Utilizando a Arquitetura UNet++ na Estimativa de Profundidade Monocular

Luiz Antonio Roque Guzzo, Kelly Assis de Souza Gazolli

[1]Instituto Federal do Espírito Santo - IFES
Serra - ES

luizguzzo@gmail.com, kasouza@ifes.edu.br

Abstract. *With the rise of convolutional neural networks, many approaches have been proposed aiming to improve the results in depth estimation, but disregarding computational costs. In this work, we present an approach that uses the UNet++ architecture, employing a MobileNetV2 network as the encoder, generating a lightweight structure with fewer parameters. Experiments performed on the NYU Depth V2 dataset showed that it is possible to achieve better results when compared to previous works, while maintaining a simpler structure.*

Resumo. *Com o surgimento das redes convolucionais, muitas abordagens foram propostas visando melhorar os resultados na estimativa de profundidade, mas desconsiderando os custos computacionais. Neste trabalho, apresentamos uma abordagem que utiliza a arquitetura UNet++, empregando uma rede MobileNetV2 como codificador, gerando uma estrutura mais leve, com um número menor de parâmetros. Os experimentos realizados na base NYU Depth V2 mostraram que é possível alcançar melhores resultados quando comparado a trabalhos anteriores, mantendo, no entanto, uma estrutura mais simples.*

1. Introdução

A estimativa de profundidade é uma das tarefas mais básicas e importantes da visão computacional [Zhao et al. 2020], uma vez que aprimora a percepção e a compreensão de cenas tridimensionais reais, levando a uma ampla gama de aplicações [Ming et al. 2021], como navegação robótica, direção autônoma e realidade virtual.

A informação de profundidade é essencial para o funcionamento de alguns sistemas. Na navegação autônoma, por exemplo, a tecnologia utilizada necessita de um forte conhecimento de seus arredores para operar com segurança, requerendo a utilização de mecanismos responsáveis pelo mapeamento do ambiente no qual a plataforma está inserida, ajudando-a a compreender o espaço tridimensional ao seu redor [Mancini et al. 2017]. Portanto, é essencial que as plataformas robóticas consigam explorar as profundidades do ambiente e estimar a distância na qual os objetos estão posicionados [de Queiroz Mendes et al. 2021].

Há algumas tecnologias amplamente comercializadas para a solução de detecção de profundidade, sendo algumas delas: LiDAR (*Light Detection and Ranging*), câmeras RGBD, câmeras estéreas, radar e sonar. Porém, esses dispositivos são pesados, têm um alto custo e um consumo elevado de energia. Tais limitações os tornam inadequados para algumas aplicações, como as utilizadas em plataformas robóticas leves (micro-veículos

aéreos e mini-terrestres, etc.) [Wofk et al. 2019] ou em sistemas de navegação cirúrgica [Chen et al. 2017, Liu et al. 2020].

Nesse contexto, a estimativa de profundidade monocular tem se apresentado como uma alternativa, uma vez que as câmeras monoculares demandam um menor espaço de instalação, são de baixo custo, captam imagens RGB, têm tamanho compacto, são robustas às condições ambientais, eficientes em termos de consumo de energia e são uma tecnologia difundida [Ma and Karaman 2018, Wofk et al. 2019].

A utilização das de Redes Neurais Convolucionais (CNNs, do inglês, *Convolutional Neural Networks*) tem sido bastante comum na estimativa de profundidade monocular [Xu et al. 2017, Alhashim and Wonka 2019, Xu et al. 2018]. Recentemente, a arquitetura Transformer [Vaswani et al. 2017] tem sido aplicada nessa tarefa [Agarwal and Arora 2022, Bhat et al. 2022, Ranftl et al. 2021], levando a um aumento da acurácia, bem como, dos custos computacionais e do tempo de latência. Como muitas aplicações têm limites de processamento, além de estarem sujeitas às restrições de latência, é necessário encontrar um equilíbrio entre os consumo dos recursos computacionais e a acurácia do algoritmo [Wofk et al. 2019].

Com o intuito de desenvolver uma solução mais leve, produzindo uma rede com tamanho reduzido, mas sem sacrificar a acurácia, este trabalho propõe uma abordagem para a estimativa de profundidade utilizando a arquitetura de rede codificador-decodificador UNet++ [Zhou et al. 2018], além de empregar como codificador a rede neural MobileNetV2 pré-treinada, que é uma rede convolucional projetada para aplicações embarcadas com enfoque visual. Para os experimentos, foi utilizada a versão reduzida da base de dados NYU Depth V2 [Silberman et al. 2012], conforme proposto por [Alhashim and Wonka 2019]. Os resultados obtidos apontam a eficácia da abordagem proposta ao alcançar resultados superiores em comparação a trabalhos anteriores, mesmo utilizando uma rede de menor complexidade, com 5 milhões de parâmetros.

