Evolutionary Approach for Automatic Generation of Multi-Objective Morphological Filters for Depth Images in Embedded Navigation Systems

A. Dourado, and E. Pedrino

***Abstract*—The efforts spent on the development of assistive technologies has led researches to explore many existing techniques as computer vision, image processing, etc. and apply them as embedded solutions to help people with several types of disabilities, including visual impairment. Embedded navigation systems for visually impaired people (VIP) often use RGB-D cameras to retrieve depth information from surroundings and present them as gray images with depth represented by gray level or black pixels if depth couldn't be estimated, which can be fixed by mathematical morphology. Morphological filters must be efficient to solve the problem and fast to avoid impact on performance. This paper presents an approach for automatic generation and optimization of low complexity and low error morphological filters to fix depth image's unknown distances based on NSGA-II and Cartesian Genetic Programming. Experiments were performed using two different error metrics and results showed that the presented approach managed to generate and optimize feasible morphological filters that fit within embedded navigation systems for VIP.**

***Index Terms*—Genetic programming, Optimization methods, Morphological operations, Embedded software, Assistive technology.**

I. Introdução

Os constantes avanços obtidos na área de sistemas embarcados possibilitaram a criação de diversas aplicações em diversos segmentos que necessitam o processamento em tempo real[1]. Alguns desses operam em cenários críticos como, por exemplo, controles de aeronaves onde o tempo de processamento se torna uma preocupação. Nos sistemas embarcados de tempo real, todo o funcionamento é pautado sobre as restrições de tempo apresentadas pelas tarefas do domínio. Logo, os sistemas devem ser capazes de completar as diversas tarefas dentro do tempo máximo estipulado, utilizando um conjunto de recursos limitados de hardware embarcado como pouca memória disponível, capacidade de processamento limitada, etc[2].

Dentro do cenário de sistemas embarcados em tempo real, os sistemas de navegação para Pessoas com Deficiência Visual (PDV) propõem o uso de dispositivos computacionais portáteis para auxiliar essas pessoas no cumprimento de tarefas cotidianas como percorrer um trajeto de forma segura, identificar e reconhecer obstáculos, etc[3]. Isso pode ser feito com a obtenção de dados por um ou mais sensores de entrada, processando-os para a extração de informações importantes e as entregando em tempo hábil para que seja útil para o usuário. Os sensores mais comuns usados nesses sistemas são sensores inerciais, sensores ultrassônicos, câmeras, etiquetas de radiofrequência, laser infravermelho e outros, que são processados por algoritmos especializados no tipo de entrada fornecida. A saída ou *feedback* para as PDVs pode ser feita de forma acústica ou tátil. No *feedback* acústico ou sonoro, a informação extraída dos sensores é transformada em sinal sonoro (tons monofônicos, polifônicos, fala sintetizada, etc.) e é enviada para o usuário via periférico de som como fones de ouvido[4]. Já no *feedback* tátil, as informações são transmitidas por meio de estímulos físicos que as represente (ex: motores de vibração)[5].

Os sistemas de navegação baseados em visão computacional, em geral, utilizam câmeras RGB-D ou câmeras de profundidade para captar imagens do ambiente e mapeá-las de acordo com as diferentes profundidades/distâncias dos objetos em relação a elas, podendo empregar técnicas distintas para isso como luz estruturada, infravermelho passivo e/ou ativo, *time-of-flight*, estereoscopia, etc[6], [7], [8]. Embora cada técnica tenha suas particularidades, todas elas apresentam um problema muito comum: suas imagens de profundidade, apresentadas em níveis de cinza, normalmente contém diversas áreas com distâncias desconhecidas apresentadas como pixels pretos[9]. Tal problema pode fazer com que os sistemas de navegação processem uma imagem da maneira incorreta e enviar informações erradas para as PDVs, potencialmente colocando sua segurança em risco.

O problema de distâncias desconhecidas em imagens de profundidade pode ser corrigido utilizando Morfologia Matemática e seus operadores[10], que são frequentemente utilizados para diversos domínios de problemas[11][12]. Por meio da definição de diversas operações e seus elementos estruturantes, é possível a criação manual de um filtro que resolva tal problema. O processo manual, entretanto, tende a ser lento e exaustivo quando se tem em mente que, a cada alteração, o filtro deve ser novamente testado em todos os cenários nos quais será utilizado. A Programação Genética Cartesiana ou *Cartesian Genetic Programming* (CGP)[13] pode ser utilizada para fazer a automatização desse processo, gerando e evoluindo filtros visando a qualidade por meio de

Este trabalho foi financiado em parte pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) - Código 001 e pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) - Projeto 2017/26421-3.

A. M. B. Dourado é professor do Instituto Federal de São Paulo – Campus Boituva e doutorando da Universidade Federal de São Carlos (e-mail: dourado@ifsp.edu.br).

E. C. Pedrino é professor da Universidade Federal de São Carlos (e-mail: emerson@dc.ufscar.br).

baixas taxas de erro em relação a resultados esperados[14]. Filtros com taxas menores de erro na correção de imagens de profundidade não são o único objetivo dentro do cenário de sistemas embarcados para PDV. A complexidade desses filtros pode ter um impacto direto na performance do sistema, comprometendo-o de forma que suas informações podem não ser confiáveis, uma vez que muito tempo pode ter sido gasto apenas com o processo de filtragem[15]. Desta forma, um filtro ideal para o cenário de tempo real deve ser capaz de corrigir as distâncias desconhecidas em imagens de profundidade e apresentar a menor complexidade possível para minimizar seu impacto na performance do sistema, ou seja, dois objetivos devem ser atendidos pelo filtro. Isso pode ser resolvido com algoritmos de Otimização Multi-Objetivo ou *Multi-Objective Optimization* (MOO)[16] visando encontrar uma solução factível que contemple os dois objetivos "baixa taxa de erro x baixa complexidade" de forma satisfatória, com um *trade-off* aceitável entre eles.

