s) da(s) porta(s) de entrada/saída; número da interrupção (IRQ), caso o periférico suporte; dados para instanciar o componente de *software* (nome do pacote e da classe de software); e uma curta descrição textual do respectivo periférico; 2) o segundo arquivo inclui o programa que será utilizado para emular o comportamento do periférico (o código-fonte do programa, com extensão “.py”, descrito na linguagem Python).

Desta forma, ao se criar uma plataforma computacional no simulador CompSim, seu Subsistema de E/S buscará pelos arquivos de descrição de interface de componente (“.csd”) e, de forma automatizada,

instanciará os respectivos componentes de *software* (“.py”), que serão conectados ao barramento virtual de periféricos (esse processo é ilustrado na Figura 3). Com isso, os novos periféricos já estarão prontos para interação com restante dos componentes de *hardware* do sistema computacional simulado.

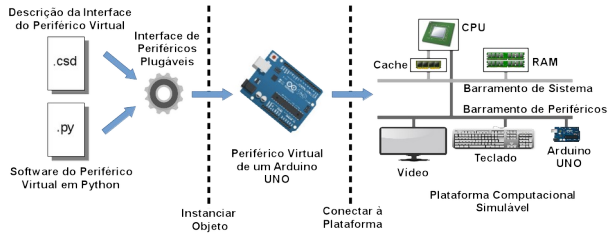


Figura 3. Criação de um periférico plugável e conexão à plataforma virtual.

O CompSim inclui um assistente para dar suporte à criação de novos periféricos, chamado de “*Device Interface Creator*” (ilustrado na Figura 4). Basicamente, ele consiste de um formulário, que deve ser preenchido, pelo projetista, com as informações da interface do novo periférico. Em seguida, o assistente gera automaticamente os respectivos arquivos “.csd” e “.py”, ou, em outras palavras, cria o respectivo Módulo de E/S. Deste ponto em diante, para concluir a criação do novo periférico, o projetista deve apenas focar no desenvolvimento de seu comportamento funcional, ao complementar sua codificação no arquivo-fonte “.py”.

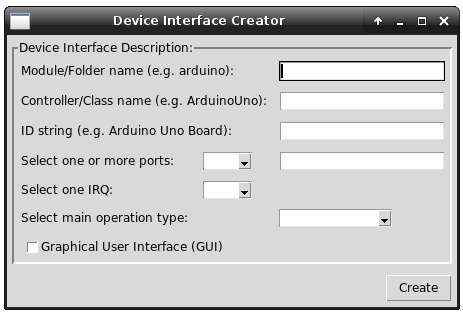


Figura 4. Assistente de criação de interface de periféricos.

Atualmente, o CompSim conta oficialmente com os periféricos Teclado, Vídeo e Arduíno UNO, como podem ser vistos nas Figuras 5(A), (B) e (C), respectivamente. Os periféricos Teclado e Vídeo foram definidos como virtuais (desenvolvidos apenas em *software*), pois emulam os comportamentos dos reais. Cabe ressaltar, que o periférico Teclado possui duas versões, onde diferem pelo mecanismo de interação com o processador (um por *polling* e o outro por interrupções). Já o periférico Arduíno UNO é dividido em dois submódulos, sendo que: o primeiro (*software*) implementa, em *software*, o Módulo de E/S e permite integração do dispositivo ao barramento de periféricos; e o segundo submódulo, que é o periférico propriamente dito, consiste de uma *board* física da plataforma aberta de prototipação Arduíno modelo UNO [9]. Desta forma, é possível criar cenários de projetos de sistemas computacionais onde pode-se realizar a simulação da plataforma computacional virtual interagindo com hardware real. Com o apoio do Arduíno UNO, é possível criar, de forma simplificada, diferentes periféricos físicos para a plataforma virtual, com apoio de diferentes componentes eletrônicos, tais como resistores, capacitores, chaves, *leds*, *displays*, sensores, motores, entre outros.

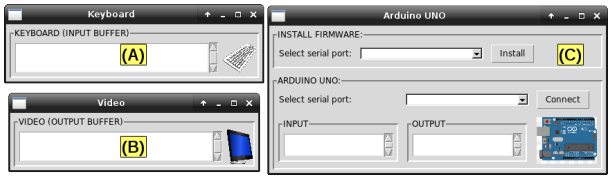


Figura 5. Periféricos do CompSim.