Este trabalho está organizado da seguinte forma: na Seção 2 é feita uma revisão da literatura; na Seção 3, é apresentada a metodologia utilizada, detalhando-se a arquitetura e os seus componentes; na Seção 4, são apresentados os experimentos realizados e uma breve discussão dos resultados obtidos; por fim, na seção 5, é apresentada a conclusão e os trabalhos futuros.

2. Revisão da Literatura

2.1. Redes Neurais Convolucionais

Arquiteturas que empregam Redes Neurais Convolucionais (CNN - *Convolutional Neural Networks*) têm sido uma escolha frequente para resolver diversas tarefas em visão computacional. Com o passar do tempo, diversas estruturas como a ResNet [He et al. 2016], DenseNet [Huang et al. 2017] e VGG [Simonyan and Zisserman 2015], foram propostas, focando na eficácia dos resultados. Contudo, a busca incessante por avanços levou à criação de redes cada vez menos eficientes em termos de tamanho e velocidade, tornando-as inadequadas para implementação em aplicações de tempo real ou destinadas a plataformas de baixo consumo, com limitações de recursos computacionais.

Com o intuito de tratar o problema mencionado, arquiteturas mais leves foram propostas, por exemplo, MobileNet [Howard et al. 2017], GhostNet [Han et al. 2020], Fbnet

[Wu et al. 2019], SqueezeNet [Iandola et al. 2016] e ShuffleNet [Zhang et al. 2017].

A rede MobileNet foi desenvolvida com o propósito de atender aplicações móveis ou de visão integrada, com enfoque principal na otimização da latência [Howard et al. 2017]. Uma nova versão chamada "MobileNetV2"foi criada com o intuito de aumentar ainda mais a eficiência em comparação com a versão original [Sandler et al. 2018]. Na MobileNetV2, foram introduzidos os Blocos Residuais Invertidos e os Gargalos Lineares (do inglês, *Inverted Residuals* e *Linear BottleNecks*), possibilitando a construção de redes neurais profundas mais leves [Sandler et al. 2018, Deng et al. 2021].

Os Blocos Residuais Invertidos, diferentemente dos blocos residuais convencionais, expandem o número de canais na camada de entrada e reduzem na camada de saída. Já os Gargalos Lineares se referem à aplicação de uma função de ativação linear na última camada de cada bloco residual, o que ajuda a preservar informações em espaços de baixa dimensão [Sandler et al. 2018]. Essas inovações incorporadas na MobileNetV2 possibilitam a construção de uma rede mais eficiente em termos de parâmetros e de utilização de memória, sem sacrificar a qualidade dos resultados.

2.2. Estimativa de Profundidade Monocular

Com o avanço das redes neurais, muitas abordagens baseadas em aprendizado profundo foram propostas para resolver o problema de estimativa de profundidade monocular. Uma abordagem popular é a utilização das redes neurais convolucionais (CNNs) para realizar tal tarefa. As CNNs aplicam uma série de filtros convolucionais para extrair características de uma imagem e usá-las para realizar as estimativas. Ademais, alguns trabalhos têm explorado a utilização de sequências de imagens provenientes de vídeos como uma abordagem para aumentar a eficácia dos resultados [Wofk et al. 2019, Luo et al. 2020, Zhang et al. 2019, Liu and Zhu 2018, Ali et al. 2021].

Dentre as propostas baseadas em CNNs, há aquelas que utilizam arquiteturas do tipo codificador-decodificador, como a U-Net. Inicialmente desenvolvida para a segmentação de imagens biomédicas [Ronneberger et al. 2015], a U-Net é uma rede neural profunda que consiste em um codificador e um decodificador simétrico. A U-Net é uma estrutura bastante utilizada para estimativa de profundidade [Alhashim and Wonka 2019, Cantrell et al. 2020, Ramamonjisoa and Lepetit 2019], pois, como utiliza conexões de salto (*skip connections*), essa arquitetura permite um fluxo de informações eficiente e funciona bem quando um grande conjunto de dados não está disponível [Zheng et al. 2023].

Proposta por [Zhou et al. 2018], a estrutura UNet++ é uma arquitetura baseada em conexões de saltos densos e aninhados. A convolução densa consiste em aplicar várias camadas de convolução sobre os mesmos mapas de características de entrada, gerando novos mapas de características que são concatenados com os anteriores. Isso permite que a rede aprenda diferentes níveis de abstração e detalhes das características extraídas. Na UNet++, esses mapas de características são conectados diretamente às sub-redes decodificadoras correspondentes, facilitando a fusão de informações semânticas entre o codificador e o decodificador, o que potencialmente favorece a tarefa de aprendizado.