Neste artigo, é apresentada uma abordagem para a automatização de geração de filtros morfológicos otimizados para a correção de imagens de profundidade utilizadas em um projeto em andamento de sistema embarcado de navegação em tempo real para PDV[17]. A abordagem utiliza CGP para a tarefa de gerar e evoluir filtros enquanto o algoritmo multi-objetivo NSGA-II os otimiza visando obter baixas taxas de erro, via dois tipos cálculo, e baixa complexidade via a quantidade de elementos estruturantes utilizados. Os resultados apresentaram filtros com qualidade satisfatória para corrigir imagens de profundidade e baixa complexidade, de forma a ter um impacto positivo na performance quando executados.

A organização deste artigo é feita em sete seções. Na Seção II são apresentados os trabalhos relacionados à correção de imagens de profundidade utilizando outras técnicas. Na Seção III é introduzida, de forma sucinta, a fundamentação sobre CGP e NSGA-II. A abordagem proposta pelos autores é apresentada na Seção IV e sua experimentação é explicada na Seção V. Os resultados são apresentados e discutidos na Seção VI. As conclusões do trabalho são apresentadas na Seção VII, bem como os trabalhos futuros.

II. Trabalhos Relacionados

Embora poucos, existem trabalhos na literatura propondo abordagens baseadas em CGP e NSGA-II. Os autores, entretanto, encontraram a maioria das aplicações voltadas para circuitos digitais visando melhorar sua eficiência, diminuir os atrasos entre os comandos ou minimizar erros[18][19]. Kalkreuth e colegas[20] propuseram uma versão do NSGA-II baseada em CGP para evoluir programas eficientes com foco na minimização da idade e tamanho dos cromossomos (indivíduos), contando sua quantidade de genes. Nenhuma imagem resultante foi apresentada e a relação "tamanho de cromossomo x performance" é imprecisa, haja visto que é possível um indivíduo ter poucas operações e elas serem complexas o suficiente para consumirem mais tempo em relação a uma série maior de operações de baixa complexidade.

Filtros morfológicos otimizados via MOO já foram propostos anteriormente para filtragem de ruídos minimizando os erros *Peak Signal-to-Ratio* (PSNR) e *Mean Squared Error* (MSE), porém sem avaliar a complexidade dos filtros[21]. O potencial também foi explorado em filtros morfológicos otimizados via operadores hipervolumétricos em imagens coloridas, além da análise de estruturas porosas em materials termoelásticos[22][23].

Trabalhos com filtros de morfologia matemática gerados por algoritmos CGP, até onde foi possível pesquisar, foram realizados apenas pelo grupo de pesquisa no qual este artigo está inserido [14][15]. Esses trabalhos foram realizados visando evoluir e implantar filtros em FPGAs, levando a indicativos que ajudaram os autores a identificar a necessidade de uma abordagem multi-objetivo.

A literatura contém diversas abordagens para corrigir o problema de distâncias desconhecidas em imagens de profundidade. A técnica denominada Inpainting foi criada para remover, substituir ou restaurar partes de imagens contendo algum tipo de dano, podendo ser aplicada no contexto de imagens de profundidade e obtendo bons resultados [24][25][9][26]. Por ser uma técnica custosa em termos de processamento como destacado por alguns autores, torna-se inviável para ser utilizada em sistemas embarcados de navegação para PDV, já que podem existir limitações de hardware e o comprometimento de informações em tempo real[27][28]. Assim como *Inpainting*, os filtros medianos e bilaterais também são capazes de corrigir imagens de profundidade, mas seu custo computacional também é alto[29][30]. A técnica de *Hole Filling* é um técnica promissora, podendo ser implementada para soluções em tempo real, mas ao custo de menor qualidade de resultado, inclusive não sendo especificado se a técnica mantém a mesma performance em sistemas embarcados com recursos limitados[31]. Assim, carece-se de uma maneira automatizada para realizar a correção dessas imagens de forma satisfatória, mantendo boa performance mesmo após embarcado em hardware com capacidade de processamento limitada. Com isso, a abordagem apresentada neste artigo pretende preencher essa lacuna aplicando NSGA-II e CGP para gerar e otimizar filtros morfológicos de forma automática, contribuindo com um novo método capaz de resolver o problema de distâncias desconhecidas em imagens de profundidade sem impacto negativo no tempo de execução.

III. Fundamentação Teórica

A. Programação Genética Cartesiana

A Programação Genética Cartesiana é um tipo de Programação Genética baseado em grafos em que cada indivíduo ou cromossomo de uma população pode ser representado na forma de um grafo direcionado, organizado em duas dimensões de genes[13]. A partir de uma série de parâmetros predeterminados, cada cromossomo é definido por sua quantidade de linhas, colunas e *levels-back*, um parâmetro que estipula a quantidade máxima de colunas anteriores que um gene pode receber como entrada. Cada gene contém uma função que é responsável por processar as entradas e enviar seu resultado como saída do gene, para ser utilizado por outros genes.

Os indivíduos de uma população CGP são submetidos a um processo de evolução utilizando, em geral, apenas mutações

como Estratégia de Evolução. De acordo com a literatura, o operador genético mais comum nas mutações é o point mutation, ou mutação pontual, que usa a estratégia $(1+\lambda = 4)$[13].