4. Metodologia

Após um levantamento do estado da arte em simuladores computacionais, verificou-se que o tema Subsistemas de E/S ou não é tratado adequadamente – na maioria dos simuladores utiliza-se abstrações que, em certa medida, tratam os conceitos fundamentais tangencialmente – ou não é abordado.

Nesse sentido, para o desenvolvimento do Subsistema de E/S do simulador CompSim, inicialmente, foi necessário realizar um levantamento teórico aprofundado sobre o tema de E/S em projetos de sistemas computacionais. Nesse estudo, avaliou-se a arquitetura de diferentes processadores, multiprocessadores e microcontroladores reais, onde levantou-se suas principais características relacionadas à E/S. Em seguida, buscou-se aplicar abordagens de modularização (desenvolvimento baseado em componentes, padrões de projeto, entre outros) para definir a interface dos componentes de *software* que modelam os periféricos. Com isso, foi possível simplificar o desenvolvimento de novos periféricos e torná-los automaticamente conectáveis ao barramento virtual de periféricos do CompSim.

Para avaliação dos mecanismos definidos para o Subsistema de E/S proposto, foram desenvolvidos exemplos de periféricos (ver seção anterior), os quais são utilizados como base na criação de novos. Uma vez que se pode disponibilizar, aos estudantes, os códigos-fonte dos periféricos apresentados na seção anterior, é possível tomá-los como base e utilizar seus principais conceitos de interface na criação de novos periféricos.

Por fim, a avaliação do suporte ao processo de ensino-aprendizagem é realizada através da aplicação de uma rubrica [10], onde os estudantes avaliam o suporte educacional e a experiência de uso do simulador. Da mesma forma, os professores das disciplinas avaliam se o suporte do simulador favorece diferentes aspectos pedagógicos.

5. Discussão

O Simulador CompSim tem sido utilizado por estudantes de diferentes turmas de um curso de Técnico em Eletrônica e de um bacharelado em Sistemas de Informação. Com o Subsistema de E/S do CompSim, os estudantes lidam de maneira prática com os conceitos de

E/S na criação de diferentes tipos de periféricos virtuais (software) e físicos (*software* e *hardware*), bem como nos aspectos de sua programação, através da criação dos respectivos *device drivers*.

Entre os periféricos criados pelos estudantes, é possível agrupá-los em três categorias básicas, com respectivos exemplos: 1) Periféricos virtuais: sintetizadores de sons com uso do *buzzer* do computador, emuladores de periféricos, tais como *display* de 7 segmentos, *display* LCD 16x2, teclados numéricos, chaves e botões, e um Arduíno UNO virtual; 2) Periféricos físicos analógicos: com o suporte da *board* do Arduíno UNO e componentes eletrônicos, foi possível criar periféricos físicos para interação do usuário com os programas que executam no simulador, tais como saída digital em *leds* e *displays* de 7 segmentos, entrada de dados digitais pelo uso de teclados numéricos e chaves tácteis, e entrada analógica pelo uso de potenciômetros e sensores (luz, temperatura e umidade); 3) Periféricos físicos digitais: com suporte de uma *board* do Arduíno UNO e uma *board* com FPGA (lógica digital reconfigurável), foi possível criar dispositivos digitais, tais como contadores digitais e registradores (*buffer*, deslocamento e carregamento paralelo de dados).

Ao final das disciplinas, um total de 54 estudantes se voluntariou para avaliar o processo de aprendizado de Subsistemas de E/S pelo uso dos recursos do CompSim. Numa escala de 1 a 4, eles destacaram positivamente que: 1) favoreceu o aprendizado de conceitos em E/S pelo desenvolvimento do pensamento de alto nível (méd. 3,8, desv. pad. 0,5); 2) dinamizou o ambiente de laboratório (méd. 3,8, desv. pad. 0,4); 3) adequou-se aos conteúdos curriculares dos respectivos cursos e ao nível escolar (méd. 3,7, desv. pad. 0,5); e 4) a disponibilidade de materiais complementares, tais como componentes eletrônicos e exemplos de códigos-fonte, permitiu auxiliar o aprendizado no uso do simulador e dos conteúdos curriculares (méd. 3,7, desv. pad. 0,4). Do ponto de vista pedagógico, foi possível constatar um aumento significativo na motivação, participação em aulas práticas, produtividade e desempenho geral das turmas, além da redução da evasão nas disciplinas.