3. Metodologia

Para alcançar os objetivos propostos, a estrutura UNet++ foi utilizada neste trabalho, adotando-se a rede MobileNetV2 como o codificador da arquitetura. Serão abordados nesta seção, os detalhes da arquitetura adotada, assim como o codificador e decodificador utilizados, e a função de perda.

3.1. Arquitetura da Rede

A arquitetura da UNet++ é uma estrutura do tipo codificador-decodificador. O codificador é responsável por extrair mapas de características de uma imagem em várias resoluções e níveis de detalhes. Cada mapa é passado para os blocos de convolução densa do decodificador que, gradualmente, gera mapas com resoluções aumentadas. Esses mapas são concatenados e refinados para formar, no caso deste trabalho, o mapa da estimativa de profundidade final de alta resolução na saída.

A Figura 1 apresenta o funcionamento da arquitetura empregada. Nela, uma imagem RGB é fornecida como entrada para a rede MobileNetV2, que produz vários mapas de características. Desses, alguns são selecionados (blocos amarelos nomeados *Encoder Feature*) e usados como parâmetros de entrada para a arquitetura principal UNet++, passando, assim, pelos blocos de convolução (blocos azuis denominados *Network Block*). Cada *Network Block* contém uma sequência de operações de convolução e ativação, especificamente, uma Conv3x3, seguida por uma função de ativação LeakyRelu. Essa sequência é repetida duas vezes. As setas pretas representam conexões de salto, indicando que o mapa de características da origem é concatenado com o mapa de entrada do bloco destino. Já as setas verdes demonstram o processo de *upsample* do mapa de características, onde a resolução é aumentada e o número de canais, reduzido.

Conforme os blocos vão se aproximando da borda do decodificador, mais mapas de características são concatenados, formando, por fim, um único mapa de características com vários canais. Esse mapa resultante passa por uma convolução de filtro pontual (*point-wise*) que irá retornar uma imagem de mesma resolução com apenas um canal, tornando-se a imagem final, que é a estimativa de profundidade.

Na arquitetura adotada, é utilizado o método mais eficiente de supervisão da UNet++, proposto em [Zhou et al. 2018], que consiste em realizar o cálculo da perda da rede apenas no último bloco convolucional da arquitetura.

3.2. Codificador

O codificador utilizado neste trabalho é a rede neural MobileNetV2, uma vez que ela é projetada especificamente para dispositivos móveis e visa reduzir a carga computacional e a memória necessária para a sua execução.

A rede Mobilenet, bem como a MobilenetV2, utiliza camadas convolucionais separáveis, ou seja, enquanto uma camada convolucional convencional utiliza um kernel MxMxN, sendo M o tamanho do kernel e N, o número de canais, a MobileNet realiza N convoluções MxM separadamente e, em seguida, uma camada convolucional pontual (1x1xN). Essa abordagem permite que a rede aprenda recursos com menos parâmetros, tornado-se, desse modo, mais eficiente do que as outras redes como ResNet e DenseNet [Howard et al. 2017].