B. Algoritmo Genético de Ordenação Não-Dominante

O Algoritmo Genético de Ordenação Não-dominante ou *Nondominated Sorting Genetic Algorithm* (NSGA-II) é um algoritmo evolucionário de otimização multi-objetivo elitista[32][33]. Nele, a população é agrupada em conjuntos denominados "*fronts*" com indivíduos contendo o mesmo nível de "dominação". É dito que um indivíduo X_1 domina outro indivíduo X_2 se os objetivos de X_1 são iguais aos de X_2 e pelo menos um deles é melhor. Os "*fronts*", por sua vez, são ordenados de acordo com o "*crowding-distance*", um operador que calcula a densidade de cada indivíduo e atribui valores maiores para aqueles com menos vizinhos próximos.

Para o processo de evolução, o NSGA-II aplica a técnica 2N. Considerando uma população de tamanho N, o algoritmo gera N filhos após selecionar pais de forma aleatória e aplicar operadores de mutação e recombinação. Os N filhos são adicionados à população de pais, fazendo com que essa dobre de tamanho: 2N. A população é ordenada em *fronts* e *crowding-distance*, e os melhores N indivíduos se tornam a população para a próxima geração[34].

IV. Abordagem Proposta

O ponto principal da abordagem é focado na adaptação do NSGA-II para que seja possível trabalhar com populações de indivíduos baseados em CGP e também definir a estratégia de evolução a ser utilizada. Assim, a abordagem pode ser detalhada em três principais pontos: sua estratégia de evolução, definindo como a população evoluirá e será otimizada; a estrutura da população, que define como cada indivíduo é construído de acordo com a base CGP; e os parâmetros necessários para o treinamento da população.

A. Estratégia de Evolução

Há uma grande diferença entre as estratégias de evolução de CGP e NSGA-II. Os autores projetaram inicialmente uma forma híbrida da estratégia $(1+\lambda)$ que selecionava os melhores *fronts* ao invés de indivíduos. Porém a estratégia 2N adaptada para trabalhar com cromossomos CGP obteve performance superior. Na abordagem adaptada para 2N, o algoritmo seleciona N indivíduos e aplica um percentual predefinido de mutação em cada um, adicionando novos filhos resultantes da mutação à população até atingir o tamanho 2N. Assim, os operadores do NSGA-II para agrupamento em *fronts* e distâncias fazem a ordenação e os melhores N cromossomos seguem para a próxima geração. Na Fig. 1, o funcionamento da abordagem é ilustrado.

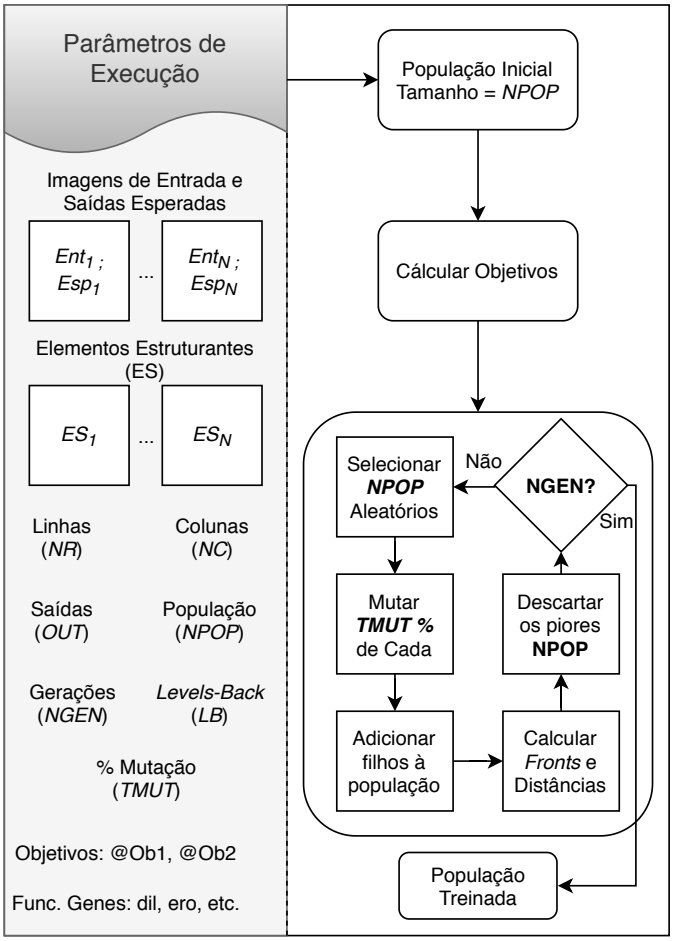


Fig. 1. Visão geral da abordagem desenvolvida.

B. População

A população agora contém indivíduos definidos por números de linhas (NR), colunas (NC), saídas (OUT) e *levels-back* (LB). Logo, um dado indivíduo é composto de $NR * NC + OUT$ genes (sendo $NR, NC \geq 1$) que podem receber entradas vindas de até LB colunas anteriores (sendo $1 \leq LB \leq NC$). Toda a população compartilha de um mesmo conjunto de entradas contendo diversos Elementos Estruturantes (EE) e uma imagem de treinamento contendo distâncias desconhecidas. Os genes de cada indivíduo recebem duas entradas de qualquer combinação de imagens ou elementos estruturantes, as processa de acordo com a função designada a ele e produz uma saída com o resultado da função, podendo ser uma imagem ou um elemento estruturante. A saída de cada gene pode ser utilizada como entrada por outros genes a partir da coluna seguinte. Na Fig. 2, é exibido um exemplo de cromossomo/indivíduo com uma linha, três colunas e *levels-back* = 3. Cada gene contém uma operação (dilatação, erosão, *and*, *or*, *not* ou *nop*) que processa suas duas entradas que podem ser um par de imagens, um par de elementos estruturantes ou um de cada. Sua saida, cujo número é o número do gene, pode servir de entrada para outro gene. O operador *nop* foi criado para funcionar como um replicador de entrada. Repassa a primeira entrada como saida se o gene foi ímpar ou a segunda entrada, caso contrário.