6. Conclusões

Este artigo apresenta uma proposta de uso do simulador CompSim para suporte ao aprendizado de conceitos de Subsistemas de E/S em projetos de sistemas computacionais. O simulador proposto inclui uma plataforma virtual de hardware e uma interface de software de E/S que permite conectar novos periféricos (virtuais e físicos), de forma automatizada, à plataforma.

Durante as práticas laboratoriais, os estudantes são estimulados a criar periféricos diversificados, utilizando diferentes recursos (e.g. linguagem de programação Python, Arduíno UNO e componentes eletrônicos), e com isso aplicar os conceitos aprendidos nas aulas teóricas. Os resultados mostraram que a abordagem proposta trouxe resultados efetivos no processo de ensino-aprendizagem em Subsistemas de E/S, além de aumentar a motivação e participação, e reduzir a evasão.

Bibliografia

[1] ACM. Association for Computing Machinery; IEEE Computer Society. (2013) Curriculum Guidelines for Undergraduate Degree Programs in Computer Science.

[2] Zorzo, A. F.; Nunes, D.; Matos, E.; Steinmacher, I.; Leite, J.; Araujo, R. M.; Correia, R.; Martins, S. (2017) Referenciais de formação para os cursos de graduação em computação. In: Sociedade Brasileira de Computação (SBC), 153p.

[3] Larraza-Mendiluze, E.; Garay-Vitoria, N. (2014) Approaches and tools used to teach the computer input/output subsystem: A survey. *IEEE Transactions on Education* 58(1): 1-6. DOI: 10.1109/TE.2014.2310711.

[4] Fernandes, S. R.; Silva, I. S. (2017) Relato de Experiência Interdisciplinar Usando MIPS. *International Journal of Computer Architecture Education* (IJCAE) 6(1): 52-61.

[5] Penna, P. H; Freitas, H. C. (2013) Análise e Avaliação de Simuladores de Sistemas Completos para o Ensino de Arquitetura de Computadores. *International Journal of Computer Architecture Education* (IJCAE) 2(1): 13-16.

[6] Duenha, L.; Azevedo, R. (2016) Utilização dos Simuladores do MPSoCBench para o Ensino e Aprendizagem de Arquitetura de Computadores. In: *International Journal of Computer Architecture Education* (IJCAE) 5(1): 26-31.

[7] Esmeraldo, G. A. R. M.; Mendes, C. S. R. ; Cartaxo, L. F.; Lisboa, E. B. (2019) Apoio ao Aprendizado em Arquitetura e Organização de Computadores: Um Estudo Comparativo entre Simuladores Computacionais. *Revista Tecnologias na Educação* 31: 1-17.

[8] Stallings, W. (2017) Computer Organization and Architecture. 10th Edition. Pearson.

[9] Badamasi, Y. A. (2014). The working principle of an Arduino. In: Proc. 2014 11th International Conference on Electronics, Computer and Computation (ICECCO), p. 1-4. IEEE.

[10] Yuan, M.; Recker, M. (2015) Not all rubrics are equal: A review of rubrics for evaluating the quality of open educational resources. *The International Review of Research in Open and Distributed Learning* 16(5): 16-38.

[11] COMPSIM. CompSim – The Computer Simulator. Disponível em: <http://compsim.crato.ifce.edu.br/>. Acesso em: 20 maio 2020.