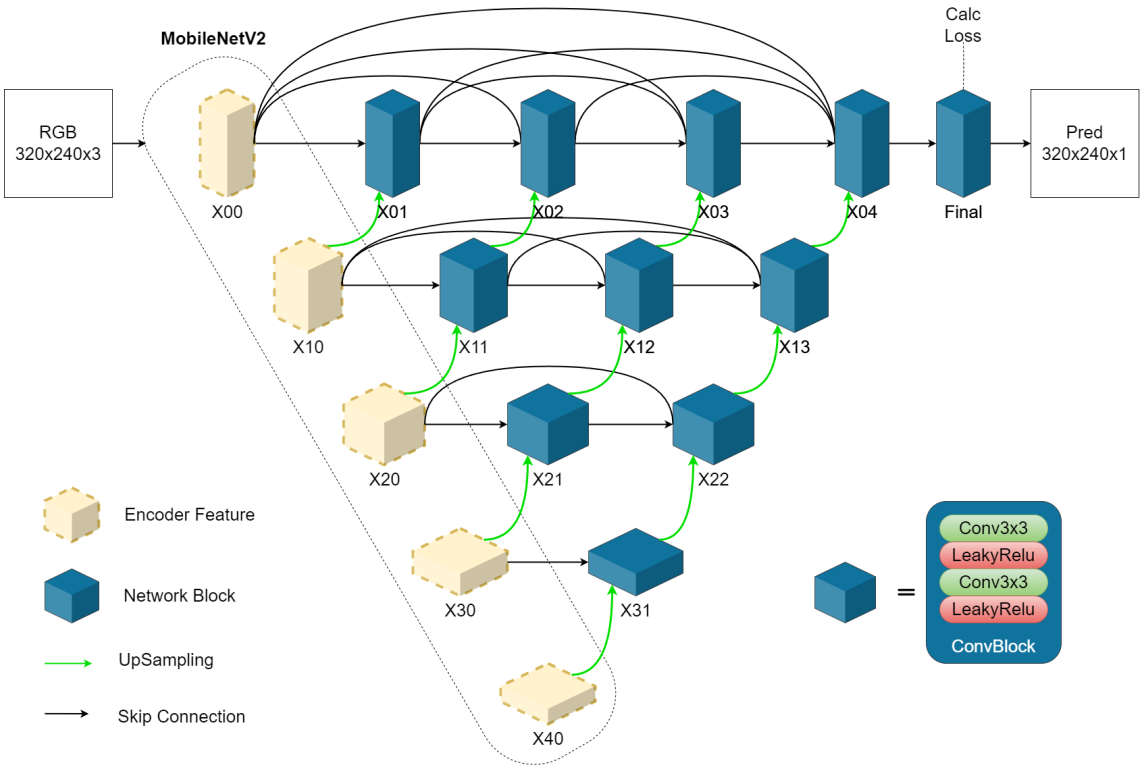

Figura 1. A arquitetura proposta: o codificador utilizado na UNet++ é uma rede MobileNetV2, cujos mapas de características gerados alimentam os blocos de convoluções.

3.3. Decodificador

O objetivo geral do decodificador é reconstruir a representação fornecida como entrada a partir de uma representação gerada por um codificador, ou seja, é produzir uma saída que corresponda à entrada original com o menor erro possível. A arquitetura UNet++ utiliza uma abordagem de multi-resolução para melhorar a precisão de suas saídas. Ela possui várias ramificações de *upsampling* utilizadas para reconstruir a imagem. Cada ramificação é responsável por produzir uma saída em uma determinada escala de resolução e com um determinado número de canais, que é então combinada com as saídas das outras ramificações para gerar a imagem final. [Zhou et al. 2018]

A cada ramificação do decodificador há uma camada de *upsampling* seguida por um bloco de convolução com camadas de ativação. A camada de *upsampling* possui a responsabilidade de aumentar a resolução da imagem recebida por meio de uma operação de convolução transposta. Já o bloco de convolução possui duas convoluções e duas camadas de ativação na seguinte ordem: convolução, ativação, convolução, ativação. O bloco serve para reduzir a dimensionalidade do mapa de características de entrada, ou seja, redução do número de canais, ajudando a reduzir o número de parâmetros e acelerar o processo de treinamento. As duas camadas de ativação no bloco servem para introduzir a não linearidade no processo.

Ao final de cada ramificação do decodificador, as saídas são combinadas por meio de uma operação de concatenação nos eixos de canais, juntando os mapas de características, combinando informações de várias escalas de resolução. Esse mapa é fornecido

como entrada para um bloco de convolução final, que reduz a sua dimensionalidade, para, em seguida, na camada de convolução ponto a ponto (*pointwise*), produzir a predição final da imagem.

3.4. Função de Perda

Normalmente, as funções de perda (*Loss Function*) para a tarefa de estimativa de profundidade são calculadas através da diferença entre o mapa de estimativa de profundidade verdadeiro (*Ground Truth* - GT), y, e o mapa de estimativa de profundidade predito pela rede, $\hat{y}$. Neste trabalho, é adotada a função de perda aplicada nos trabalhos [Alhashim and Wonka 2019] e [Rudolph et al. 2022], que consiste em realizar a soma ponderada de três funções de perda, apresentadas a seguir:

1. Erro absoluto médio (*Mean Absolute Error* - MAE), definido na Equação 1:

$$L_{depth}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |y_p - \hat{y}_p| \tag{1}$$

2. A Similidade Estrutural (*Structural Similarity Index Measure* - SSIM) é uma métrica normalmente utilizada para tarefas de reconstrução de imagens, conforme definido por [Wang et al. 2004]. Ela é definida na Equação2:

$$L_{SSIM}(y, \hat{y}) = \frac{1 - SSIM(y, \hat{y})}{2} \tag{2}$$

3. Perda de gradiente da imagem (*Gradient Loss*) que ajuda o modelo a aprender sobre as bordas, definida na Equação 3:

$$L_{grad}(y, \hat{y}) = \frac{1}{n} \sum_{p}^{n} |\mathbf{g_x}(y_p - \hat{y}_p)| + |\mathbf{g_y}(y_p - \hat{y}_p)| \tag{3}$$

A soma de todas as 3 funções de perda resulta na Equação 4 que calcula função de perda principal:

$$L(y, \hat{y}) = \lambda L_{depth}(y, \hat{y}) + L_{grad}(y, \hat{y}) + L_{SSIM}(y, \hat{y}) \tag{4}$$

O valor de λ considerado é 0,1, conforme proposto por [Alhashim and Wonka 2019], definindo a magnitude do SSIM para o cálculo da perda geral; e o N representa a quantidade total de pixels das imagens.