C. Parâmetros

Um conjunto de parâmetros de execução deve ser definido para nortear o funcionamento esperado do algoritmo desde o

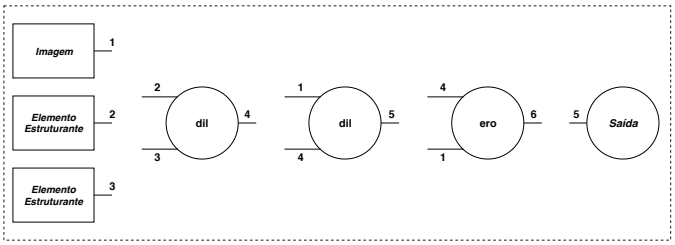


Fig. 2. Exemplo de cromossomo de uma linha, três colunas e *levels-back* = 3.

momento em que ele é executado. Na Tabela I, são listados os parâmetros necessários para realizar um treinamento com a abordagem apresentada.

TABELA I
PARÂMETROS OBRIGATÓRIOS PARA O TREINAMENTO

Parâmetro	Descrição
IM_SET	Conjunto de pares de imagem de treinamento (imagem defeituosa, imagem esperada).
SE_SET	Conjunto de elementos estruturantes
NGEN	Quantidade de gerações
NPOP	Tamanho da população
NC	Número de colunas
NR	Número de linhas
LB	Colunas de *Levels-back*
OUT	Número de saídas
TMUT	Percentual de Mutação
FITFNS	Funções de objetivo/*fitness*
GENFNS	Conjunto de funções de gene

É importante observar que o conjunto de pares de imagens de treinamento deve refletir os cenários nos quais o sistema embarcado de navegação para PDV será utilizado para maximizar seu funcionamento. Os pares de imagens são compostos por uma imagem de profundidade contendo áreas de distâncias desconhecidas e uma imagem de validação com o resultado ideal ou esperado após a aplicação de um filtro morfológico. Na Fig. 3, é apresentado um exemplo de par de imagens de profundidade que podem compor um conjunto de treinamento.

(a) (b)

Fig. 3. Par de imagens de profundidade para treinamento. Uma contendo áreas com distâncias desconhecidas (a) e sua saída esperada para validação (b).

V. EXPERIMENTOS

Para validar o funcionamento da abordagem apresentada na seção anterior, foram coletadas dez imagens de profundidade de cenários reais onde o sistema de navegação para PDV dos autores será utilizado em breve. As imagens foram coletadas por duas câmeras diferentes. A primeira foi um sensor Kinect V1 e a segunda foi Intel Realsense D435i, ambas em resolução de 640x480 pixels. Cada imagem foi duplicada e as cópias foram manualmente corrigidas para serem usadas como alvo para o cálculo de erros. Os cenários, entretanto, não se baseiam em uma localidade apenas. O sistema de navegação será testado tanto em ambiente interno quanto ambientes externos ao passo que, durante análise preliminar, os autores observaram que diferentes perspectivas de uma mesma cena não melhoram a qualidade dos filtros gerados ou sua complexidade. Logo, dez imagens foram suficientes para ambientes internos da universidade (corredores, obstáculos baixos e elevados, etc.) e externos pela cidade (calçadas com transeuntes, orelhões, parquímetros, etc.).

Os parâmetros de execução foram definidos empiricamente após diversas execuções, variando valores para números de linhas, colunas, população, gerações, *levels-back*, mutação e dimensão de elementos estruturantes. Foi observado que grandes populações geram melhores resultados em poucas gerações. O número de linhas de cada indivíduo se mostrou irrelevante, enquanto a quantidade de colunas e *levels-back* estão diretamente ligadas em termos de performance quando seus valores são próximos ou iguais. Os parâmetros de execução são apresentados na Tabela II.

TABELA II
PARÂMETROS UTILIZADOS

Parâmetro	Utilizado
IM_SET	10 pares
SE_SET	40 EEs
NGEN	100
NPOP	1000
NC	500
NR	1
LB	500
OUT	1
TMUT	25%
FITFNS	SSIM, RMSE; Complexidade
GENFNS	dilate, erode, and, or, not, nop

Como visto na Tabela II, um conjunto de quarenta elementos estruturantes foi utilizado, com seis tipos de formas: retângulo, quadrado, losango, octógono, círculo e linha. As dimensões de cada EE variam entre 1, 3, 5 e 7, sendo observado que EEs maiores não contribuem positivamente para a qualidade ou complexidade dos filtros gerados. O EE em forma de linha também tem seu ângulo variando entre 0º, 45º, 90º e 135º.