Ensino de Programação utilizando Computação Física: uma Revisão Sistemática da Literatura

Humberto A. P. Zanetti, Marcos A. F. Borges

Faculdade de Tecnologia
Universidade de Campinas, Limeira – SP, Brasil
h016304@dac.unicamp.br, marcosborges@ft.unicamp.br

Resumo: Esta pesquisa tem como objetivo discutir sobre a teoria, as aplicações e as principais estratégias didáticas para ensino de Programação utilizando recursos de Computação Física. Pesquisas sobre esse tema são recentes. No meio acadêmico, ainda há questões a serem discutidas sobre o alinhamento entre as atuais pesquisas. Essas questões podem causar má interpretação ou dúvidas ao se escolher ou adotar práticas sobre o tema. Nesta Revisão Sistemática da Literatura, foram avaliados 15 artigos, com o objetivo de apresentar uma visão crítica sobre as pesquisas mais recentes e suas aplicações.

Palavras-chave: *Ensino de programação; Computação física; Revisão da literatura.*

1. Introdução

Atualmente, a tecnologia faz parte da rotina diária de todos e é muito acessível. Jovens estudantes dominam boa parte dos recursos tecnológicos e computacionais, devido à sua vivência com tecnologia desde a infância. É cada vez mais esperado que a escola forneça um ambiente em que a tecnologia esteja presente, não só como uma ferramenta de ensino, mas em dinâmicas nas quais esses jovens deixem de ser meros "consumidores" de tecnologia e se tornem "criadores". Nesse contexto, o papel fundamental da escola e dos educadores é orientar e conduzir práticas que catalisem a motivação e a criação com esses jovens.

A programação de computadores tem potencial para ser uma atividade educativa que pode auxiliar de maneira efetiva a aquisição de habilidades cognitivas fundamentais para diversas outras tarefas, como a atitude crítica-reflexiva e a resolução de problemas [13]. Também auxilia o aluno a explicitar um modelo mental de solução, podendo expressar de maneira quase fiel a sequência de resolução e verificar a causa-e-efeito [14].

Computação Física (CF) é a integração da computação com o mundo físico, através de sensores e atuadores, amparados por sistemas embarcados, comunicando-se via redes de computadores e provendo ambientes automatizados [22]. A CF tem como objetivo conectar o mundo virtual com o real, com a criação de novas interfaces intuitivas entre objetos e seres humanos. Utilizar da CF em práticas de ensino pode oferecer recursos didáticos que ajudem a promover a criatividade e a compreensão da aplicação de conceitos [15].

Atividades envolvendo componentes eletrônicos e robótica, alinhados com o ensino de programação, podem diminuir as dificuldades de alunos iniciantes no processo do aprendizado de programação. Por exemplo, utilizando um robô móvel e uma arena com obstáculos, para demonstrar conceitos como estruturas condicionais (*if-else*) e de repetição (*for* e *while*) [23].

2. Metodologia

Para este trabalho, foi realizada uma Revisão Sistemática da Literatura (RSL), que permitiu identificar e analisar principais aspectos da área de pesquisa que envolve CF e ensino de Programação.

2.1. Questões de Pesquisa

O principal objetivo desta RSL é avaliar estudos realizados no período de 2010 a 2019, que apresentem contribuições ao tema de uso da CF no ensino de programação. A principal questão a ser respondida com este presente trabalho é: "*Quais são as abordagens didáticas aplicadas para o ensino de programação utilizando ferramentas e artefatos de Computação Física?*".Com base na questão principal apresentada, outras 2 questões de pesquisa (QP) mais específicas foram definidas:

QP1: *Quais as motivações para o uso de CF no ensino de programação?*

QP2: *Quais são as principais plataformas e ferramentas de CF utilizadas nas práticas?*

2.2. Métodos Aplicados na Revisão da Literatura

A presente RSL foi realizada seguindo a metodologia PRISMA (*Preferred Reporting Items for Systematic Reviews and Meta-Analyses*) [3]. As bases de dados utilizadas para a busca de artigo foram ACM, Google Scholar, IEEE, Science Direct, Scopus e Springer, em trabalhos publicados entre os anos de 2010 e 2019. A chave (*string*) de pesquisa utilizava termos em português e inglês, como "computação física" ("*physical computing*"), ensino ou aprendizagem ("*teach*", "*teaching*", "*learning*") e programação ("*programming*", "*coding*")

Os critérios de inclusão dos trabalhos selecionados foram definidos a partir da capacidade de atender pelo menos uma das questões de pesquisa e, após isso, seguir os outros critérios de inclusão, mostrados no Quadro 1. Foi feita a aplicação dos critérios de inclusão e exclusão através de uma leitura rápida (*scanning*), dando ênfase nos resumos, seções de resultados e/ou conclusões. Maiores detalhes da metodologia adotada estão disponíveis no site do projeto.[1]

1 Endereço do site: https://tinyurl.com/projetoCFAS

Quadro 1. Critérios de inclusão e exclusão.