Assim, a função de perda final considera as três funções de perda, ajudando a reduzir a diferença entre a estimativa de profundidade obtida e o GT.

4. Experimentos e Resultados

4.1. Base de Dados NYU Depth V2

Os experimentos aqui apresentados foram realizados com base na versão reduzida da base de dados NYU Depth V2 [Silberman et al. 2012], conforme proposto por [Alhashim and Wonka 2019]. Esta versão contém 50.688 imagens para treino e 654 imagens para teste. A separação de treino e teste foi estabelecida previamente na versão de [Alhashim and Wonka 2019] do conjunto de dados. Durante as fases de treinamento e teste, as imagens de entrada da rede foram pré-processadas da seguinte forma:

- Inicialmente, a imagem de entrada e o GT foram redimensionados para a resolução 320x240 por meio da interpolação bilinear, visando aumentar a eficiência da rede.
- Em seguida, o GT e a predição foram cortadas conforme proposto por [Eigen et al. 2014], a fim de remover os ruídos da base de dados, tais como: os valores dos píxeis além do limite definido e as ”molduras”que todas as imagens apresentam.
- Após a realização da predição pela rede, a imagem prevista foi redimensionada de volta para a resolução original utilizando interpolação bilinear. Esse passo é importante para a comparação da predição com o *ground truth* original e para a avaliação da acurácia do modelo.

4.2. Detalhes da Implementação

O modelo proposto foi treinado e validado em um computador com as seguintes especificações: placa de vídeo RTX 2060 6GB, memória RAM 16GB, processador Intel Core i9-9900KF 3.60GHz, utilizando o framework Pytorch 1.12.1, linguagem Python 3.8.10 e sistema operacional Ubuntu 20.04.

Para o codificador, foi utilizada a rede MobileNetV2 pré-treinada, mas o modelo completo foi treinado com novos exemplos. Os demais blocos convolucionais foram iniciados com pesos aleatórios. O *batchsize* foi definido como 6.

No treinamento, foi utilizado o otimizador Adam, com $\beta 1 = 0.9$, $\beta 2 = 0.999$. A arquitetura foi treinada com 20 épocas, utilizando um taxa de aprendizado variante de $10e^{-4}$, que foi reduzida para $10e^{-5}$ após 15 épocas. Foram utilizadas, também, técnicas de aumento de dados (*data augmentation*) como espelhamento horizontal (50% probabilidade) e troca de canais de cores (25% probabilidade) das imagens de treinamento. Tais modificações das imagens na parte de treino e teste foram adotados pelos trabalhos [Alhashim and Wonka 2019, Rudolph et al. 2022].

Os resultados da rede foram normalizados da seguinte forma: $y = \frac{depthMax}{yOriginal}$, sendo depthMax, o valor máximo de profundidade encontrado na base de dados, que no caso da base de dados NYU Depth V2 é de 10 metros.

4.3. Métricas

As métricas de validação propostas por [Eigen et al. 2014], apresentadas nas equações 5, 6, 7 e 8, foram empregadas:

- *Root-mean-square-error* (RMSE):

$$RMSE = \sqrt{\frac{1}{N}\sum[\hat{y} - y]^2} \tag{5}$$

- *Mean absolute relative error* (REL):

$$REL = \frac{1}{N}\sum\left(\frac{|\hat{y} - y|}{y}\right) \tag{6}$$

- *Scale-invariant error* (log10):

$$log10 = \frac{1}{N}\sum|log_{10}(y_i) - log_{10}(\hat{y}_i)| \tag{7}$$

- *Deltas thresholds* (δ_i):

$$\delta_j : \% \text{ of } \hat{i} \text{ s.t. } \max\left(\frac{y}{\hat{y}}, \frac{\hat{y}}{y}\right) < 1.25^j, j \in \{1, 2, 3\} \tag{8}$$

considerando y_i como o píxel do GT y; $\hat{y}_i$ como o píxel da predição do GT $\hat{y}$; N como a quantidade total de píxeis; e δ_j como a porcentagem de píxeis com erro relativo abaixo do limiar controlado pela constante j.

4.4. Resultados

A Tabela 1 apresenta os resultados obtidos na base de dados NYU Depth V2 pela abordagem proposta, bem como por outros métodos de estimativa de profundidade monocular presentes na literatura. Os trabalhos utilizados na comparação, em sua maioria, foram treinados e testados em uma resolução reduzida de 320x240 píxeis. Os resultados do nosso trabalho superaram os demais em todas as métricas avaliadas, demonstrando um melhor desempenho.