A otimização é feita sobre duas funções de objetivo, baixa taxa de erro e complexidade. As melhores soluções devem ser de baixa complexidade, para garantir boa performance em sistemas embarcados, e ainda ser capaz de corrigir imagens de profundidade satisfatoriamente, apresentando o menor erro possível. Para o primeiro objetivo, duas métricas foram utilizadas para calcular o erro entre imagens de treinamento (x) e imagens de validação/alvo (y) durante o treinamento: SSIM (Eq. 1), a similaridade estrutural entre x e y, normalizadas por C_1 e C_2 [35]; e RMSE (Eq. 2), a raíz do erro médio

quadrado[36].

$$SSIM_{x,y} = 1 - \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)} \quad (1)$$

$$RMSE_{x,y} = \sqrt{\frac{1}{N * M}\sum_{i=1}^{N}\sum_{j=1}^{M}(x(i,j) - y(i,j))^2} \quad (2)$$

O segundo objetivo, *Complexidade* (Eq. 3), é dado pela soma dos tamanhos de todos os elementos estruturantes *listaEE(1...S)* utilizados pelo indivíduo. Assim, o valor da complexidade tem impacto direto na performance do filtro, já que complexidades altas significam mais operações realizadas.

$$Complexidade = \sum_{i=1}^{S} count(listaEE(i)) \quad (3)$$

VI. Resultados e Discussão

Os resultados foram baseados em duas sessões de treinamento, sendo a primeira considerando a métrica RMSE e a segunda, SSIM. Para ambos os casos, o segundo objetivo foi o mesmo, a complexidade. Assim, os autores realizaram a análise dos resultados selecionando os indivíduos com o menor erro médio e, consequentemente, maior complexidade, e uma solução factível, que foi capaz de corrigir imagens de profundidade de maneira satisfatória mantendo uma complexidade aceitável, totalizando quatro soluções. Os resultados também apresentam os tempos de execução das soluções para cada imagem utilizada com cenários reais de uso Os filtros otimizados foram implementados na linguagem C++ e sua performance aferida no módulo embarcado NVIDIA Jetson Nano. Na Tabela III, são apresentados os resultados da melhor solução de cada um dos dois erros, considerando o menor erro obtido, para cada imagem, a complexidade dos filtros (*Compl*) e o tempo de execução em milissegundos (T_{ms}).

TABELA III
Resultados de erros, complexidade (*Compl*) e tempo (T_{ms}) em milissegundos para os erros RMSE e SSIM das soluções com menores erros

	RMSE			*SSIM*		
	Erro	*Compl*	T_{ms}	*Erro*	*Compl*	T_{ms}
Img_{01}	0.0846	1009	9.2	0.0271	3200	15.5
Img_{02}	0.0418	1009	8.3	0.0176	3200	15.0
Img_{03}	0.0811	1009	8.5	0.0900	3200	15.8
Img_{04}	0.0834	1009	8.3	0.0729	3200	15.9
Img_{05}	0.0511	1009	8.1	0.0356	3200	15.1
Img_{06}	0.0248	1009	8.0	0.0119	3200	15.0
Img_{07}	0.0543	1009	8.1	0.0389	3200	15.1
Img_{08}	0.1464	1009	8.3	0.0111	3200	14.9
Img_{09}	0.0953	1009	8.5	0.0723	3200	15.5
Img_{10}	0.0784	1009	8.4	0.0240	3200	15.1

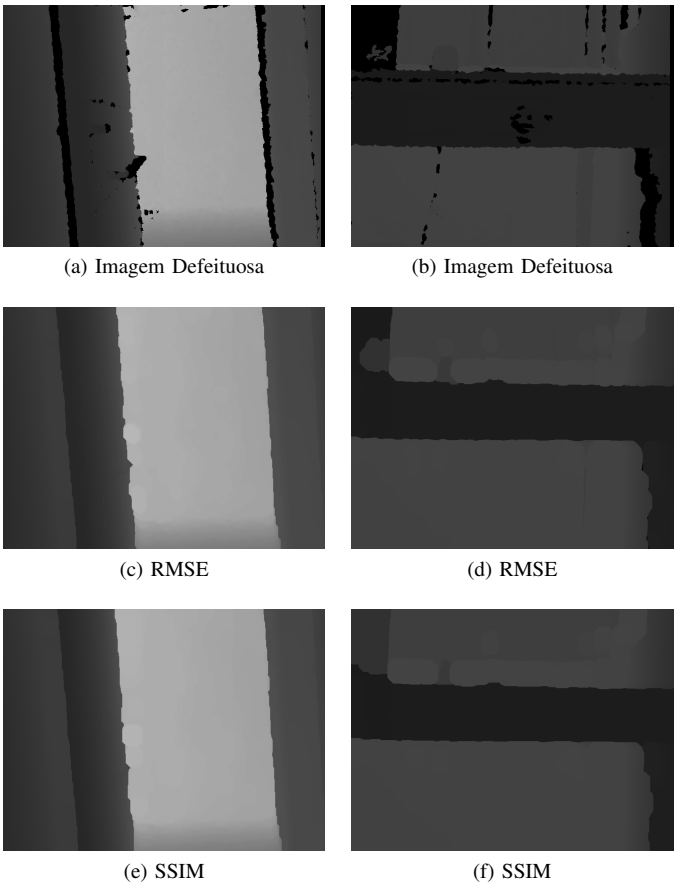
De acordo com os dados apresentados na Tabela III, observa-se uma grande diferença entre as métricas em relação tanto ao valor dos erros, quanto à complexidade e tempo de execução. A melhor solução de SSIM visando apenas qualidade é 3x mais complexa em comparação à RMSE, porém com taxas de erros consideravelmente menores. Isso também se reflete no tempo de execução, sendo RMSE quase 50% mais rápida que SSIM para realizar a filtragem. Porém, é necessário considerar também o segundo objetivo, complexidade, na escolha de soluções aceitáveis para serem utilizadas como filtros morfológicos em um sistema de navegação para PDV. Assim, os autores selecionaram uma solução aceitável treinada com SSIM e outra com RMSE para avaliação. Os resultados são apresentados na Tabela IV.