Critérios de Inclusão	Critérios de Exclusão
I1. Publicados após o ano de 2010, e até 2019. **I2**. Artigos com práticas pedagógicas e/ou proponha alguma metodologia ou ferramenta didática. **I3**. Abordar práticas de programação e CF	**E1**. Artigos que apresentem apenas revisão da literatura. **E2**. Não abordar ensino de programação ou CF. **E3**. Trabalhos que não contemplem nenhumas das questões de pesquisa

Com isso, foram selecionados 15 artigos que atendem o propósito desta RSL, que foram publicados entre 2013 e 2018, embora tenham sido buscadas pesquisas sobre o tema desde 2010, só surgiram trabalhos relevantes no período indicado. Após essa seleção, foi realizado um fichamento através de um formulário de extração com o propósito de identificar as contribuições de cada artigo.

3. Resultados e Discussão

Esta seção apresenta uma visão geral dos artigos selecionados na Seção 3 e as discussões das questões apresentadas na Seção 2.

3.1. Discussão sobre a QP1

Foi analisado se há relatos ou descrições nos artigos que apontem quais foram as motivações que levaram a adoção da CF nas práticas descritas ou no desenvolvimento de alguma plataforma ou ferramenta. Dentre as motivações descritas, foi possível categorizar 5 áreas distintas, que conseguem trazer um consenso e agrupamentos coerentes entre os trabalhos.

A categorização utilizada nesta RSL foi feita baseando-se nos relatos e descrições mais comuns nos artigos recuperados pelas buscas. Essa categorização tem como objetivo prover uma referência para o mapeamento de futuros pesquisadores interessados em CF e ensino de programação. As áreas definidas foram: “Ambiente dinâmico e/ou motivador” (ADM); “Compreensão de Conceitos ou Paradigmas de Programação” (CCPP); “Engajamento e Criação” (EC); “Resolução de Problemas” (RP); e “Visualização e Identificação de Erros” (VIE). O Quadro 2 traz uma breve descrição de cada categoria e a relação de artigos.

Os artigos que se enquadram na categoria ADM foram a maioria, pois a busca por um ambiente mais dinâmico e motivacional é algo muito buscado em práticas didáticas que envolvam programação. O ambiente tradicional de programação pode ser pouco estimulante, potencializando possíveis problemas e dúvidas que os alunos tenham [8]. Para esses trabalhos, a adoção de recursos da CF pode ser um aliado para esse propósito, devido a uma maior ludicidade das atividades.

A segunda categoria com mais artigos, a CCPP, aborda um problema bastante comum entre os iniciantes em programação, a pouca capacidade de abstração. A aprendizagem de programação demanda habilidades como abstração, capacidade de identificar generalização e pensamento crítico [4]. Meios que ajudem a visualização, de maneira mais concreta, fora do computador, dos resultados de certos conceitos de programação podem ser bastante positivos.

Quadro 2. Descrição das categorias e relação dos artigos.

Id.	Descrição	Artigos
ADM	Relatos que indicavam que a presença de CF tinha o objetivo de criar um ambiente mais dinâmico e motivacional, diferente do tradicional, despertando maior interesse pelos alunos.	[1], [2], [6], [9], [10], [11], [12], [16], [17], [21]
CCPP	Artigos que procuram meios de trazer instrumentos didáticos para ajudar os alunos na abstração e visualização de conceitos abstratos relacionados à programação.	[1], [5], [6], [7], [10], [12], [18], [19]
EC	Participação ativa dos alunos e estímulo à criação de novos artefatos e soluções usando CF, estimulando a cultura *maker*	[5], [20]
RP	Trabalhos que abordam a habilidade de tomada de decisão e resolução de problemas, estimulando a participação do aluno na construção de soluções.	[10], [21]
VIE	Estudo que utilizam a CF para proporcionar um ambiente que possa trazer *feedback* que sejam melhores interpretados e os erros visualizados de maneira mais tangível.	[10], [16]

Há um alinhamento com os artigos da categoria VIE, pois a busca por instrumento que tornem os conceitos de programação mais tangíveis auxilia na interpretação e na percepção de erros que estão ocorrendo, e consequentemente, a busca por soluções.