A abordagem aqui empregada, que utiliza 5 milhões de parâmetros, conseguiu superar até mesmo os casos em esse valor é significativamente maior. Isso inclui o trabalho de [Wang and Zhu 2022], que usa aproximadamente o dobro de parâmetros, e o trabalho de [Dong et al. 2021], que usa aproximadamente quatro vezes mais parâmetros. Já o trabalho de [Wofk et al. 2019], que utiliza um menor número de parâmetros (3,9 milhões), alcançou resultados inferiores, principalmente para o δ e RMSE.

Para a visualização dos resultados alcançados, foi realizado um pós-processamento com a colorização da imagem de saída da rede, a fim de facilitar o compreendimento da estimativa de profundidade obtida. Exemplos das imagens resultantes são apresentados na Figura 2, onde as colunas, da esquerda para direita, correspondem à imagem de entrada da rede, à imagem GT da estimativa e à estimativa da rede, respectivamente.

Tabela 1. Comparação entre diferentes métodos avaliados na base de dados NYU Depth v2.

Resolution	Method	Parametros	Abs Rel ↓	RMSE ↓	log10 ↓	δ_1 ↑	δ_2 ↑	δ_3 ↑
320x240	[Dong et al. 2021]	24,95M	0,146	0,537	-	0,799	0,951	0,988
320x240	[Tu et al. 2021]	5,7M	0,147	0,531	0,062	0,801	0,956	0,989
320x240	[Wofk et al. 2019]	**3,9M**	0,165	0,576	0,067	0,777	0,949	0,987
320x240	[Rudolph et al. 2022]	5,7M	0,144	0,514	0,06	0,812	0,958	0,989
320x240	[Eigen et al. 2014]	-	0,215	0,907	-	0,611	0,887	0,971
320x240	[Eigen and Fergus 2015]	-	0,158	0,641	-	0,769	0,950	0,988
320x240	[Peng Wang et al. 2015]	-	0,22	0,745	0,094	0,605	0,890	0,970
320x240	[Li et al. 2017]	-	0,152	0,611	0,064	0,789	0,955	0,988
128x128	[Wang and Zhu 2022]	14,9M	0,214	0,760	0,087	0,663	0,900	0,973
320x240	[Xu et al. 2017]	-	0,139	0,609	0,063	0,793	0,948	0,984
320x240	Arquitetura Proposta	5M	**0,138**	**0,497**	**0,059**	**0,820**	**0,961**	**0,990**