TABELA IV
Resultados de erros, complexidade (*Compl*) e tempo (T_{ms}) em milissegundos para os erros RMSE e SSIM das soluções factíveis escolhidas

	RMSE			*SSIM*		
	Erro	*Compl*	T_{ms}	*Erro*	*Compl*	T_{ms}
Img_{01}	0.0870	557	6.8	0.1069	410	5.3
Img_{02}	0.0445	557	6.7	0.0346	410	4.8
Img_{03}	0.0807	557	6.6	0.0740	410	5.0
Img_{04}	0.0812	557	6.4	0.0605	410	4.9
Img_{05}	0.0512	557	6.3	0.0343	410	4.8
Img_{06}	0.0259	557	6.6	0.0305	410	4.8
Img_{07}	0.0559	557	6.5	0.0656	410	4.9
Img_{08}	0.1379	557	6.6	0.0741	410	5.1
Img_{09}	0.0898	557	6.4	0.0772	410	5.1
Img_{10}	0.0652	557	6.4	0.0477	410	4.9

Os erros apresentados por RMSE na Tabela IV são muito similares aos apresentados pela melhor solução da mesma métrica, apesar de ser pouco menos de 50% menos complexa. Já a solução factível para SSIM demonstraram comportamento esperado já que, em comparação com sua melhor solução de qualidade, teve um aumento aceitável na taxa de erro, mas com aproximadamente 87% menos complexidade e ainda foi melhor que RMSE, tanto na solução factível quanto na melhor solução. Como esperado, o filtro factível otimizado com a métrica SSIM obteve performance superior ao RMSE devido à complexidade menor, chegando a ser, em média, 23% mais rápido e ainda apresentar menores taxas de erro. Na Fig. 4, é exibida a qualidade visual que as soluções factíveis de RMSE e SSIM produziram quando aplicadas duas imagens de fora do conjunto de treinamento. É possível observar que, visualmente, os resultados são parecidos e considera-se satisfatória a qualidade da filtragem para o contexto de sistemas de navegação para PDV. Porém, em uma análise mais detalhada, os autores encontraram pequenas regiões não corrigidas por RMSE (Fig. 4d). De forma geral, ambas soluções apresentaram bons resultados, com destaque ao filtro otimização com SSIM.

Em relação a outras abordagens da literatura, a técnica de *Inpainting* foi implementada e executada para as dez imagens de treinamento e seus resultados foram comparados às saídas esperadas com $SSIM$, alem de seus tempos de execução aferidos. Testes preliminares realizados pelos autores mostraram que *Inpainting* gera resultados com melhor qualidade que outras alternativas da literatura como *Hole Filling* e filtros recursivos, com tempo de execução relativamente



(a) Imagem Defeituosa (b) Imagem Defeituosa

(c) RMSE (d) RMSE

(e) SSIM (f) SSIM

Fig. 4. Resultados visuais da aplicação dos filtros factíveis.

baixo. Portanto, a comparação foi realizada diretamente com o metodo *Inpainting* apenas, como é mostrado na Tabela V.

TABELA V
RESULTADOS DE ERROS, COMPLEXIDADE (*Compl*) E TEMPO (T_{ms}) EM MILISSEGUNDOS PARA O ERRO SSIM DAS SOLUÇÕES FACTÍVEIS ESCOLHIDAS NA TABELA IV E A TÉCNICA DE INPAINTING

	Inpainting			*Proposto*		
	Erro	*Compl*	T_{ms}	*Erro*	*Compl*	T_{ms}
Img_{01}	0.1135	–	131	0.1069	410	5.3
Img_{02}	0.0948	–	154	0.0346	410	4.8
Img_{03}	0.1066	–	151	0.0740	410	5.0
Img_{04}	0.1014	–	136	0.0605	410	4.9
Img_{05}	0.0785	–	121	0.0343	410	4.8
Img_{06}	0.0562	–	138	0.0305	410	4.8
Img_{07}	0.0696	–	125	0.0656	410	4.9
Img_{08}	0.1713	–	134	0.0741	410	5.1
Img_{09}	0.1245	–	135	0.0772	410	5.1
Img_{10}	0.0963	–	143	0.0477	410	4.9

Como observado na Tabela V, o método Inpainting apresenta resultados piores que o proposto pelos autores em termos de qualidade e, principalmente, velocidade. Apesar do intervalo de 120 a 150 milissegundos ser baixo, ao considerar que a filtragem é apenas uma das várias tarefas que um sistema de navegação para deficientes visuais realiza, fazendo com que as soluções factíveis apresentem tempos até 32 vezes menores.

VII. CONCLUSÃO

Neste artigo foi apresentada uma abordagem para gerar, de forma automática, filtros morfológicos para corrigir áreas de distâncias desconhecidas em imagens de profundidade usadas em sistemas embarcados de navegação para PDV. A abordagem apresentada foi projetada para, além de gerar, otimizar tais filtros com foco em baixo erro e baixa complexidade, visando manter boa performance quando utilizados em arquiteturas embarcadas com hardware limitado. O funcionamento da abordagem foi ilustrado, bem como foram listados os parâmetros obrigatórios.

Dois treinamentos foram realizados, alterando apenas a métrica do primeiro objetivo, os erros RMSE e SSIM. Os resultados mostraram que a abordagem conseguiu gerar automaticamente filtros morfológicos capazes de corrigir as distâncias desconhecidas nas imagens de profundidade. Comparações quantitativas entre os erros da melhor solução de cada métrica e soluções factíveis escolhidas pelos autores mostraram que SSIM se adapta melhor para o contexto deste trabalho, principalmente nas soluções factíveis, com melhor qualidade e baixo tempo de execução devido à baixa complexidade. A solução factivel de $SSIM$ foi comparada com o método *Inpainting* da literatura, gerando resultados que reforçam que a abordagem dos autores é solida e capaz de gerar filtros morfológicos que corrigem imagens de profundidade não apenas com boa qualidade, mas tambem baixa complexidade compatíveis com hardwares limitados.