Os trabalhos da categoria EC dão ênfase na relação de engajamento do aluno junto a prática e ao incentivo de utilizar a tecnologia não apenas como um consumidor, mas sim como um criador.

Os trabalhos que se enquadram na categoria RP tinham como objetivo proporcionar um cenário que explorasse a habilidade de tomada de decisão e resolução de problemas, que são essenciais para a programação.

É importante notar que alguns artigos aparecem em mais de uma categoria, denotando algumas intersecções. Os artigos [1], [6], [10] e [12] aparecem em ADM e CCPP, mostrando que trabalhos buscam ambientes motivadores com o objetivo de trazer maneiras de ajudar na abstração de conceitos complexos. O trabalho A3 aparece nas categorias CCP e EC, por tratar sobre a criação de artefatos a partir elementos da CF e programação, para promover a visualização de conceitos abstratos. Por fim, uma análise possível de interseção de categorias é o artigo [10], presente em 4 categorias distintas (ADM, CCPP, RP e VIE), tratando-se de uma pesquisa que apresenta práticas detalhadas, lúdicas, motivacionais e focadas em trazer conceitos abstratos de maneira mais tangível através de puzzles.

3.2. Discussão sobre a QP2

O objetivo principal desta questão foi fazer um levantamento de quais são as principais plataformas ou ferramentas utilizadas, podendo se identificar, inclusive,

se há novas ferramentas sendo desenvolvidas. Buscou-se identificar o cenário atual e quais tecnologias são mais adotadas nas pesquisas e as novas demandas.

As tecnologias utilizadas ou desenvolvidas nos trabalhos selecionados foram: a plataforma de prototipagem Arduíno, o kit de robótica *Lego® Mindstorms*, a plataforma de tecnologias vestíveis *Lilypad* Arduino e as placas de eletrônica *Makey Makey* e *micro:bit*. Também houve 3 trabalhos que desenvolveram suas próprias tecnologias. A Figura 1 mostra um gráfico com relação total dos artigos e seus identificadores, além dos números de artigos.

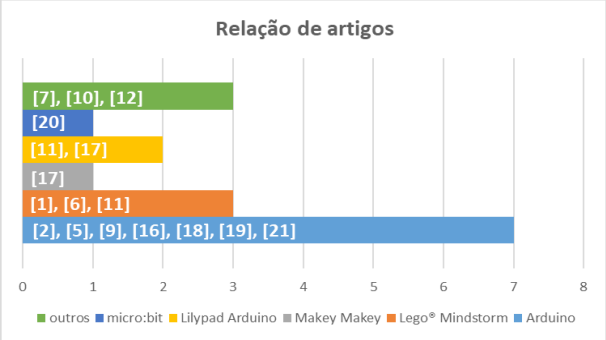


Figura 1. Relação dos artigos por ferramenta/plataforma.

A presença do Arduíno na maioria dos trabalhos era esperada, por se tratar de uma plataforma de CF acessível, tanto por disponibilidade no mercado, quanto devido ao seu caráter de projeto aberto de hardware. Também existe muito material e literatura voltado à plataforma. A versatilidade do Arduíno é outro ponto positivo na sua adoção, pois pode ser incluído em projetos simples e complexos, envolvendo práticas de robótica e Internet das Coisas.

O baixo custo e fácil manutenção também são atrativos para instituições de ensino. Para esse estudo, a plataforma *Lylipad* (baseado em Arduíno) foi separada das demais, devido à sua aplicação específica em tecnologias vestíveis. Os dois estudos que usam *Lilypad* possuem práticas com um foco estritamente em desenvolvimento de projetos objetivando a criação de dispositivos vestíveis (*wearable*) e ambos usam outras tecnologias conjuntamente: um usa também *Makey Makey* e outro *Lego® Mindstorms*.