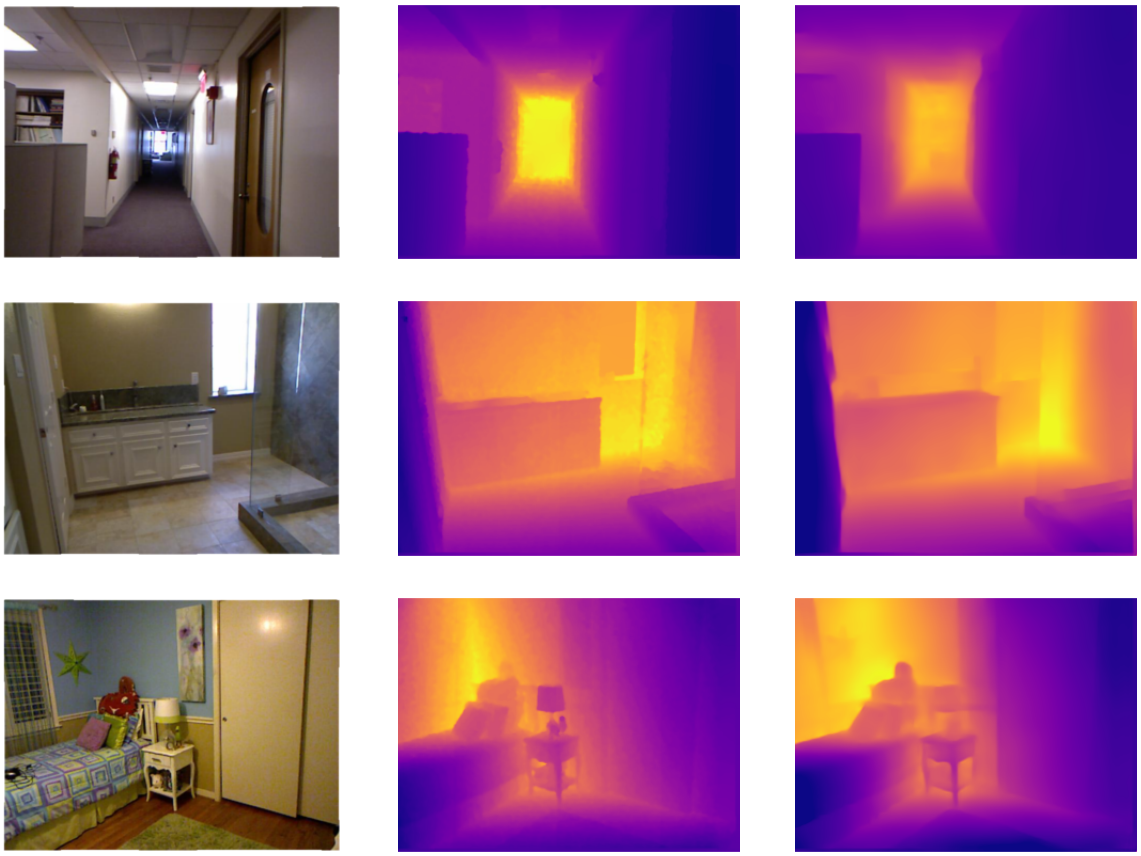

Figura 2. Visualização dos resultados obtidos pela abordagem proposta com efeitos visuais. (a) Imagem RGB; (b) GT; (c) Predição da rede.

5. Conclusão

Neste trabalho foi proposta uma arquitetura para estimativa de profundidade monocular que se destaca por sua menor complexidade, com apenas 5 milhões de parâmetros. A abordagem utiliza a arquitetura UNet++ e a rede MobileNetV2 como codificador. Os resultados obtidos na base de dados NYU Depth v2 demonstraram a competitividade do método, mesmo quando comparado com abordagens mais complexas. Essa arquitetura mais eficiente é especialmente adequada para cenários onde o recurso computacional é um fator importante. Como trabalhos futuros, pretende-se validar o modelo em ambientes de recursos computacionais limitados e otimizar a rede por meio de técnicas como *pruning*, *compression* e *quantization*.

6. Agradecimentos

Os autores agradecem ao apoio do IFES, à FAPES/CAPES via PDPG (Programa de Desenvolvimento de Pós-Graduação - Parcerias Estratégicas nos Estados, proc 2021-2S6CD, FAPES nº132/2021), à FAPES pelo projeto nº1023/2022 P:2022-8TZV6 e ao CNPq (projeto 407742/2022-0). O autor Luiz Antonio agradece à FAPES pela bolsa.

Referências

Agarwal, A. and Arora, C. (2022). Attention Attention Everywhere: Monocular Depth Prediction with Skip Attention.

Alhashim, I. and Wonka, P. (2019). High Quality Monocular Depth Estimation via Transfer Learning.

Ali, U., Bayramli, B., Alsarhan, T., and Lu, H. (2021). A lightweight network for monocular depth estimation with decoupled body and edge supervision. *Image and Vision Computing*, 113:104261.

Bhat, S. F., Alhashim, I., and Wonka, P. (2022). LocalBins: Improving Depth Estimation by Learning Local Distributions.

Cantrell, K., Miller, C., and Morato, C. (2020). Practical Depth Estimation with Image Segmentation and Serial U-Nets:. In *Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems*, pages 406–414, Prague, Czech Republic. SCITEPRESS - Science and Technology Publications.

Chen, L., Tang, W., John, N. W., Wan, T. R., and Zhang, J. J. (2017). Augmented Reality for Depth Cues in Monocular Minimally Invasive Surgery. *arXiv:1703.01243 [cs]*.

de Queiroz Mendes, R., Ribeiro, E. G., dos Santos Rosa, N., and Grassi, V. (2021). On deep learning techniques to boost monocular depth estimation for autonomous navigation. *Robotics and Autonomous Systems*, 136:103701.

Deng, Y., Xiao, J., and Zhou, S. Z. (2021). A lightweight real-time stereo depth estimation network with dynamic upsampling modules. In *VISIGRAPP*.

Dong, X., Garratt, M. A., Anavatti, S. G., and Abbass, H. A. (2021). MobileXNet: An Efficient Convolutional Neural Network for Monocular Depth Estimation.

Eigen, D. and Fergus, R. (2015). Predicting Depth, Surface Normals and Semantic Labels with a Common Multi-scale Convolutional Architecture. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2650–2658, Santiago, Chile. IEEE.

Eigen, D., Puhrsch, C., and Fergus, R. (2014). Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. *arXiv:1406.2283 [cs]*.

Han, K., Wang, Y., Tian, Q., Guo, J., Xu, C., and Xu, C. (2020). Ghostnet: More features from cheap operations. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1577–1586.

He, K., Zhang, X., Ren, S., and Sun, J. (2016). Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., and Adam, H. (2017). MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv:1704.04861 [cs]*.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017). Densely connected convolutional networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2261–2269.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., and Keutzer, K. (2016). Squeezenet: Alexnet-level accuracy with 50x fewer parameters and lt;0.5mb model size.

Li, J., Klein, R., and Yao, A. (2017). A Two-Streamed Network for Estimating Fine-Scaled Depth Maps from Single RGB Images.

Liu, M. and Zhu, M. (2018). Mobile Video Object Detection with Temporally-Aware Feature Maps. *arXiv:1711.06368 [cs]*.