A abordagem apresentada já tem sido utilizada para gerar filtros morfológicos para um projeto em andamento de sistema embarcado de navegação para PDV, de acordo com cada cenário de uso. Como trabalhos futuros, os autores pretendem testar outras métricas de erro, visando aprimorar os filtros gerados. Pretende-se também criar bibliotecas que sejam capazes de exportar os filtros gerados para outras linguagens de programação, prontos para execução. Por fim, futuramente a abordagem será formalizada com o objetivo de atrair a colaboração da comunidade para expandi-la.

AGRADECIMENTOS

O presente trabalho foi financiado pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) – Código de Financiamento 001 e pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPES) – Projeto 2017/26421-3. Agradecimentos também ao Instituto Federal de São Paulo e à O.N.G. Para-D.V. que contribuíram de forma a tornar este trabalho possível.

REFERÊNCIAS

[1] A. G. Alvarez, D. A. Evin, and S. Verrastro, "Implementation of a Speech Recognition System in a DSC," *IEEE Latin America Transactions*, vol. 14, no. 6, pp. 2657–2662, jun 2016.

[2] A. Barkalov, L. Titarenko, and M. Mazurkiewicz, *Foundations of Embedded Systems*, ser. Studies in Systems, Decision and Control. Cham: Springer International Publishing, 2019, vol. 1.

[3] M. Poggi and S. Mattoccia, "A Wearable Mobility Aid for the Visually Impaired based on embedded 3D Vision and Deep Learning," in *First IEEE Workshop on ICT Solutions for eHealth (IEEE ICTS4eHealth 2016) in conjunction with the Twenty-First IEEE Symposium on Computers and Communications*, 2016.