O kit didático *Lego® Mindstorms* aparece em 3 artigos, sendo utilizado em práticas de robótica e programação, que é justamente o objetivo principal dessa ferramenta. Mesmo com seu alto custo no mercado, seus benefícios são notáveis nos estudos, como a versatilidade de construções de modelos automatizados e a facilidade de programação. As demais plataformas didáticas, *Makey Makey* e *micro:bit*, que foram desenvolvidas para criar pequenos projetos e ter uma interface de programação intuitiva, aparecem em apenas 1 artigo para cada ferramenta. Ambas são ferramentas com recursos limitados e custo relativamente alto, se comparadas com as plataformas citadas anteriormente.

Houve 3 trabalhos que desenvolveram tecnologias próprias para serem utilizadas em práticas com CF: *Talkoo kit* no artigo [7], *Bots & (Main)Frames* no artigo [10] e *i*CaTch* no artigo [12]. A *Talkoo* kit tem como objetivo criar um ambiente de ensino de conceitos de computação, com facilidade de montagem de projeto através de módulos *plug-and-play* e com ambiente de programação visual. O projeto *Bots & (Main)Frames* propõe um ambiente de desafios (*puzzle*) para o ensino de programação, utilizando blocos físicos baseados em tecnologia tangível. Por fim, a plataforma *i*CaTch* apresenta um kit didático usando CF e tecnologia vestíveis, para estimular o ensino de programação e criação de projetos de dispositivos vestíveis.

4. Conclusões

Este estudo teve como principal objetivo fazer o levantamento e análise de pesquisas que abordam recursos da CF no ensino e aprendizagem de programação. Para alcançar esse objetivo, foi realizada uma RSL buscando por trabalhos nas principais bases de trabalhos acadêmicos publicados a partir de 2010.

Foi observada uma maior concentração de artigos de pesquisas que buscam a CF como um recurso didático com propósito de criar um ambiente motivador, que promova maior engajamento por parte do aluno e que apresente elementos que auxilia no aprendizado de conceitos de programação, muitas vezes abstratos. Esses aspectos são recorrentes em diversos trabalhos que objetivam novas estratégias ou recursos para o ensino de programação. Também foi identificado que há poucos trabalhos focados em desenvolvimento de habilidades necessárias ao programador, como resolução de problemas ou identificação e correção de erros.

A plataforma mais utilizada nas pesquisas foi o Arduíno, seguida pelo kit didático *Lego® Mindstorms*, ferramentas essas presentes há mais de uma década no mercado. A plataforma *Lilypad*, baseada em Arduíno, apresenta uma solução para tecnologias vestíveis, podendo trazer soluções criativas e acessíveis. As ferramentas *Makey Makey* e *micro:bit*, mais recentes, também foram citadas.

Bibliografia

[1] Anfurrutia, F. I. et al. (2016) Incorporating educational robots and visual programming environments in introductory programming courses. In 2016 International Symposium on Computers in Education (SIIE) (pp. 1-4). IEEE. DOI: 10.1109/SIIE.2016.7751835.

[2] Arakliotis, S.; Nikolos, D. G.; Kalligeros, E. (2016) LAWRIS: A rule-based arduino programming system for young students. In 2016 5th International Conference on Modern Circuits and Systems Technologies (MOCAST) (pp. 1-4). IEEE. DOI: 10.1109/MOCAST.2016.7495150.

[3] Galvão, T. F.; Pansani, T. D. S. A.; Harrad, D. (2015) Principais itens para relatar Revisões sistemáticas e Meta-análises: A recomendação PRISMA. *Epidemiologia e Serviços de Saúde* 24: 335-342.

[4] Gomes, A.; Mendes, A. J. (2007) Learning to program-difficulties and solutions. In: International Conference on Engineering Education–ICEE, v. 2007.

[5] Jang, Y.; Lee, W.; Kim, J. (2015) Assessing the usefulness of object-based programming education using arduino. *Indian Journal of Science and Technology* 8(S1): 89-96. DOI: 10.17485/ijst/2015/v8iS1/57701.

[6] Jin, K. H.; Haynie, K.; Kearns, G. (2016) Teaching elementary students programming in a physical computing classroom. In: Proc. 17th annual conference on information technology education, p. 85-90. DOI: 10.1145/2978192.2978238.