Liu, X., Sinha, A., Ishii, M., Hager, G. D., Reiter, A., Taylor, R. H., and Unberath, M. (2020). Dense Depth Estimation in Monocular Endoscopy With Self-Supervised Learning Methods. *IEEE Transactions on Medical Imaging*, 39(5):1438–1447.

Luo, X., Huang, J.-B., Szeliski, R., Matzen, K., and Kopf, J. (2020). Consistent Video Depth Estimation. *arXiv:2004.15021 [cs]*.

Ma, F. and Karaman, S. (2018). Sparse-to-Dense: Depth Prediction from Sparse Depth Samples and a Single Image. *arXiv:1709.07492 [cs]*.

Mancini, M., Costante, G., Valigi, P., Ciarfuglia, T. A., Delmerico, J., and Scaramuzza, D. (2017). Toward Domain Independence for Learning-Based Monocular Depth Estimation. *IEEE Robotics and Automation Letters*, 2(3):1778–1785.

Ming, Y., Meng, X., Fan, C., and Yu, H. (2021). Deep learning for monocular depth estimation: A review. *Neurocomputing*, 438:14–33.

Peng Wang, Xiaohui Shen, Zhe Lin, Cohen, S., Price, B., and Yuille, A. (2015). Towards unified depth and semantic prediction from a single image. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2800–2809, Boston, MA, USA. IEEE.

Ramamonjisoa, M. and Lepetit, V. (2019). Sharpnet: Fast and accurate recovery of occluding contours in monocular depth estimation.

Ranftl, R., Bochkovskiy, A., and Koltun, V. (2021). Vision Transformers for Dense Prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12179–12188.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation.

Rudolph, M., Dawoud, Y., Güldenring, R., Nalpantidis, L., and Belagiannis, V. (2022). Lightweight Monocular Depth Estimation through Guided Decoding. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2344–2350.

Sandler, M., Howard, A. G., Zhu, M., Zhmoginov, A., and Chen, L. (2018). Mobilenetv2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4510–4520. Computer Vision Foundation / IEEE Computer Society.

Silberman, N., Hoiem, D., Kohli, P., and Fergus, R. (2012). Indoor segmentation and support inference from rgbd images. In Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., and Schmid, C., editors, *Computer Vision – ECCV 2012*, pages 746–760, Berlin, Heidelberg. Springer Berlin Heidelberg.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In Bengio, Y. and LeCun, Y., editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

Tu, X., Xu, C., Liu, S., Li, R., Xie, G., Huang, J., and Yang, L. T. (2021). Efficient Monocular Depth Estimation for Edge Devices in Internet of Things. *IEEE Transactions on Industrial Informatics*, 17(4):2821–2832.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Wang, Y. and Zhu, H. (2022). Monocular Depth Estimation: Lightweight Convolutional and Matrix Capsule Feature-Fusion Network. *Sensors*, 22(17):6344.

Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. (2004). Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Transactions on Image Processing*, 13(4):600–612.

Wofk, D., Ma, F., Yang, T.-J., Karaman, S., and Sze, V. (2019). Fastdepth: Fast monocular depth estimation on embedded systems. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6101–6108.

Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., and Keutzer, K. (2019). Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 10734–10742. Computer Vision Foundation / IEEE.

Xu, D., Ricci, E., Ouyang, W., Wang, X., and Sebe, N. (2017). Multi-scale Continuous CRFs as Sequential Deep Networks for Monocular Depth Estimation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 161–169, Honolulu, HI. IEEE.

Xu, D., Wang, W., Tang, H., Liu, H., Sebe, N., and Ricci, E. (2018). Structured Attention Guided Convolutional Neural Fields for Monocular Depth Estimation.

Zhang, H., Shen, C., Li, Y., Cao, Y., Liu, Y., and Yan, Y. (2019). Exploiting temporal consistency for real-time video depth estimation. *arXiv:1908.03706 [cs]*.

Zhang, X., Zhou, X., Lin, M., and Sun, J. (2017). Shufflenet: An extremely efficient convolutional neural network for mobile devices. *CoRR*, abs/1707.01083.

Zhao, C., Sun, Q., Zhang, C., Tang, Y., and Qian, F. (2020). Monocular depth estimation based on deep learning: An overview. *Science China Technological Sciences*, 63(9):1612–1627.

Zheng, Q., Yu, T., and Wang, F. (2023). Dcu-net: Self-supervised monocular depth estimation based on densely connected u-shaped convolutional neural networks. *Computers Graphics*, 111:145–154.

Zhou, Z., Rahman Siddiquee, M., Tajbakhsh, N., and Liang, J. (2018). Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support - 4th International Workshop, DLMIA 2018 and 8th International Workshop, ML-CDS 2018 Held in Conjunction with MICCAI 2018*, Lecture Notes in Computer Science, pages 3–11. Springer Verlag.