[4] B. Schauerte, M. Martinez, A. Constantinescu, and R. Stiefelhagen, "An assistive vision system for the blind that helps find lost things," *Computational Science and Its Applications–ICCSA 2007*, vol. 7383 LNCS, no. Chapter 83, pp. 566–572, jan 2012.
[5] L. Zeng, "Non-visual 2D representation of obstacles," *ACM SIGACCESS Accessibility and Computing*, no. 102, pp. 49–54, jan 2012.
[6] C. D. Mutto, P. Zanuttigh, and G. M. Cortelazzo, "TOF Cameras and Stereo Systems: Comparison and Data Fusion," in *TOF Range-Imaging Cameras*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, vol. 9783642275, pp. 177–202.
[7] S. Mattoccia and M. Poggi, "A passive RGBD sensor for accurate and real-time depth sensing self-contained into an FPGA," *Proceedings of the 9th International Conference on Distributed Smart Camera - ICDSC '15*, pp. 146–151, 2015.
[8] L. Dunai, B. D. Garcia, I. Lengua, and G. Peris-Fajarnes, "3D CMOS sensor based acoustic object detection and navigation system for blind people," in *IECON Proceedings (Industrial Electronics Conference)*, jan 2012, pp. 4208–4215.
[9] F. Qi, J. Han, P. Wang, G. Shi, and F. Li, "Structure guided fusion for depth map inpainting," *Pattern Recognition Letters*, vol. 34, no. 1, pp. 70–76, jan 2013.
[10] E. C. Pedrino and V. O. Roda, "Real-time morphological pipeline architecture using high-capacity programmable logical devices," *Journal of Electronic Imaging*, vol. 16, no. 2, p. 023002, apr 2007.
[11] J. I. Pastore, E. Moler, and V. Ballarin, "Operadores morfológicos multi-escala y distancia geodésica aplicados a la segmentación de imágenes de tomografía axial computada," *IEEE Latin America Transactions*, vol. 5, no. 1, pp. 28–31, 2007.
[12] M. V. Rodríguez, L. A. Hernández, J. P. Rangel, and A. D. González, "Real-Time Monitoring of Voltage Variations using Mathematical Morphology," *IEEE Latin America Transactions*, vol. 14, no. 5, pp. 2138–2145, 2016.
[13] J. F. Miller and P. Thomson, "Cartesian Genetic Programming," in *Genetic Programming*. Berlin, Heidelberg: Springer Berlin Heidelberg, jan 2000, vol. 1802, no. Chapter 9, pp. 121–132.
[14] P. Paris, E. Pedrino, and M. Nicoletti, "Automatic learning of image filters using Cartesian genetic programming," *Integrated Computer-Aided Engineering*, vol. 22, no. 2, pp. 135–151, feb 2015.
[15] E. C. Pedrino, J. H. Saito, E. R. R. Kato, O. Morandin, L. M. Del Val Cura, V. O. Roda, M. L. Tronco, and R. H. Tsunaki, "Automatic construction of image operators using a genetic programming approach," in *2011 11th International Conference on Intelligent Systems Design and Applications (ISDA)*. IEEE, aug 2015, pp. 636–641.
[16] J. Branke, J. Branke, K. Deb, K. Miettinen, and R. Slowiński, *Multiobjective optimization: Interactive and evolutionary approaches*. Springer Science & Business Media, 2008, vol. 5252.
[17] A. M. B. Dourado and E. C. Pedrino, "Embedded Navigation and Classification System for Assisting Visually Impaired People," pp. 516–523, 2018.
[18] R. Hrbacek, V. Mrazek, and Z. Vasicek, "Automatic design of approximate circuits by means of multi-objective evolutionary algorithms," in *2016 International Conference on Design and Technology of Integrated Systems in Nanoscale Era (DTIS)*. IEEE, apr 2016, pp. 1–6.
[19] J. Hilder, J. A. Walker, and A. Tyrrell, "Use of a multi-objective fitness function to improve cartesian genetic programming circuits," in *2010 NASA/ESA Conference on Adaptive Hardware and Systems*. IEEE, jun 2010, pp. 179–185.
[20] R. Kalkreuth, G. Rudolph, and J. Krone, "More efficient evolution of small genetic programs in Cartesian Genetic Programming by using genotypie age," in *2016 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, jul 2016, pp. 5052–5059.
[21] L. Dou, D. Xu, H. Chen, and Y. Liu, "Image de-noising based on mathematical morphology and multi-objective particle swarm optimization," in *Ninth International Conference on Digital Image Processing (ICDIP 2017)*, C. M. Falco and X. Jiang, Eds., vol. 10420, no. Icdip, jul 2017, p. 104202I.
[22] J. Deng, J. Yan, and G. Cheng, "Multi-objective concurrent topology optimization of thermoelastic structures composed of homogeneous porous material," *Structural and Multidisciplinary Optimization*, vol. 47, no. 4, pp. 583–597, 2013.
[23] M. Koppen and K. Franke, "Pareto-dominated Hypervolume Measure: An Alternative Approach to Color Morphology," in *7th International Conference on Hybrid Intelligent Systems (HIS 2007)*. IEEE, sep 2007, pp. 234–239.
[24] Y. Keaomanee and P. Youngkong, "RGB-D Depth Inpainting with Color Guide Inverse Distance Weight," in *2019 4th International Conference on Information Technology (InCIT)*. IEEE, oct 2019, pp. 249–253. [Online]. Available: https://ieeexplore.ieee.org/document/8912139/
[25] Atapour-abarghouei and Amir, "Immaculate Depth Perception: Recovering 3D Scene Information via Depth Completion and Prediction," PhD, Durham University, 2019.
[26] A. Hervieu, N. Papadakis, A. Bugeau, P. Gargallo, and V. Caselles, "Stereoscopic Image Inpainting: Distinct Depth Maps and Images Inpainting," in *2010 20th International Conference on Pattern Recognition*. IEEE, aug 2010, pp. 4101–4104.
[27] R. Liu, Z. Deng, L. Yi, Z. Huang, D. Cao, M. Xu, and R. Jia, "Hole-filling Based on Disparity Map and Inpainting for Depth-Image-Based Rendering," *International Journal of Hybrid Information Technology*, vol. 9, no. 5, pp. 145–164, may 2016.
[28] M. A. Garduño-Ramón, I. R. Terol-Villalobos, R. A. Osornio-Rios, and L. A. Morales-Hernandez, "A new method for inpainting of depth maps from time-of-flight sensors based on a modified closing by reconstruction algorithm," *Journal of Visual Communication and Image Representation*, vol. 47, pp. 36–47, aug 2017.
[29] Li Chen, Hui Lin, and Shutao Li, "Depth image enhancement for Kinect using region growing and bilateral filter," pp. 3070–3073, 2012.
[30] K. Lai, L. Bo, X. Ren, and D. Fox, "A large-scale hierarchical multi-view RGB-D object dataset," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, may 2011, pp. 1817–1824.
[31] K. Raviya, V. V. Dwivedi, A. Kothari, and G. Gohil, "Real Time Depth Hole Filling using Kinect Sensor and Depth Extract from Stereo Images," *Oriental journal of computer science and technology*, vol. 12, no. Issue 3, pp. 115–122, 2019.
[32] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.
[33] F. De Fatima Do Nascimento Silva, D. Lopes Martins, A. Duarte Doria Neto, M. Allysson Felipe Rodrigues, and W. Da Mata, "Optimization of the Oil Production Fields Submitted the Water Injection, Using the Algorithm NSGA-II," *IEEE Latin America Transactions*, vol. 14, no. 9, pp. 4166–4172, 2016.
[34] D. Kalyanmoy, *Multi-objective optimization using evolutionary algorithms*, 1st ed. John Wiley and Sons, 2001.
[35] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image Quality Assessment: From Error Visibility to Structural Similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, apr 2004.
[36] T. Chai and R. R. Draxler, "Root mean square error (RMSE) or mean absolute error (MAE)? -Arguments against avoiding RMSE in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014.

Antonio Miguel Batista Dourado possui graduação em Ciência da Computação na Universidade Eurípides de Marília (UNIVEM) e Mestrado em Ciência da Computação na Universidade Federal de São Carlos (UFSCar). Atualmente é professor efetivo do Instituto Federal de São Paulo no Campus Boituva e doutorando em Ciência da Computação na Universidade Federal de São Carlos, desenvolvendo tecnologias e sistemas assistivos para auxiliar na navegação autônoma de pessoas com deficiência visual.

Emerson Carlos Pedrino é professor (Dr.) Associado do Departamento de Computação da Universidade Federal de São Carlos. Possui graduação em Engenharia Elétrica Eletrônica pela Universidade de São Paulo e em Bacharelado em Física Computacional também pela Universidade de São Paulo, - EESC (2016) e IFSC (2000) -, Mestrado em Engenharia Elétrica pela Universidade de São Paulo - EESC (2003) - Doutorado em Engenharia Elétrica pela Universidade de São Paulo - EESC (2008) - e Pós-doutorado em Engenharia Eletrônica no Departamento de Engenharia Eletrônica da Universidade de York, Inglaterra.