[7] Katterfeldt, E. S. et al. (2018) Physical computing with plug-and-play toolkits: Key recommendations for collaborative learning implementations. *International Journal of Child-Computer Interaction* 17: 72-82. DOI: 10.1016/j.ijcci.2018.03.002

[8] Lahtinen, E.; Ala-Mutka, K.; Järvinen, H. M. (2005) A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin* 37(3): 14-18. DOI: 10.1145/1067445.1067453.

[9] Martin, C.; Hughes, J.; Richards, J. (2017) Learning Experiences in Programming: The Motivating Effect of a Physical Interface. In: Proc. 9th International Conference on Computer Supported Education (CSEDU) p. 162–172. DOI: 10.5220/0006375801620172.

[10] Melcer, E. F. (2017) Exploring the Effects of Physical Embodiment in a Puzzle-Based Educational Programming Game. In: Proc. 2017 ACM SIGCHI Conference on Creativity and Cognition, p. 532-538) DOI: 10.1145/3059454.3078704.

[11] Merkouris, A.; Chorianopoulos, K.; Kameas, A. (2017) Teaching programming in secondary education through embodied computing platforms: Robotics and wearables. *ACM Transactions on Computing Education* (TOCE) 17(2): 1-22. DOI: 10.1145/3025013.

[12] Ngai, G. et al. (2013) Designing i* CATch: A multipurpose, education-friendly construction kit for physical and wearable computing. *ACM Transactions on Computing Education* (TOCE) 13(2): 1-30. DOI: 10.1145/2483710.2483712.

[13] Papert, S. (1994) A máquina das crianças. Porto Alegre: Artmed.

[14] Pea, R. D.; Kurland, D. M. (1984) On the cognitive effects of learning computer programming. *New Ideas in Psychology* 2(2): 137-168. DOI: 10.1016/0732-118X(84)90018-7.

[15] Przybylla, M.; Henning, F.; Schreiber, C.; Romeike, R. (2017) Teachers' Expectations and Experience in Physical Computing. In: Int. Conf. on Informatics in Schools: Situation, Evolution, and Perspectives, p. 49-61. Springer. DOI: 10.1007/978-3-319-71483-7_5.

[16] Przybylla, M.; Romeike, R. (2014) Physical Computing and Its Scope--Towards a Constructionist Computer Science Curriculum with Physical Computing. *Informatics in Education* 13(2): 241-254. DOI: 10.15388/infedu.2014.05.

[17] Richard, G. T.; Kafai, Y. B. (2015) Making physical and digital games with e-textiles: a workshop for youth making responsive wearable games and controllers. In: Proc. 14th Int. Conf. on Interaction Design and Children, p. 399-402. DOI: 0.1145/2771839.2771926.

[18] Rubio, M. A. et al. (2014) Enhancing an introductory programming course with physical computing modules. In: Proc. 2014 IEEE Frontiers in Education Conference (FIE), p.1-8. IEEE. DOI: 10.1109/FIE.2014.7044153.

[19] Rubio, M. A.; Hierro, C. M.; Pablo, A. P. D. M. (2013) Using arduino to enhance computer programming courses in science and engineering. In: Proc. EDULEARN13 Conference (1-3). IATED. Barcelona, Spain.

[20] Sentance, S. et al. (2017) Teaching with physical computing devices: the BBC micro: bit initiative. In: Proc. 12th Workshop on Primary and Secondary Computing Education, p. 87-96. DOI: 10.1145/3137065.3137083.

[21] Sohn, W. (2014) Design and evaluation of computer programming education strategy using Arduino. *Advanced Science and Technology Letters* 66(1): 73-77. DOI: 10.14257/astl.2014.66.18.

[22] Stankovic, J. A. et al. (2005) Opportunities and obligations for physical computing systems. *Computer* 38(11): 23-31. DOI: 10.1109/MC.2005.386.

[23] Zanetti, H., e Oliveira, C. (2015). Práticas de ensino de Programação de Computadores com Robótica Pedagógica e aplicação de Pensamento Computacional. In: Anais dos Workshops do Congresso Brasileiro de Informática na Educação, 4(1): 1236.